SYBASE iAnywhere

SQL Anywhere® 12
変更点とアップグレード

バージョン 12.0.1

2012 年 1 月

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 12.0.1
2012 年 1 月

Copyright © 2012 iAnywhere Solutions, Inc. Portions copyright © 2012 Sybase, Inc. All rights reserved.

iAnywhere との間に書面による合意がないかぎり、このマニュアルは現状のまま提供されるものであり、その使用または記載内容の誤りに対して一切の責任を負いません。

次の条件に従うかぎり、このマニュアルの一部または全体を使用、印刷、複製、配布することができます。1) マニュアルの一部または全体にかかわらず、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) iAnywhere 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。

iAnywhere®、Sybase®、http://www.sybase.com/detail?id=1011207 に示す商標は Sybase, Inc. またはその関連会社の商標です。® は米国での登録商標を示します。

このマニュアルに記載されているその他の会社名と製品名は各社の商標である場合があります。

目次

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

はじめに

このマニュアルでは、SQL Anywhere 12 とそれ以前のバージョンに含まれる新機能について説明します。

SQL Anywhere のバージョン 10 より前における新機能や動作の変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

バージョン 12.0.1 の新機能

バージョン 10 以前の SQL Anywhere の新機能と動作変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

12.0.1 での追加事項：新機能と動作変更

「SQL Anywhere 12.0.1 のマニュアルの更新：」このリリースの SQL Anywhere 12.0.1 マニュアルは、2011 年 2 月 15 日の SQL Anywhere 12.0.1 の初回のマニュアルリリース時のバージョンよりも更新されています。このマニュアルには、EBF 3554 (Windows) までの新機能と動作変更が含まれています。特に、SQL Anywhere OnDemand Edition のクラウドの新機能をサポートするために必要な変更も含んでいます。

SQL Anywhere OnDemand Edition に対するサポートの追加

SQL Anywhere OnDemand Edition とは、独立系ソフトウェアベンダー (ISV) がビジネスアプリケーションをクラウドに移行し、ソフトウェアを Software as a Service (SaaS) として提供できるようにしたものです。ISV は、大規模なクラウドアプリケーションを構築、配備および管理できるようになります。

SQL Anywhere OnDemand Edition をサポートするよう、SQL Anywhere にはいくつかの拡張および動作変更が行われています。次に主要な変更点を示します。これ以外の変更の詳細は、http://dcx.sybase.com/cloud100 の SQL Anywhere OnDemand Edition のマニュアルを参照してください。

- **クラウド内のデータベース用の新しいログインモード**　login_mode データベースオプションに、CloudAdmin という新しい値が追加されました。このログインモードは、クラウドの内部で使用します。「login_mode オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Sybase Central と Interactive SQL の [接続] ウィンドウの拡張**　Sybase Central と Interactive SQL の **[接続]** ウィンドウに、クラウド内のテナントデータベースに接続するために、**[クラウドのコンピューターのデータベースに接続]** オプションが用意されました。

- **[SQL Anywhere の ODBC 設定] ウィンドウの拡張**　ODBC データソースアドミニストレーターの **[SQL Anywhere の ODBC 設定]** ウィンドウに、クラウド内のテナントデータベースに接続するために、**[クラウドのコンピューターのデータベースに接続]** オプションが用意されました。

- **NodeType (NODE) 接続パラメーターの強化**　NodeType 接続値で、MIRROR 値と READONLY 値を新規にサポートするようになりました。

  MIRROR　テナントのミラーデータベースにアプリケーションをリダイレクトします。

  READONLY　すべての読み込み専用データベースコピー (コピーノードまたはミラーコピー) にアプリケーションをリダイレクトします。読み込み専用コピーノードがない場合は、READONLY は MIRROR 設定と同様になります。読み込み専用コピーノードのみある場合は、READONLY は COPY 設定と同様になります。

「NodeType (NODE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい NodeType (NODE) 接続パラメーターの動作変更**　クラウド内のデータベースに接続する際の NodeType 接続パラメーターのデフォルト値は PRIMARY です。これでは、テナントのプライマリデータベースへのリダイレクトが自動的に実行されます。NodeType を DIRECT に設定した場合、リダイレクトはされず、データベースが指定されたホストで実行されている場合のみ接続が成功します。

  「NodeType (NODE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **クラウドデータベースへの接続時に必要な DatabaseName 接続パラメーター**　クラウド内のデータベースに接続するときは、DatabaseName 接続パラメーターを指定する必要があります。クラウド内のデータベースにはデフォルトのデータベース名はありません。「DatabaseName (DBN) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ServerName 接続パラメーターのクラウド内での制限**　クラウド内のデータベースに接続する際、NodeType 接続パラメーターが DIRECT に設定されているときは、ServerName 接続パラメーターしか使用できません。多くの場合、クラウドサーバーへの接続時には、ServerName または NodeType または DIRECT は指定しないようにする必要があります。「ServerName (Server) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL 文の強化**　クラウド内のテナントデータベースに対して使用しやすくするために、次の文が強化されています。

  - 「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』
  - 「ALTER SYNCHRONIZATION USER 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』
  - 「START SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』

  クラウド内でデータベースが実行されている場合、一部の SQL Anywhere 文を使用できなかったり、制限がある場合があります。この情報については、それぞれのファンクションとシステムプロシージャーの説明で説明しています。

- **システムプロシージャーとファンクションをクラウド内のデータベースに実行した場合の結果の制限**　データベースがクラウド内で実行されている場合、一部の SQL Anywhere ファンクションとシステムプロシージャーでは、現在のテナントデータベースに関する情報しか返されません。同じクラウドサーバー上で実行される別のテナントデータベースに関する情報は示されません。この情報については、それぞれのファンクションとシステムプロシージャーの説明で説明しています。

- **クラウドで使用できない一部のデータベースオプション、サーバーオプション、接続パラメーター**　一部のデータベースオプション、サーバーオプション、接続パラメーターは、クラウドでは使用できません。詳細については、それぞれのオプションまたはパラメーターの説明で説明しています。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

SQL Anywhere のその他の新機能と動作変更

- **IPv6 アドレスの指定**　ポート番号を含む IPv6 アドレスは、角カッコまたはカッコで囲む必要があります。「SQL Anywhere での IPv6 サポート」『SQL Anywhere サーバー データベース管理』を参照してください。

- **テーブルのロードの強化**　以前のリリースでは、読み込み専用データベースのテンポラリテーブルのテーブルのロード操作は実行できませんでした。この制限は取り除かれました。

  以前、プライマリ、ミラーまたはルートノードサーバー上で実行されるデータベースに対する LOAD TABLE のデフォルトは、WITH FILE NAME LOGGING でした。現在、これらのデータベースタイプに対する LOAD TABLE ログのデフォルトは WITH ROW LOGGING になりました。

- **データベーススキーマの比較と同一化に対するサポートの追加**　Sybase Central を使用して、2 つのデータベースを比較できます。この比較により SQL 文を生成して、2 つのデータベース間の違いを判別できます。その SQL 文を実行し、1 つのデータベースを別のデータベースと同じにできます。「データベーススキーマの比較」『SQL Anywhere サーバー データベース管理』と「もう一方と一致させるためのデータベーススキーマの変換」『SQL Anywhere サーバー データベース管理』を参照してください。

- **DBA ユーザーのアカウントのロックアウト**　不正なパスワードの入力が、ログインポリシーで定義した接続試行の失敗回数を超過した場合、DBA ユーザーのアカウントは 1 分間ロックされます。データベースから無期限にロックアウトされることはありません。「DBA 認証」『SQL Anywhere サーバー データベース管理』を参照してください。

- ***OEM.ini* ファイルの新しいディレクトリオプション**　OEM.ini ファイルの **[preferences]** セクションの **directory** オプションでは、管理ツールによって使用される、ユーザー専用の設定ファイルを保存するディレクトリを指定します。これらのファイルには、管理ツールの設定と履歴に関する情報が含まれます。ディレクトリ名は、デリミターで終了しない完全修飾されたものを指定する必要があります。「管理ツールの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Timeout、SendBufferSize、ReceiveBufferSize TCP プロトコルオプションの新しい制限**　TCP/IP を使用する際、次のプロトコルオプションに上限が追加されました。

  - **ReceiveBufferSize (RCVBUFSZ) プロトコルオプション**　TCP/IP プロトコルスタックで使用されるバッファーに指定できるサイズの上限は 1 MB です。

  - **SendBufferSize (SNDBUFSZ) プロトコルオプション**　TCP/IP プロトコルスタックで使用されるバッファーに指定できるサイズの上限は 1 MB です。

  - **Timeout (TO) プロトコルオプション**　通信の確立 (TCP/IP) 時の応答の待機に指定できる最大時間は 3600 秒です。

  この上限を超える値を指定すると、接続エラーが返されます。「接続エラー : %1」『エラーメッセージ』を参照してください。

- **データベースミラーリングの強化**　-xa サーバーオプションで、次の値がサポートされるようになりました。

○**DBN=*** の指定は、データベースがサーバーを監視サーバーとして使用できることを意味します。

○認証文字列を省略すると、ミラーサーバーが提供する認証文字列を検証しないようにできます。

○認証文字列を 1 つのみ指定した場合、すべてのデータベースでその認証文字列が使用されることを意味します。

「-xa dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

高可用性の読み込み専用スケールアウトシステムでは、コピーノードを監視サーバーとして使用できます。この機能を使用するには、データベースミラリングシステム内のすべてのサーバーをアップグレードする必要があります。「監視サーバーとしてコピーノードを使用する」『SQL Anywhere サーバー データベース管理』を参照してください。

- **文のパフォーマンスの向上**　ストアドプロシージャーまたはユーザー定義関数を使用する文のパフォーマンスが向上しました。

- **URL の不明な HTTP メソッドを示す LogFormat (LF) プロトコルオプション**　HTTP 要求メソッドがサポートされていなかったり、URI のフォーマットが不正であるか必要なデータベース名がないために Web 要求が失敗した場合は、HTTP メソッド (@M) と HTTP バージョン (@V) では、文字列 **???** を返し、URI (@U) が **>>>** に続いて指定された要求を返します。「LogFormat (LF) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいプロパティ**　次の接続プロパティが追加されました。

  ○ LastCommitRedoPos

  次のデータベースプロパティが追加されました。

  ○ LastCommitRedoPos
  ○ LastWrittenRedoPos
  ○ LastSyncedRedoPos

  次のデータベースサーバープロパティが追加されました。

  ○ ProcessID

- **ストアドプロシージャーまたはユーザー定義関数を使用する文のパフォーマンスの向上**　データベースサーバーがユーザー定義プロシージャーおよび関数を処理する方法が向上し、パフォーマンスが向上しました。このパフォーマンスの向上は、プロシージャーの数とユーザー定義関数の呼び出し数に依存します。

- **ALTER SERVER 文の動作変更**　以前のリリースでは、ALTER SERVER 文のすべての句でオートコミットが実行されていました。現在では、CONNECTION CLOSE 句では、オートコミットは実行されません。「ALTER SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **OPENSTRING 句の強化**　FROM 句の OPENSTRING 句で [NOT | AUTO] COMPRESSED オプションと ENCRYPTED オプションがサポートされるようになりました。ENCRYPTED オプションを使用すると、ENCRYPTED 句を指定したときに UNLOAD 文によって作成されたファイルを読み込むことができます。 「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **証明書作成ユーティリティ (createcert) の強化**　証明書作成ユーティリティに多数のオプションが追加され、簡単に証明書が作成できるようになりました。「証明書作成ユーティリティ (createcert)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **クエリプラン文字列の強化**　プランには、文の既知の値が含まれるようになりました。長いプランには、さらに文のキャッシュプランの情報も含まれるようになりました。

  アプリケーションプロファイリングを使用する場合、長いプランには、クエリの最適化の方法と、部分インデックススキャンで使用される述部に関する情報が追加されています。

- **クエリ最適化**　検索引数可能な IN 述部の推定 (AND 述部に変換できない OR 述部からの部分インデックススキャンに使用可能) に最適化が追加されました。 「クエリ処理中に実行される最適化」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ADO.NET Entity Framework 4.2 に追加されたサポート**　SQL Anywhere で、ADO.NET Entity Framework 4.2 Code First 機能がサポートされるようになりました。「Entity Framework のサポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SetupVSPackage .NET アセンブリインストーラーの新しい salocation オプション**　SetupVSPackage アプリケーションで salocation オプションを使用し、SQL Anywhere のインストール場所を指定できるようになりました。「.NET クライアントの配備」『SQL Anywhere サーバー プログラミング』を参照してください。

- **JDBC 4.0 ドライバーの OSGi バンドルに対するサポートの追加**　SQL Anywhere の JDBC 4.0 ドライバー (sajdbc4.jar) に適切なマニフェスト情報が含まれ、OSGi (Open Services Gateway initiative) バンドルとしてロードできるようになりました。「JDBC サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

SQL Remote の動作の変更

SQL Remote では、リモートデータベースの SQL Remote メッセージのメッセージシステムとして HTTP/HTTPS がサポートされています。データベースを設定するには、SET REMOTE OPTION コマンドを使用します。

次の項を参照してください。

- 「HTTP メッセージシステム」『SQL Remote』
- 「SET REMOTE OPTION 文 [SQL Remote]」『SQL Remote』
- 「チュートリアル：HTTP メッセージシステムを使用したレプリケーションシステムの設定」『SQL Remote』
- 「チュートリアル：統合データベースをメッセージサーバーとして HTTP メッセージシステムを使用したレプリケーションシステムの設定」『SQL Remote』
- 「チュートリアル：HTTP メッセージシステムを使用し、Relay Server を経由して統合データベースをメッセージサーバーとしたレプリケーションシステムの設定」『SQL Remote』

Ultra Light の動作の変更

- **ALTER TABLE ADD CONSTRAINT でのユニークな制約の追加時の NULL 入力不可カラムの指定**　ALTER TABLE 文の向上により、新しいユニークな制約を追加するとき、すべての制約カラムを NULL 入力不可にしなければならなくなりました。

  「ALTER TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Android スマートフォンでの Mobile Link ファイル転送の新規サポート**　DatabaseManager.createFileTransfer メソッドを使用して、Android クライアントと Mobile Link サーバー間でファイルを転送できるようになりました。

  次の項を参照してください。

  - DatabaseManager.createFileTransfer メソッド [Ultra Light J]『Ultra Light - Java プログラミング』
  - FileTransfer インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』
  - FileTransferProgressListener インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』
  - FileTransferProgressData インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』

- **Android スマートフォンでの ZLIB 圧縮の新規サポート**　HTTPS プロトコルを介して Ultra Light J API で Android スマートフォンと Mobile Link サーバーの同期を取る際、ZLIB データ圧縮がサポートされるようになりました。

  Android スマートフォンで ZLIB 圧縮を許可するには、次のメソッドを利用します。

  - StreamHTTPParms.setZlibCompression
  - StreamHTTPParms.setZlibDownloadWindowSize
  - StreamHTTPParms.setZlibUploadWindowSize
  - StreamHTTPParms.zlibCompressionEnabled

  次の項を参照してください。

  - 「Mobile Link クライアントネットワークプロトコルオプション」『Mobile Link クライアント管理』
  - StreamHTTPParms インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』

- **Android スマートフォンでの End-to-end 暗号化 (E2EE) の新規サポート**　HTTPS プロトコルを介して Ultra Light J API で Android スマートフォンと Mobile Link サーバーの同期を取る際、End-to-end 暗号化がサポートされるようになりました。

  Android スマートフォンで E2EE をサポートするには、次のメソッドを利用します。

  - StreamHTTPParms.getE2eePublicKey
  - StreamHTTPParms.setE2eePublicKey

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

次の項を参照してください。

- 「Mobile Link クライアントネットワークプロトコルオプション」『Mobile Link クライアント管理』
- StreamHTTPParms インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』

- **Android スマートフォンでのデフォルトの信用できる証明書ストアの信頼できる証明書への HTTPS プロトコルを介しての同期の新規サポート**　証明書は、次の優先度の規則に従って使用されます。

  1. StreamHTTPSParms.setTrustedCertificates メソッドが呼び出された場合、指定したファイルの証明書が使用されます。
  2. StreamHTTPSParms.setTrustedCertificates メソッドが呼び出されず、ulinit または ulload ユーティリティによって証明書がデータベースが設定されている場合、それらの証明書が使用されます。
  3. StreamHTTPSParms.setTrustedCertificates メソッド、ulinit または ulload ユーティリティのいずれによっても証明書が指定されず、Android を使用している場合は、証明書はオペレーティングシステムの信頼できる証明書ストアから読み込まれます。この証明書ストアは、Web サーバーを安全に管理するために HTTPS を介して接続するときに、Web ブラウザーで使用されます。

  次の項を参照してください。

  - StreamHTTPSParms.setTrustedCertificates メソッド [Ultra Light J]『Ultra Light - Java プログラミング』

- **Android スマートフォンでのカーソルの移動方法の新規サポート**　Ultra Light J API で、他の Ultra Light API で利用可能な SQL 結果セットのナビゲーションメソッドがすべてサポートされるようになりました。

  Android スマートフォンで、次のナビゲーションメソッドが利用できるようになりました。

  - ResultSet.afterLast
  - ResultSet.beforeFirst
  - ResultSet.first
  - ResultSet.getRowCount
  - ResultSet.last
  - ResultSet.relative

  次の項を参照してください。

  - 「SQL 結果セットのナビゲーション」『Ultra Light - Java プログラミング』
  - ResultSet インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』

- **Android スマートフォンでの BlackBerry 専用のいくつかの ResultSet メソッドのサポート**　BlackBerry スマートフォンでサポートされる文字列パラメーターを必要とする get メソッドが、Android スマートフォンで利用できるようになりました。

  Android で新規にサポートされるようになったメソッドは、次のとおりです。

  ○ResultSet.getBlobInputStream(String name)

  ○ResultSet.getBoolean(String name)

  ○ResultSet.getClobReader(String name)

  ○ResultSet.getBytes(String name)

  ○ResultSet.getDate(String name)

  ○ResultSet.getDecimalNumber(String name)

  ○ResultSet.getDouble(String name)

  ○ResultSet.getFloat(String name)

  ○ResultSet.getLong(String name)

  ○ResultSet.getSize(String name)

  ○ResultSet.getString(String name)

  ○ResultSet.getUUIDValue(String name)

  ○ResultSet.isNull(String name)

  ResultSet インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Android スマートフォンでの ResultSetMetaData メソッドの新規サポート**　Android スマートフォンで、次のメソッドと Blackberry スマートフォンに返される戻り値と非常に似た戻り値がサポートされるようになりました。

  ○ResultSetMetaData.getAliasName(int column_no)

  ○ResultSetMetaData.getDomainName(int column_no)

  ○ResultSetMetaData.getCorrelationName(int column_no)

  ○ResultSetMetaData.getQualifiedName(int column_no)

  ○ResultSetMetaData.getTableColumnName(int column_no)

  ○ResultSetMetaData.getTableName(int column_no)

  ○ResultSetMetaData.getWrittenName(int column_no)

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ResultSetMetadata インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Android スマートフォンでの DatabaseInfo.getPageSize メソッドの新規サポート**　DatabaseInfo.getPageSize メソッドによって Ultra Light データベースに対しクエリが実行され、ページサイズが取得されるようになりました。

  DatabaseInfo.getPageSize メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Android スマートフォンでの DatabaseInfo.getNumberRowsToUpload メソッドの新規サポート**　DatabaseInfo.getNumberRowsToUpload メソッドで、アップロードを待機している行数が返されるようになりました。

  DatabaseInfo.getNumberRowsToUpload メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Android スマートフォンに返される DatabaseInfo.getRelease 値へのビルド番号の新規追加**　DatabaseInfo.getRelease メソッドで、ソフトウェアの完全なリリース番号が返されます。

  DatabaseInfo.getRelease メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Android スマートフォンでの Extra Mobile Link クライアントネットワークプロトコルオプション設定の新規サポート**　Mobile Link クライアントネットワークプロトコルオプションのセミコロンで区切られたリストを指定および取得する StreamHTTPParms.getExtraParameters メソッドと StreamHTTPParms.setExtraParameters メソッドが追加されました。Android スマートフォンで以前にサポートされていなかった他のオプションもサポートするようになりました。

  次の項を参照してください。

  - StreamHTTPParms.getExtraParameters メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』
  - StreamHTTPParms.setExtraParameters メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』

- **DatabaseManager クラス Ultra Light J API でのエラーレポートの向上**　リリースメソッドの呼び出し後、connect または createDatabase メソッドが呼び出された場合に SQLE_NOT_CONNECTED エラーコードで ULjException error エラーがスローされるようになりました。

  DatabaseManager.release メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Ultra Light J API で利用可能になった再開可能な HTTP**　再開可能な HTTP が有効になっている場合、Ultra Light J では信頼性の低いネットワークでネットワークが頻繁に失敗しないよう、ネットワークの割り込みが許容されます。

次の項を参照してください。

- StreamHTTPParms.setRestartable メソッド [Ultra Light J]『Ultra Light - Java プログラミング』
- StreamHTTPParms.isRestartable メソッド [Ultra Light J]『Ultra Light - Java プログラミング』

- **Ultra Light J API での NULL 結果セットの処理の向上** ResultSet.getString メソッドで取得される NULL 値が、空の文字列ではなく NULL として返されるようになりました。

  ResultSet.getString メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **ロー制限により BlackBerry スマートフォンで常にインデックスの遅延ロードが有効** ロー制限が有効な状態でアクセスされる BlackBerry スマートフォン上のデータベースで、常にインデックスが遅延ロードされるようになりました。

  次の項を参照してください。

  - ConfigPersistent.setRowScoreFlushSize メソッド [Ultra Light J]『Ultra Light - Java プログラミング』
  - ConfigPersistent.setRowScoreMaximum メソッド [Ultra Light J]『Ultra Light - Java プログラミング』

- **BlackBerry Enterprise Server を介した BlackBerry スマートフォンの同期の安全性の向上** BlackBerry Enterprise Server (BES) と一部の BlackBerry スマートフォンの HTTPS プロトコルを介しての同期の失敗を避けるには、URL サフィックスに ";EndToEndRequired" の語を追加する必要がある場合があります。

  StreamHTTPParms.setURLSuffix メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Ultra Light Java Edition データベースロードユーティリティとその -z オプションでのデータベースページサイズの調整** Ultra Light Java Edition のデータベースのページサイズを、新しい -z オプションを使用してバイト単位で指定できるようになりました。

  「Ultra Light Java Edition データベースロードユーティリティ (ULjLoad)」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light for M-Business Anywhere での Connection.getNewUUID と UUID.toString メソッドのサポート** Connection.getNewUUID メソッドと UUID.toString メソッドが再度使用できるようになりました。

  次の項を参照してください。

  - Connection.getNewUUID メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』
  - UUID.toString メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』

- **Ultra Light C++ API でテーブルカラムの修飾された名前が取得可能** ul_column_name_type 列挙の ul_name_type_qualified 識別子を ULResultSetSchema.GetColumnName メソッドと使用し

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

た場合、ResultSet オブジェクトの関連付けられているカラムの修飾された名前を取得できます。

次の項を参照してください。

- ul_column_name_type 列挙体 [Ultra Light C++]『Ultra Light C/C++ プログラミング』
- ULResultSetSchema.GetColumnName メソッド [Ultra Light C++]『Ultra Light C/C++ プログラミング』

- **Ultra Light C++ API での Pocket PC 2003 プラットフォームの新規サポート**　Pocket PC 2003 で Ultra Light がサポートされるようになりました。TLS と HTTPS 間の同期での暗号化はサポートされていません。

- **Xcode 4.2 の CustDB と Names サンプルへのサポート**　Xcode 3.2 と 4.2 で CustDB サンプルと Names サンプルがサポートされるようになりました。

Mobile Link の新機能

- **Mobile Link サーバー API への NetworkData クラスの追加**　要求で使用される HTTP ヘッダーとクライアント側証明書にアクセスできるよう、Java と .NET スクリプト API に NetworkData クラスが追加されました。

  次の項を参照してください。

  - NetworkData インターフェイス [Mobile Link サーバー Java]『Mobile Link サーバー管理』
  - NetworkData インターフェイス [Mobile Link サーバー .NET]『Mobile Link サーバー管理』

- **mlsrv12 -x オプションへの collect_network_data プロトコルの新規追加**　各同期時に同期スクリプトでネットワーク情報を読み込むには、collect_network_data プロトコルオプションで mlsrv12 -x オプションを指定して使用します。「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

Mobile Link の動作の変更

- **同期されないカラムに対する更新**　同期されないカラムに対する更新 (同期モデルで定義したタイムスタンプカラムまたは論理削除カラムは除く) で、同期モデルでトリガーが使用された場合、タイムスタンプを維持するためにタイムスタンプカラムが更新されなくなりました。

- **テーブルマッピングの方向メニュー**　マッピングの方向を "Not synchronized" に変更した場合、テーブルマッピングの方向メニューは有効のままとなります。「テーブルマッピングとカラムマッピングの変更」『Mobile Link クイックスタート』を参照してください。

- **mlsis と dbmlsync の -qi オプション**　mlsis と dbmlsync の使用法に -qi オプションが追加されました。「-qi dblsn オプション」『Mobile Link サーバー起動同期』と「-qi dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **mlreplay の新しいエラーメッセージ**　Mobile Link リプレイユーティリティに次のエラーメッセージが追加されました。

  - **MLGENREPLAYAPI_FAILED_TO_DETERMINE_CLIENT_TYPE (-5062)**　記録されたプロトコルが dbmlsync クライアントからのものであるか判断できません。

- ○ **MLREPLAY_INCOMPATIBLE_RECORDED_PROTOCOL_VERSION (-5075)**　記録された %1 は、このバージョンの mlreplay には対応しないプロトコルバージョンです。
- ○ **MLGENREPLAYAPI_INCOMPATIBLE_RECORDED_PROTOCOL_VERSION (-5076)**　記録された %1 は、このバージョンの Replay API generator には対応しないプロトコルバージョンです。

- **dbmlsync のロギングの強化**　Dbmlsync で、実行しているオペレーティングシステムとマシンのアーキテクチャに関する情報をログに出力できるようになりました。また、実行プログラムがコンパイルされたプラットフォームも出力されます。「SQL Anywhere クライアントのログ」『Mobile Link クライアント管理』を参照してください。

- **SHA2 証明書に対するサポート**　SHA2 を使用して署名された証明書が、Mobile Link サーバーと Mobile Link クライアントで新規にサポートされるようになりました。「trusted_certificate」『Mobile Link クライアント管理』と「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

Relay Server の新機能

- **Windows IIS6、IIS7、Apache への Relay Server のクイック配備**　Relay Server には、Windows IIS 6.0、Windows IIS 7.0 または 7.5 と Apache の配備を段階的に処理する一連のユーティリティが用意されています。詳細については、次の項を参照してください。
  - ○「Windows Server 2003 上の Microsoft IIS 6.0 への Relay Server コンポーネントの配備」『Relay Server』
  - ○「Windows Server 2008／Windows Server 2008 R2 上の Microsoft IIS 7.0 または 7.5 への Relay Server コンポーネントの配備」『Relay Server』
  - ○「Linux 上の Apache への Relay Server コンポーネントの配備」『Relay Server』

- **Relay Server での ID 転送のサポート**　SAP Gateway には、信頼できる仲介者を通じて転送される X.509 証明書など、クライアントを認証するいくつかの手段があります。Relay Server では、リモートクライアントからの ID 情報の転送と、HTTP ヘッダーを使用したの ID 情報の SAP NetWeaver Gateway または Web Dispatcher への転送が可能です。「バックエンドファームセクションのプロパティ」『Relay Server』を参照してください。

SQL Anywhere の新機能

次に、SQL Anywhere バージョン 12.0.1 の新機能を示します。

サポートされるプラットフォームのリストに加えられた変更については、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **空間データのサポートの強化**　SQL Anywhere の空間データサポートに以下の強化が行われました。

  パフォーマンスの改善　次の空間操作のパフォーマンスが大幅に改善されました。

  ●シェイプファイルのロード時間

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

●多角形と複数多角形および曲面の空間参照系に属する多角形を含むコレクションのロード

●複雑な多角形と複数多角形のロード (たとえば、多数のリングまたは多数の点を含むリングで定義される多角形)

●複雑なジオメトリに適用される ST_Union および ST_Intersection などの空間セット操作

●ST_Contains and ST_Intersects などの空間述部

●ジオメトリの 1 つが点である空間述部

●インデックス付きの曲面ジオメトリカラムに ST_WithinDistance または ST_Distance を適用する空間述部

Interactive SQL: インポートウィザードでシェイプファイルをサポート　インポートウィザードに ESRI シェイプファイルをインポートするオプションが含まれるようになりました。「インポートウィザード (Interactive SQL) でのデータのインポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

Interactive SQL: INPUT 文の新しい FORMAT SHAPEFILE および SRID 句　INPUT 文で FORMAT SHAPEFIL 句を使用した ESRI シェイプファイルのロードをサポートするようになりました。INPUT 文にシェイプファイルをロードするときに使用する SRID を指定する SRID 句も追加されました。「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

新しい st_geometry_load_shapefile システムプロシージャー　st_geometry_load_shapefile システムプロシージャーを使用すれば、ファイルの名前、データをロードするために使用する SRID およびデータを作成してロードするテーブルの名前を提供することによって ESRI シェイプファイルをロードできます。テーブルのカラムは、シェイプファイルで指定されたカラム名から取得されます。「st_geometry_load_shapefile システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

> **注意**
> この新しいストアドプロシージャーにアクセスするには、データベースをアップグレードする必要があります。

ST_CircularString 補間許容値を指定する新しいデータベースオプション　ST_CircularString ジオメトリの補間を制御するために st_geometry_interpolation オプションが追加されました。「st_geometry_interpolation オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

ST_WithinDistanceFilter の強化　曲面の空間参照系のジオメトリで空間述部 ST_WithinDistanceFilter がサポートされるようになりました。ST_Geometry タイプの ST_WithinDistanceFilter メソッド『SQL Anywhere サーバー 空間データサポート』を参照してください。

● **java_class_path オプションの強化**　java_class_path database オプションを使用して、システムクラスローダーが Java VM を起動する前にクラスパスに追加するクラスと JAR ファイルを指定できるようになりました。このオプションは、アプリケーションでディレクトリまた

は JAR ファイルを提供する場合に便利です。「java_class_path option」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい MATVIEW ODBC 接続パラメーター**　MATVIEW 接続パラメーターを使用すれば、ODBC 関数 SQLTables を実行するときに実体化ビューのテーブルタイプに返す文字列を指定できます。デフォルトでは、SQLTables 関数によって返される値は VIEW です。「MatView (MATVIEW) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい -kp データベースサーバーオプション**　-kp オプションを使用して、標準形式の *servername/hostname@REALM* でサーバープリンシパルを指定できるようになりました。-kp オプションは、データベースサーバーに対する Kerberos 認証された接続を有効にします。「-kp dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **PartnerState プロパティの強化**　PartnerState プロパティが DB_PROPERTY 関数と共に使用した場合に次のいずれかの値を返すようになりました。

  - **connected**　現在のサーバーから指定されたサーバーへの接続および指定されたサーバーから現在のサーバーへの接続があります。
  - **incoming only**　指定されたサーバーからこのサーバーへの接続があります。
  - **outgoing only**　このサーバーから指定されたサーバーへの接続があります。
  - **disconnected**　このサーバーと指定されたサーバーの間に接続がありません。
  - **NULL**　データベースはミラーリングされていません。

  PartnerState データベースプロパティ『SQL Anywhere サーバー データベース管理』を参照してください。

- **MirrorServerState プロパティの強化**　MirrorServerState プロパティが、DB_EXTENDED_PROPERTY 関数と共に使用した場合に次のいずれかの値を返すようになりました。

  - **connected**　現在のサーバーから指定されたサーバーへの接続および指定されたサーバーから現在のサーバーへの接続があります。
  - **incoming only**　指定されたサーバーからこのサーバーへの接続があります。
  - **outgoing only**　このサーバーから指定されたサーバーへの接続があります。
  - **disconnected**　このサーバーと指定されたサーバーとの間に接続はありません。
  - **NULL**　データベースはミラーリングされていません。

  MirrorServerState データベースプロパティ『SQL Anywhere サーバー データベース管理』、「DB_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』、「DB_EXTENDED_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい sp_forward_to_remote_server プロシージャー**　sp_forward_to_remote_server ストアドプロシージャーを使用して、アプリケーションがリモートサーバー上で SQL 文を実行し、この文によって生成される結果セットを取得できます。「sp_forward_to_remote_server システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  注意
  このストアドプロシージャーにアクセスするには、データベースをアップグレードする必要があります。

- **新しい sa_user_defined_counter_add システムプロシージャー**　sa_user_defined_counter_add システムプロシージャーを使用して、ユーザー定義のプロパティの値を変更できます。sa_user_defined_counter_add システムプロシージャーを参照してください。「sa_user_defined_counter_add システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  注意
  このストアドプロシージャーにアクセスするには、データベースをアップグレードする必要があります。

- **新しい sa_user_defined_counter_set システムプロシージャー**　sa_user_defined_counter_set システムプロシージャーを使用して、ユーザー定義のプロパティの値を設定できます。「sa_user_defined_counter_set システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  注意
  このストアドプロシージャーにアクセスするには、データベースをアップグレードする必要があります。

- **SQLANYSAMP12 環境変数**　Unix および Mac OS X インストーラーで *sa_config* および *sample_config* スクリプトに SQLANYSAMP12 環境変数が設定されるようになりました。Unix では、*sample_config* スクリプトを使用して、ユーザー別のサンプルのコピーを作成できます。これはマルチユーザーインストールに便利です。シングルユーザーインストールの場合、*sa_config* スクリプトで *$SQLANY12/samples* に SQLANYSAMP12 を設定します。「SQLANYSAMP12 環境変数」『SQL Anywhere サーバー データベース管理』と「UNIX および Mac OS X ファイル」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データ型 LONG VARCHAR、LONG BINARY、および LONG NVARCHAR の IN パラメーターをリモートプロシージャーコールで使用可能**　リモートプロシージャーコールにデータ型 LONG VARCHAR、LONG BINARY、および LONG NVARCHAR の IN パラメーターを含むことができるようになりました。さらに、データ型 VARCHAR、NVARCHAR、および BINARY のパラメーターが 255 バイトに制限されなくなりました。「リモートプロシージャーの作成 (Sybase Central の場合)」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **CREATE SERVER 文の USING 句に変数を含めることが可能**　CREATE SERVER 文の USING 句に変数を含めることができるようになりました。この機能を使用すれば、ユーザーは動的なリモートデータアクセスサーバーを作成できます。「CREATE SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **リモートデータアクセスで SQL Anywhere の ODBC ドライバーを直接ロード可能**　Windows と UNIX の両プラットフォームで ODBC ドライバーマネージャーをバイパスして、リモートデータアクセスで SQL Anywhere ドライバーが直接ロードされるようにリモートサーバーを定義できます。リモートサーバーを指定する際、次の構文を使用します。この後には、接続文字列の残りの部分を続けます。

  ```
  CREATE SERVER remote-server CLASS 'saodbc' USING 'DRIVER=SQL Anywhere Native;...';
  ```

  DRIVER=SQL Anywhere Native' を使用しないで定義されている SQL Anywhere リモートサーバーが複数ある場合、リモートデータアクセスは他のリモートサーバーのドライバーマネージャーを引き続き使用します。

- **CREATE EXISTING TABLE 文および CREATE PROCEDURE 文の AT 句に変数を含めることが可能**　CREATE EXISTING TABLE および CREATE PROCEDURE 文の AT 句に変数を含めることができるようになりました。この機能を使用すれば、ユーザーはプロキシテーブルまたはプロキシプロシージャを複数のリモートテーブルまたはプロシージャーにマッピングできます。「CREATE EXISTING TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「CREATE PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **MERGE 文の RAISERROR 句および RAISERROR 文への強化**　RAISERROR 文を使用することによって、SQL Anywhere データベースサーバーはアプリケーションでカスタマイズしたエラーを発生させることができます。データベースサーバーはまた組み込みグローバル変数 SQLCODE を提供します。この変数の値を調べて、現在の接続の最後の文の実行中に発生した特定のエラーを識別できます。データベースサーバーは固定 -631 エラーメッセージの代わりに SQLCODE のユーザー指定のエラー番号をレポートするようになりました。「MERGE 文」『SQL Anywhere サーバー SQL リファレンス』と「RAISERROR 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **VALIDATE TEXT INDEX 文**　VALIDATE TEXT INDEX 文を使用して、テキストインデックス内の単語の位置情報が正常であることを確認できます。位置情報が壊れている場合、エラーが生成されます。「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **TOP および LIMIT 句の拡張構文**　TOP { ALL | limit } START AT startat および LIMIT limit [OFFSET offset] 句が *limit*, *offset*、および *startat* 引数の簡単な計算式をサポートするようになりました。TOP は ALL limit をサポートし、指定された *startat* 値の後にすべてのローが返されることを示します。(*limit* + *offset*) と (*limit* + *startat* -1) の最大値が 9223372036854775807 = 2^64-1 に増えました。

  次の項を参照してください。

  - 「SELECT、UPDATE、DELETE クエリブロック内のロー制限句」『SQL Anywhere サーバー SQL の使用法』
  - 「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「UPDATE 文」『SQL Anywhere サーバー SQL リファレンス』

- **SQL Anywhere OLE DB プロバイダーが DBTYPE_DBTIMESTAMPOFFSET データ型をサポート**　SQL Anywhere OLE DB プロバイダーが DBTYPE_DBTIMESTAMPOFFSET データ型を

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

サポートするようになりました。DBTYPE_DBTIMESTAMPOFFSET (146) は TIMESTAMP WITH TIME ZONE (または DATETIMEOFFSET) データ型をサポートする OLE DB 型です。このデータ型をサポートすることにより、SQL Anywhere データベースと他のデータベース管理システム (SQL Anywhere を含む) 間でのデータテーブルの転送が容易になります。

- **Web サービスで HTTP リダイレクト操作の改善された制御をサポート** CREATE PROCEDURE および CREATE FUNCTION 文の新しい SET REDIR 句は許容される最大数のリダイレクションの制御を可能にし、自動的にリダイレクトする HTTP ステータスを指定します。

  303 ステータスを受信する POST HTTP メソッドを指定する Web サービスプロシージャが GET HTTP メソッドを使用してリダイレクト要求を発行します。

  HTTP クライアントプロシージャーで相対パスのリダイレクションを処理するようになりました。以前は、サーバーで絶対 URL が提供された場合にのみリダイレクションを実行できました。

  リダイレクトを受け取る GET メソッドのみが、リダイレクト応答のロケーションヘッダー URL によって指定されたクエリコンポーネントを提供します。リダイレクトを受け取る POST メソッドがリダイレクト応答のロケーションヘッダーによって指定されたパスとクエリコンポーネントを含む要求の URL を発行します。この要求の本文には、プロシージャーによって生成されたクエリコンポーネントが含まれます。

  URL 句内および (自動的に生成されて) プロシージャーに渡されたパラメーターからの両方でクエリパラメーターを指定できるようになりました。これは、GET HTTP メソッドを指定するプロシージャーにのみ該当します。

- **Windows Mobile でセキュア Web サービスをサポート** Windows Mobile で HTTPS and HTTPS_FIPS を使用する Web サービスプロシージャーがサポートされるようになりました。

- **SQL Anywhere JDBC ドライバーが PreparedStatement.setClob() をサポート** SQL Anywhere JDBC ドライバーが PreparedStatement.setClob() をサポートするようになりました。

  以前のリリースでは、SQL Anywhere JDBC ドライバーは PreparedStatement.setBlob、ResultSet.getBlob、および ResultSet.getClob のサポートを提供しました。SQL Anywhere JDBC ドライバーは 3 つの PreparedStatement.setClob メソッドのうちの 2 つをサポートするようになりました。これらのメソッドについては、以下に説明します。

  ```
  PreparedStatement.setClob( int parameterIndex, Clob x )
  PreparedStatement.setClob( int parameterIndex, Reader reader, long length )
  ```

  以下のバリエーションは、SQL Anywhere JDBC ドライバーではサポートされていません。

  ```
  PreparedStatement.setClob( int parameterIndex, Reader reader )
  ```

  PreparedStatement.setClob(int parameterIndex, Clob x) オーバーロードを使用する場合、ユーザー提供の Clob 実装は Clob.length および Clob.getCharacterStream メソッドのみサポートする必要があります。また、大きい文字列および大きい文字ストリームの場合、SQL Anywhere JDBC ドライバー内の新しい PreparedStatement.setClob メソッドが PreparedStatement.setString および PreparedStatement.setCharacterStream メソッドよりもパ

フォーマンスとメモリ使用率の両方で推奨されます。「JDBC サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL Anywhere Java VM ClassLoader がシャットダウンフックをサポート**　データベースサポートでの Java の提供に使用される SQL Anywhere Java VM ClassLoader を使用して、アプリケーションでシャットダウンフックをインストールできます。「Java VM でのシャットダウンフック」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL Anywhere .NET SetupVSPackage インストーラー**　SetupVSPackage アプリケーションで Global Assembly Cache や Windows Microsoft.NET *machine.config* ファイルの更新などのいくつかのインストーラー機能を実行できるようになりました。SQL Server 2008 以降がシステムにインストールされている場合、SetupVSPackage で *MSSqlToSA.xml* および *SAToMSSql10.xml* と呼ばれる 2 つのマッピングファイルを SQL Server の *DTS¥MappingFiles* フォルダーにインストールすることもできます。「.NET クライアントの配備」『SQL Anywhere サーバー プログラミング』を参照してください。

SQL Anywhere の動作の変更

次に、バージョン 12.0.1 で導入された SQL Anywhere の動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **LOAD および UNLOAD TABLE 文で使用されるエスケープ文字**　以前のリリースでは、これらの文のエスケープ文字は 1 バイトより長くすることはできませんでした。現在は、エスケープ文字に指定する文字列は 255 バイトを超えないことが推奨されていますが、文字列は 1 文字以下にすることが推奨されています。

- **最大キャッシュサイズに応じて最小および初期キャッシュサイズが増える**　初期キャッシュサイズまたは最小キャッシュサイズを最大キャッシュサイズの 8 分の 1 より小さい値に設定すると、最大キャッシュサイズに関連して、初期および最小キャッシュサイズが自動的に増えます。その結果、最小キャッシュサイズおよび初期キャッシュサイズが以前のバージョンよりも大きくなる場合がよくあります。

- **reserved_keywords オプションの範囲の変更**　以前は、reserved_keywords オプションを個別のユーザーに指定したり、設定が一時的な範囲であることを指定できました。しかし、特定の DDL 文を実行するときに一時的な設定やユーザーレベルの設定が対応する PUBLIC 設定と異なる場合、データベースの受信または再構築時に問題が発生する場合があります。

  reserved_keywords オプションへの次の動作変更は、新しい 12.0.1 バージョンのデータベースに適用されます。

  ○一時的な設定および非 PUBLIC 設定は許可され**ません**。

  reserved_keywords オプションへの次の動作変更は、既存のバージョン 12 のデータベースに適用されます。

  ○既存の非 PUBLIC 設定は実行中に無視されます。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○既存の非 PUBLIC 設定の削除が許可されます。

○Dbunload は、既存の非 PUBLIC 設定を無視します。

- **マルチプログラミングレベルのオプション**　以前は、ネットワークデータベースサーバー (dbsrv12) に厳密に適用されるマルチプログラミングレベルのデータベースサーバーオプションはパーソナルデータベースサーバー (dbeng12) によって、エラーを生成することなく無視されました。現在は、以下のデータベースサーバーオプションは、パーソナルデータベースサーバーで使用される場合、警告を生成します。

  ○「-gna」　dbsrv12 サーバーオプション
  ○「-gnh」　dbsrv12 サーバーオプション
  ○「-gnl」　dbsrv12 サーバーオプション
  ○「-gns」　dbsrv12 サーバーオプション

- **-zt データベースサーバーオプションに対する変更**　-zt データベースサーバーオプションを設定することによって、ReqCountBlockIO プロパティと ReqTimeBlockIO プロパティが利用可能になることはなくなりました。これらの 2 つのプロパティは、RequestTiming プロパティがオンにされているかどうかにかかわらず現在では常に利用できます。

- **SQLTables 関数の ODBC 接続パラメーターへの変更**　ODBC 関数 SQLTables を実行すると、TABLE_TYPE カラムはデフォルトで実体化ビューに対して値 VIEW を返すようになりました。以前は、この関数は値 MATERIALIZED VIEW を返しました。MATVIEW 接続パラメーターを使用してこのパラメーターを変更できます。「MatView (MATVIEW) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **CONVERT 関数**　フォーマット化された文字列の解析が強化されて、文字列の時間部分がフォーマット hh:nn:ss.ssssssAA に一致すると受け入れられるようになりました。時間文字列は時間の桁を指定する必要がありますが、その他の時間部分はすべてオプションです。AM/PM インジケーターは時間部分が省略されるかどうかに関わらず、常に受け入れられます。これにより、秒の後のマイクロ秒を表す 6 桁までが許容されます。

  この変更は文字列の TIME および TIMESTAMP への変換にも影響を与えます。「CONVERT 関数 [データ型変換]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  以前は、*format-style* を指定して CONVERT 関数を使用して文字列を時間に変換すると、SQL Anywhere 10 以降では以前のバージョンで許可された変換が拒否されました。たとえば、次の文はバージョン 9 では受け入れられましたが、バージョン 10 以降では拒否されました。

  ```
  SELECT CONVERT( TIME, '11:45am', 14 ) tm_conv
  ```

  文字列を TIME に変更する動作が SQL Anywhere のバージョン 9 からバージョン 10 以降で変更され、バージョン 10 以降では、文字列を TIMESTAMP に変換するために使用される規則と同じ規則が適用されます。たとえば、文字列 11:45am はフォーマットスタイル 14 (hh:nn:ss.sss) とは正確には一致しません。なぜなら、この文字列にはスタイルに存在しない am インジケーターが含まれるからです。

  次の文はバージョン 9 では受け入れられましたが、バージョン 10 以降では拒否されました。文字列がスタイルフォーマット 101 (mm/dd/yyyy) と一致しないからです。

  ```
  SELECT CONVERT( TIME, '1991-02-03 11:45', 101 )
  ```

- **サーバーライセンスユーティリティ (dblic) によるライセンスされたユーザー名と会社名の変更**　ユーザーカウントとライセンスタイプを指定しなくてもライセンスされたユーザーと会社を変更できるようになりました。「サーバーライセンス取得ユーティリティ (dblic)」『SQL Anywhere サーバー データベース管理』を参照してください。

SQL Anywhere の廃止予定機能とサポート終了機能

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

バージョン 12.0.1 で廃止された機能のリストを次に示します。

- **-kr オプション**　以前の Kerberos サーバープリンシパルは、*server-name@default-realm* の形式にする必要があります。

  領域を含めた Kerberos サーバープリンシパルは -kp サーバーオプションで指定できます。-kp で指定されるサーバープリンシパルは、データベースサーバーを実行するコンピューターで Kerberos keytab ファイルに抽出されている必要があります。-kp オプションが指定されている場合、-kr オプションは指定できません。「-kp dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL Anywhere JDBC 3.0 ドライバー**　SQL Anywhere JDBC 3.0 ドライバーはこのソフトウェアリリースで廃止されています。*sajdbc.jar* を使用するアプリケーションを *sajdbc4.jar* に切り替えることをお勧めします。「JDBC サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

Mobile Link の新機能

次に、バージョン 12.0.1 で導入された Mobile Link の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

統合データベース

次に、SQL Anywhere バージョン 12.0.1 での Mobile Link 用の統合データベースサポートの強化を示します。

- **Sybase IQ をサポート**　Mobile Link サーバーは Sybase IQ を統合データベースとしてサポートするようになりました。「Sybase IQ 統合データベース」『Mobile Link サーバー管理』を参照してください。 推奨ドライバーの詳細については、http://www.sybase.com/detail?id=1011880 を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **ASE 15.5 のサポートを追加**　Windows および Linux 上の Mobile Link サーバーで ASE 15.5 サーバー上で動作する統合データベースをサポートするようになりました。推奨ドライバーの詳細については、http://www.sybase.com/detail?id=1011880 を参照してください。

Mobile Link サーバー

次に、SQL Anywhere バージョン 12.0.1 での Mobile Link サーバーの強化を示します。

- **データベースワーカースレッドの自動調整**　Mobile Link サーバーはスループットを最大化するために、データベースワーカースレッドの数を自動的に調整できるようになりました。「-wm mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **新しい名前付きシステムパラメーター**　次の名前付きシステムパラメーターが Mobile Link に追加されて、スクリプトで新しいリモート ID またはユーザー名を識別できるようになりました。

  - **new_remote_id**　この新しいシステムパラメーターは新しいリモート ID を示しており、authenticate_user and begin_synchronization 接続スクリプト、および begin_synchronization テーブルスクリプトで使用できます。

  - **new_username**　この新しいシステムパラメーターは新しいユーザー名を示しており、authenticate_user and begin_synchronization 接続スクリプト、および begin_synchronization テーブルスクリプトで使用できます。

  詳細については、次の項を参照してください。

  - 「authenticate_user 接続イベント」『Mobile Link サーバー管理』
  - 「begin_synchronization 接続イベント」『Mobile Link サーバー管理』
  - 「begin_synchronization テーブルイベント」『Mobile Link サーバー管理』

- **ASE 15.5 をサポートするために bigtime および bigdatetime データ型を追加**　これらのデータ型は、SQL Anywhere および Ultra Light の TIME および TIMESTAMP にそれぞれマッピングされます。「Adaptive Server Enterprise データのマッピング」『Mobile Link サーバー管理』を参照してください。

- **認証パラメーターの最大長が拡張された**　各 Mobile Link 認証パラメーターは 4000 バイトまでの長さで使用できます。以前の制限は 128 バイトでした。「認証パラメーター」『Mobile Link サーバー管理』を参照してください。

mslrv12 の新機能

- **データベースワーカースレッドの自動調整を可能にする新しい -wm オプション**　-wm mslrv12 オプションが追加されました。-wm オプションを使用してデータベースワーカースレッドの最大数を設定し、Mobile Link サーバーで最適なデータベースワーカースレッド数を自動的に選択できるようにします。「-wm mslrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-cmax、-cmin および -cinit オプションがサイズパラメーターのパーセンテージの使用をサポート**　-cmax、-cmin、および -cinit サーバーオプションのサイズパラメーターをパーセンテー

ジとして指定できるようになりました。「-cmax mlsrv12 オプション」『Mobile Link サーバー管理』、「-cmin mlsrv12 オプション」『Mobile Link サーバー管理』、「-cinit mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-x mlsrv12 オプションの新しい log_bad_request パラメーター** -x mlsrv12 オプションで HTTP、HTTPS、および OE の log_bad_request オプションをサポートするようになりました。yes に設定すると、Mobile Link サーバーは不完全または予期しない HTTP 要求を受信した場合にエラーを出力します。「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **エンドツーエンド暗号化で DER でコード化されたキーをサポート** -x mlsrv12 オプションの e2ee_private_key オプションで PEM と DER の両方でコード化されたキーをサポートするようになりました。

新しい Mobile Link によるリモートデータベース集中管理機能

SQL Anywhere バージョン 12.0.1 で導入された Mobile Link によるリモートデータベースの集中管理機能の強化の一覧を以下に示します。

- **タスク条件の新しい変数** リモートデータベースの集中管理用 Mobile Link エージェントでタスク条件の 2 つの新しい変数をサポートするようになりました。{is_online} と {network_conn_name} の 2 つです。変数 {is_online} は、クライアントデバイスがエージェントの Mobile Link サーバーにアクセスできるネットワークに接続されているかどうかを確認するために使用できます。変数 {network_conn_name} は、Mobile Link サーバーと通信を行うエージェントによって使用されるネットワーク接続の名前を評価します。「パラメーターの変数」『Mobile Link サーバー管理』を参照してください。

- **タスク実行のランダムな遅延の新規サポート** リモートタスクに秒単位で設定されるランダムな遅延間隔を設定できるようになりました。各エージェントはこの間隔を使用して、各タスクの実行を遅らせるランダムな秒数を生成します。「リモートタスク」『Mobile Link サーバー管理』を参照してください。

- **リモートデータベースの管理用 Mobile Link エージェントを対話形式で設定できるようになりました。** Mobile Link エージェントをリモートデバイスで一連の設定ウィンドウを使用して、対話形式で設定できるようになりました。

サーバーユーティリティ

次に、SQL Anywhere バージョン 12.0.1 での Mobile Link サーバーユーティリティの強化を示します。

Mobile Link リプレイユーティリティ (mlreplay)

- **mlreplay ユーティリティは複数のシミュレートされたクライアントを実行し、コマンドラインのみを使用して記録されたプロトコルをリプレイできます。** -n mlreplay オプションを使用して、多数のシミュレートされたクライアントを管理します。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい -rep mlreplay オプションの追加**　-rep mlreplay オプションで、シミュレートしたクライアントを使用して、記録されたプロトコルをリプレイする回数を指定します。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

- **新しい -rnt mlreplay オプションの追加**　-rnt mlreplay オプションは、シミュレートしたクライアントに指定したランタイムに達するまでプロトコルリプレイの新たな繰り返しを開始するように指示します。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

- **設定ファイルをサポートする新しい @data mlreplay および mlgenreplayapi オプションの追加**　*@data* mlreplay オプションを使用すれば、mlreplay と mlgenreplayapi を設定ファイルに格納されたコマンドラインの指定で実行できます。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』、「生成された Mobile Link リプレイ API ユーティリティ (mlgenreplayapi)」『Mobile Link サーバー管理』、「設定ファイル」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい -ls mlreplay オプションの追加**　-ls mlreplay オプションは、シミュレートしたクライアントごとの統計情報をログに記録します。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

- **新しい -os mlreplay および mlgenreplayapi オプションの追加**　-os オプションはログファイルの最大サイズを制限します。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』と「生成された Mobile Link リプレイ API ユーティリティ (mlgenreplayapi)」『Mobile Link サーバー管理』を参照してください。

- **コマンドラインで複数のシミュレートしたクライアントを実行可能**　-u、-p、および -r パラメーターでアスタリスク文字を使用して、コマンドラインで複数のユーザー名、パスワード、およびリモート ID をそれぞれ指定できるようになりました。アスタリスク文字はシミュレートしたクライアント呼び出しごとにシミュレートしたクライアント数に置き換えられます。

  新しい -rp mlreplay オプションを使用すれば、アスタリスクのワイルドカード記号を別の記号に変更できます。次に新しい記号を使用して、複数のユーザー名、パスワード、およびリモート ID を指定できます。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

Sybase Central の Mobile Link プラグイン

次に、SQL Anywhere バージョン 12.0.1 での Sybase Central 用 Mobile Link プラグインの強化を示します。

- **同期モデルの展開で同期プロファイルを作成できるようになりました**　同期モデルをリモートデータベースに展開するときに、自動的に同期プロファイルが作成されるようになりました。

- **同期モデルが Sybase IQ をサポート**　Sybase IQ 統合データベースで同期モデルを作成できるようになりました。「Sybase IQ 統合データベース」『Mobile Link サーバー管理』を参照してください。

- **トリガーの代わりにカラムデフォルトを使用するタイムスタンプダウンロード**　SQL Anywhere、Sybase IQ、IBM DB2 LUW、および MySQL 統合データベースを使用する同期モ

デルでトリガーを使用して TIMESTAMP カラムを維持することなく、タイムスタンプベースのダウンロードを使用できるようになりました。ローが挿入または更新されるときに、トリガーの代わりにカラムデフォルト値を使用して、ロー内の TIMESTAMP カラムを自動的に更新できます。これを簡単にするために、テーブルマッピングエディターの**同期モデル作成ウィザード**および **[ダウンロードタイプ]** タブの **[タイムスタンプダウンロードのオプション]** ページに新しい **[トリガーの代わりにカラムデフォルトを使用する]** オプションが追加されました。

- **統合データベースからリモートスキーマ名をインポート可能**　**プロジェクト作成ウィザード**または**統合データベース追加ウィザード**のいずれを使用する場合も、プロジェクトに統合データベースを追加するときは、ウィザードによって、プロジェクトに存在しないリモートスキーマ名が統合データベースに定義されているかどうかが自動的にチェックされます。定義されている場合は、それをインポートするかどうか尋ねられます。

- **ファイルからグループを作成可能**　**グループ作成ウィザード**を使用すれば、エージェントの名前を含むテキストファイルを指定して、グループのエージェントを選択できます。統合データベースで定義されたエージェントを参照して、メンバーを追加することもできます。

- **リモートタスクコマンドの新しいヘルプボタン**　**[コマンドタイプ]** ドロップダウンリストの右にリモートタスクの作成に使用するヘルプを表示できる疑問符マークのアイコンが付きました。

- **エージェントの新しい [説明] フィールド**　**[エージェントのプロパティ]** ウィンドウおよび **Mobile Link エージェント作成ウィザード**にエージェントの説明を入力できる **[説明]** フィールドができました。

- **新しい CustDB サンプルプロジェクトファイルを追加**　CustDB サンプルプロジェクトファイル、*%SQLANY12%¥MobiLink¥CustDB¥project.mlp* を使用できるようになり、CustDB プロジェクトを簡単に操作して、データベーススクリプトを表示できるようになりました。詳細については、「Mobile Link の CustDB サンプル」『Mobile Link クイックスタート』と「チュートリアル: Ultra Light CustDB サンプルアプリケーションの構築」『Ultra Light データベース管理とリファレンス』を参照してください。

Mobile Link クライアント

次に、SQL Anywhere バージョン 12.0.1 での Mobile Link クライアントの強化を示します。

- **HTTP セッションの自動再開**　HTTP 接続が失われると、Mobile Link クライアントが自動的に Mobile Link サーバーに接続して同期を続けようとします。数回試行した後にクライアントが Mobile Link サーバーに再接続できないと、同期は失敗します。

- **エンドツーエンド暗号化で DER でコード化されたキーをサポート**　e2ee_public_key クライアントプロトコルオプションで PEM でコード化されたキーと DER でコード化されたキーをサポートするようになりました。

- **ECC 曲線サポートを 15 曲線をサポートするように拡張**　サポートされる ECC 曲線は、sect163k1、sect163r2、sect233k1、sect233r1、sect283k1、sect283r1、sect409k1、sect409r1、sect571k1、sect571r1、secp192r1、secp224r1、secp256r1、secp384r1、および secp521r1 です。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **x.509 証明書からのパブリックキーのサポート**　エンドツーエンド暗号化で x.509 証明書からのパブリックキーをサポートするようになりました。「e2ee_public_key」『Mobile Link クライアント管理』を参照してください。

Relay Server

このリリースでは、次の Relay Server 機能が追加されています。

- **Relay Server Outbound Enabler (RSOE) が Web サーバーおよび HTTP プロキシに対して HTTP 認証 (基本認証とダイジェスト認証) をサポートするようになりました。**　Outbound Enabler に次の新しい (オプションの) ネットワーク接続オプションが追加されました。
  - **http_userid**　認証のユーザー ID。
  - **http_password**　認証のパスワード。
  - **http_proxy_userid**　プロキシ認証のユーザー ID。
  - **http_proxy_password**　プロキシ認証のパスワード。
  - **proxy_host**　プロキシサーバーのホスト名または IP アドレスを指定します。
  - **proxy_port**　プロキシサーバーのポート番号を指定します。

  詳細については、次の項を参照してください。

  - 「Outbound Enabler」『Relay Server』
  - 「Mobile Link クライアントネットワークプロトコルオプション」『Mobile Link クライアント管理』

Mobile Link の動作の変更

次に、バージョン 12.0.1 で導入された Mobile Link の動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

Mobile Link サーバーの変更

- **64 ビットのオペレーティングシステムでの 32 ビットの Mobile Link サーバーのサポート廃止**　64 ビットのオペレーティングシステムでは、Mobile Link サーバーは 64 ビットのアプリケーションとしてのみインストールされ、64 ビットのアプリケーションとして動作します。旧バージョンの 12 のインストールからシステムに残っている 32 ビットの Mobile Link サーバーは 64 ビットシステムでは動作できません。

注意
12.0.1 より前のバージョンの Mobile Link サーバーの場合、Mobile Link サーバーが 64 ビットのオペレーティングシステム上で 32 ビットのアプリケーションとして実行された場合、エラーは発生**しません**。

12.0.1 以降のバージョンでは、Mobile Link サーバーが 64 ビットのオペレーティングシステム上で 32 ビットのアプリケーションとして実行された場合、エラー -10381 が返されるようになりました。「Mobile Link サーバーは、64 ビットオペレーティングシステムで、64 ビットアプリケーションとして実行する必要があります」『エラーメッセージ』を参照してください。

- **-w mlsrv12 オプションの新しい動作** -wm オプションを使用してデータベースワーカースレッドの数を自動的に調整した場合に、mlsrv12 の -w オプションでデータベースワーカースレッドの初期数が設定されるようになりました。以前は、このオプションでデータベースワーカースレッドの数を設定しました。「-w mlsrv12 オプション」『Mobile Link サーバー管理』と「-wm mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-cm オプションが -cmax オプションのエイリアスになる** -cm オプションが -cmax オプションのエイリアスになりました。「-cmax mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

Mobile Link サーバーユーティリティの変更

次に、バージョン 12.0.1 で導入された Mobile Link サーバーユーティリティの動作の変更を示します。

Mobile Link リプレイユーティリティ (mlreplay)

- **-o、-ot mlreplay および mlgenreplayapi オプションの新しい動作** mlreplay および mlgenreplayapi の -o および -ot オプションでコマンドラインオプションをログに記録するようになりました。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』と「生成された Mobile Link リプレイ API ユーティリティ (mlgenreplayapi)」『Mobile Link サーバー管理』を参照してください。

- **一定の条件下で -n オプションと -sci mlreplay オプションを同時に使用可能** -n オプションと -sci mlreplay オプションは、-n オプションで指定されシミュレートされたクライアントの数がシミュレートされたクライアント情報ファイルのシミュレートされたクライアント数と同じか少ないときには、一緒に使用できます。同時に使用するときは、-n では実行するシミュレートされたクライアントの数を指定します。

  これらのオプションを使用することにより、1 つのシミュレートされたクライアント情報ファイルにシミュレートされたクライアント数 *x* を指定することによって、1 ～*x* の範囲の数のシミュレートされたクライアントのプロトコルをリプレイできます。以前の動作は、正確に *x* 数のシミュレートされたクライアントで記録されたプロトコルをリプレイするためにだけ使用できます。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

- **IdentifySimulatedClient および FiniIdentifySimulatedClient コールバックでコードテンプレートを使用可能** リプレイセッションに関する一般ユーザー名、パスワード、およびリモート

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ID を使用するときに実装できるように、これらのコールバックには生成時にコメントアウトされたコードが記述されています。「IdentifySimulatedClient コールバック」『Mobile Link サーバー管理』と「FiniIdentifySimulatedClient コールバック」『Mobile Link サーバー管理』を参照してください。

Mobile Link クライアントの変更

次に、バージョン 12.0.1 で導入された Mobile Link クライアントの動作の変更を示します。

- **[SQL Anywhere の DBMLSync 設定] ウィンドウで次の変更が行われました**

  ○**[リモートプログレスに対してリトライする]**、**[リモートのオフセットが後]**、**[リモートのオフセットが前]**、**[競合する接続の削除]**、**[サイトスクリプト]**、**[コマンドラインヘルプ]** オプションは削除されました。

  ○**[パブリケーション]** オプションは **[サブスクリプション]** オプションに置き換えられました。

  ○Mobile Link パスワードを変更しようとすると、ソフトウェアはウィンドウを閉じる前に、空以外のパスワードを入力したこと、新しいパスワードと確認用のパスワードが一致することを確認します。

  ○不正な Mobile Link パスワードが入力されたために、同期の試行後にウィンドウが開くと、問題を示すメッセージが表示されます。以前は、エラーメッセージは dbmlsync ウィンドウに表示されました。

- **永続的な HTTP および HTTPS の同期の新しいデフォルト**　デフォルトで、HTTP または HTTPS を使用して同期する Mobile Link クライアント (Ultra Light J を除く) が永続的な同期を使用するようになりました。以前の動作を返すには、クライアントはクライアントネットワークプロトコルオプションで persistent=no を指定する必要があります。

Sybase Central の Mobile Link プラグインの変更

次に、バージョン 12.0.1 で導入された Sybase Central の Mobile Link プラグインの動作の変更を示します。

- **同期モデル展開ウィザードの [Mobile Link ユーザーおよびサブスクリプション] ページが変わりました**　**同期モデル展開ウィザード**の **[Mobile Link ユーザーおよびサブスクリプション]** ページの名前が **[Mobile Link ユーザーおよび同期プロファイル]** ページに変わりました。このページに同期プロファイル名を指定するオプションができました。同期モデルが展開されると、同期プロファイルが作成されるようになりました。

- **リモートタスクに適したデフォルトを選択する同期モデル展開オプション**　**同期モデル展開ウィザード**の最初のページにリモートデータベースの集中管理用のリモートタスクを作成するために適切な設定でウィザードを初期化するようにリモートデータベースに展開する場合のオプションが追加されました。このオプションはデフォルトで選択されています。

Mobile Link の廃止予定機能とサポート終了機能

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

- **IBM DB2 メインフレームのサポート終了**　IBM DB2 メインフレームは統合データベースとしてサポートされなくなりました。ただし、DB2 LUW (Linux、UNIX、Windows) は従来どおり、Mobile Link で統合データベースとしてサポートされます。

QAnywhere の新機能

次に、バージョン 12.0.1 で導入された QAnywhere の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

QAnywhere 12.0.1 に新機能はありません

QAnywhere の動作の変更と廃止予定機能

QAnywhere 12.0.1 では動作の変更や廃止された機能はありません。サポートされているプラットフォームおよびバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

SQL Remote の新機能

次に、バージョン 12.0.1 で導入された SQL Remote の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **SET REMOTE OPTION 文の新しい制御パラメーター**　SQL Remote が継続モードで実行されていて、メッセージシステムへのアクセス時にエラーが発生すると、次の項目を制御できるようになりました。

  ○シャットダウンまでに SQL Remote で送受信を再試行する回数。max_retries 共通オプションについては、「SET REMOTE OPTION 文 [SQL Remote]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○再試行間で待機する時間。pause_after_failure 共通オプションについては、「SET REMOTE OPTION 文 [SQL Remote]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

SQL Remote の動作の変更

次に、バージョン 12.0.1 で導入された SQL Remote の変更点を示します。

- **見つからないメッセージを処理するパフォーマンスの向上** SQL Remote バージョン 12.0.1 以前では、マルチパートのメッセージのすべてのメッセージが受信される前に統合データベースが終了すると、リモートデータベースはすべての受信メッセージを削除して、統合データベースに再送信要求を送信します。統合データベースは再送信要求を満たすために該当するトランザクションログをスキャンします。SQL Remote バージョン 12.0.1 では、起動時に SQL Remote が見つからないメッセージに関する警告メッセージを発行し、既存のメッセージを削除して、コミットを含み、マルチパートのメッセージの最後のパートであるメッセージまたは前のマルチパートのメッセージ後の次のメッセージとなるメッセージのみを再送信するように求めます。

Ultra Light の新機能

次に、バージョン 12.0.1 で導入された Ultra Light の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

一般的な機能

- **Android のスマートフォンで Ultra Light がサポートされるようになりました** Ultra Light が Android のスマートフォンをサポートするようになりました。Ultra Light J API の Android 実装を使用して、Ultra Light データベースを使用するアプリケーションを作成できます。この実装の使用方法は、BlackBerry スマートフォンに固有の Ultra Light データベースをサポートする BlackBerry の実装とは異なります。詳細については、「Windows Mobile 用 Ultra Light API の選択」『Ultra Light データベース管理とリファレンス』と「Ultra Light J アプリケーションの開発」『Ultra Light - Java プログラミング』を参照してください。

  新しいチュートリアルでは、Android のスマートフォンまたは Eclipse 環境のシミュレーターで Ultra Light アプリケーションを作成する方法を示します。このチュートリアルは、*%SQLANY12%¥UltraLiteJ¥Android¥CustDB* ディレクトリにある新しいコードサンプルに基づいています。「チュートリアル：Android アプリケーションの構築」『Ultra Light - Java プログラミング』を参照してください。

- **BlackBerry アプリケーション作成用のチュートリアルで Eclipse 環境がサポートされるようになりました** BlackBerry チュートリアルが Eclipse 環境を使用するように更新されました。「チュートリアル：BlackBerry アプリケーションの構築」『Ultra Light - Java プログラミング』を参照してください。

- **動的キャッシュサイズ決定** Ultra Light で、データベース操作への応答で保証され、利用可能なメモリで許容される場合、データベースファイルキャッシュが増加するようになりました。

アプリケーションで明示的にキャッシュサイズを決定できるようにもなりました (通常、アプリケーションがメモリ使用率を減らすように要求される場合)。次の項を参照してください。

- 「Ultra Light CACHE_SIZE 接続パラメーター」『Ultra Light データベース管理とリファレンス』
- 「Ultra Light CACHE_MIN_SIZE 接続パラメーター」『Ultra Light データベース管理とリファレンス』
- 「Ultra Light CACHE_MAX_SIZE 接続パラメーター」『Ultra Light データベース管理とリファレンス』
- 「Ultra Light cache_allocation オプション」『Ultra Light データベース管理とリファレンス』

- **Ultra Light でダウンロード専用テーブルをサポート**

  _download_only サフィックス　(Ultra Light Java Edition データベースではない) Ultra Light データベースでは、統合データベース上のテーブルへの変更が同期中にダウンロードされますが、ローカルの変更は Mobile Link に送信されません。サフィックス **_download_only** 付きのテーブルは、ダウンロード専用とマークされます。CREATE TABLE and ALTER TABLE SQL 文の synchronization constraint 句で **SYNCHRONIZE DOWNLOAD** を指定して、テーブルをダウンロード専用に設定することもできます。「Ultra Light ダウンロード専用テーブル」『Ultra Light データベース管理とリファレンス』、「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』、「ALTER TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

  新しい TABLE_IS_DOWNLOAD_ONLY フラグ　テーブルを同期化するときに、コミットされていないクライアント側の変更が原因でダウンロードの衝突が生じる場合があります。Ultra Light および Ultra Light Java Edition では、systable システムテーブルの table_flags カラムに TABLE_IS_DOWNLOAD_ONLY フラグを含むことができるようになりました。

  テーブルの sync=download 属性　アンロードされた Ultra Light データベースの XML フォーマットでテーブルに sync="download" 属性も含まれるようになりました。「systable システムテーブル」『Ultra Light データベース管理とリファレンス』を参照してください。

- **TLS ID をデータベース内に格納できる**　(Ultra Light Java Edition を含まない) Ultra Light データベースでは、X.509 証明書、プライベートキー、およびオプションでクライアントの証明書に署名した一連の証明書認証機関の証明書で構成される TLS ID を作成時に Ultra Light データベースに格納できるようになりました。ulinit および ulload ユーティリティに新しいオプションが追加されました。「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』、「Ultra Light データベースへの XML のロードユーティリティ (ulload)」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light J コードサンプルがデバイスおよびプラットフォームサポート別に整理されました**　コードサンプルは、次のディレクトリに置かれるようになりました。

  ○BlackBerry のサンプルの場合：*%SQLANY12%¥UltraLiteJ¥BlackBerry*

  ○Java J2ME のサンプルの場合：*%SQLANY12%¥UltraLiteJ¥J2ME*

  ○Java J2SE のサンプルの場合：*%SQLANY12%¥UltraLiteJ¥J2SE*

  ○Android のサンプルの場合：*%SQLANY12%¥UltraLiteJ¥Android*

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「サンプルコード」『Ultra Light - Java プログラミング』を参照してください。

プラットフォームとデバイス

- **新しいデバイス**　Ultra Light J プログラミングインターフェイスで Android スマートフォンがサポートされるようになりました。Ultra Light J プログラミングインターフェイスは BlackBerry プラットフォームと Android プラットフォームで共通で、アプリケーションは各プラットフォーム用に Java で開発する必要があります。BlackBerry スマートフォンでは、基礎になる DBMS は Ultra Light の Java 実装ですが、Android では、DBMS は iPhone や iPad、Windows Mobile、および Windows でも提供される C++ バージョンです。

  次の項を参照してください。

  - 「Ultra Light 概要」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light J」『Ultra Light - Java プログラミング』
  - 「Ultra Light J アプリケーションの開発」『Ultra Light - Java プログラミング』
  - 「チュートリアル：Android アプリケーションの構築」『Ultra Light - Java プログラミング』
  - 「Ultra Light J API リファレンス」『Ultra Light - Java プログラミング』

プログラミングインターフェイス

Ultra Light.NET

- **非同期の同期**　非同期の同期をサポートするために次のメンバーが追加されました。

  - **public delegate void ULSyncProgressedDlg(IAsyncResult result, ULSyncProgressData data)**　このメソッドは、同期化中に進捗情報を提供するために呼び出されます。ULSyncProgressedDlg デリゲート [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  - **public IAsyncResult ULConnection.BeginSynchronize()**　このメソッドは、同期化を実行する新しいスレッドを作成して、すぐに返します。ULConnection.BeginSynchronize メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  - **public IAsyncResult ULConnection.BeginSynchronize(Control control, ULSyncProgressedDlg dlg, object state)**　このメソッドは、同期化を実行する新しいスレッドを作成して、すぐに返します。ULConnection.BeginSynchronize メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  - **public void ULConnection.CancelSynchronize(IAsyncResult asyncResult)**　このメソッドは、同期化スレッドに終了を伝えて、すぐに返します。ULConnection.CancelSynchronize メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  - **public void ULConnection.EndSynchronize(IAsyncResult asyncResult)**　このメソッドは非同期で起動された同期が終了するまでブロックします。同期が失敗すると、ULException がスローされます。ULConnection.EndSynchronize メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

- **public bool ULSyncProgressData.IsFinalSyncProgress**　これが同期の最終同期進捗メッセージである場合、このプロパティは true です。
  ULSyncProgressData.IsFinalSyncProgress プロパティ [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

Ultra Light for M-Business Anywhere

この API は、Ultra Light バージョン 12.0.0 および 12.0.1 で廃止されています。

Ultra Light J

- **Android デバイス用のデータベース設定サポート**　新しい DatabaseManager.CreateConfigurationFileAndroid メソッドは、Android デバイス上のファイルに保存される永続的なデータベースの ConfigFileAndroid オブジェクトを確立します。
  ConfigFileAndroid インターフェイス [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **ConfigPersistent インターフェイスへの Andoroid 関連の更新**　次のメンバーは、Ultra Light J の Android サポートの影響を受けます。

メンバー名	状態
enableAesDBEncryption メソッド	Android 専用の新しいメソッドです。データベースの AES 暗号化を可能にします。ConfigPersistent.enableAesDBEncryption メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
getAutoCheckpoint メソッド	BlackBerry では廃止され、Android では有効です。
getConnectionString メソッド	Android 専用の新しいメソッドです。SetConnectionString で登録される接続文字列を取得します。ConfigPersistent.getConnectionString メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
getCreationString メソッド	Android 専用の新しいメソッドです。SetCreationString で登録される作成文字列を取得します。ConfigPersistent.getCreationString メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

メンバー名	状態
getDatabaseKey メソッド	Android 専用の新しいメソッドです。SetDatabaseKey で登録されるデータベース暗号化キーを取得します。ConfigPersistent.getDatabaseKey メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
getUserName メソッド	Android 専用の新しいメソッドです。setUserName で設定されるユーザーの名前を取得します。ConfigPersistent.getUserName メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
setAutocheckpoint メソッド	BlackBerry では廃止され、Android では有効です。
setConnectionString メソッド	Android 専用の新しいメソッドです。データベースの作成または接続に使用される接続文字列を設定します。ConfigPersistent.setConnectionString メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
setCreationString メソッド	Android 専用の新しいメソッドです。データベースを作成するために使用される作成文字列を設定します。ConfigPersistent.setCreationString メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
setDatabaseKey メソッド	Android 専用の新しいメソッドです。暗号化キーを設定します。ConfigPersistent.setDatabaseKey メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
setEncryption メソッド	BlackBerry 専用です。
setUserName メソッド	Android 専用の新しいメソッドです。ユーザーの名前を設定します。ConfigPersistent.setUserName メソッド [Android] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

詳細については、ConfigPersistent インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Connection インターフェイスへの Andoroid 関連の更新**　次のメンバーは、Ultra Light J の Android サポートの影響を受けます。

メンバー名	ステータス
checkpoint メソッド	BlackBerry では廃止され、Android では有効です。
emergencyShutdown メソッド	Android では使用できません。
getDatabaseId メソッド	Android では使用できません。
getDatabaseProperty メソッド	Android では使用できません。
getState メソッド	Android では使用できません。
isSynchronizationDeleteDisabled メソッド	Android では使用できません。
OPTION_BLOB_FILE_BASE_DIR 変数	BlackBerry および Java SE 専用です。

詳細については、Connection インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **DatabaseManager クラスへの Andoroid 関連の更新**　次のメンバーは Android では使用できません。

  ○createConfigurationNonPersistent メソッド

  ○createFileTransfer メソッド

  詳細については、DatabaseManager クラス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **ResultSet インターフェイスへの Andoroid 関連の更新**　次のメンバーは Android では使用できません。

  ○getBoolean メソッド

  ○getBytes メソッド

  ○getClobReader メソッド

  ○getDate メソッド

  ○getDecimalNumber メソッド

  ○getDouble メソッド

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○getFloat メソッド

○getLong メソッド

○getSize メソッド

○getString メソッド

○getUUIDValue メソッド

○isNull メソッド

詳細については、ResultSet インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

Ultra Light の動作の変更と廃止予定機能

次に、バージョン 12.0.1 で導入された Ultra Light の廃止予定機能と動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

動作の変更

- **SET OPTION 動作への変更**　Ultra Light は、次の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するようになりました。「SET OPTION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **永続的なネットワークプロトコルオプションのデフォルト設定が true になりました**　Mobile Link クライアント (Ultra Light J を除く) の永続的なネットワークプロトコルオプションのデフォルト値が "true" になりました。永続的な接続を使用する同期は、特に HTTPS の場合、非永続的な接続よりも高速です。ただし、仲介サーバーが非永続的な接続を要求したり、クライアントで仲介サーバーが永続的な接続に対応していないことが検出されたりすると、残りの同期化セッションで接続が自動的に非永続的な接続にダウングレードされます。「persistent」『Mobile Link クライアント管理』を参照してください。

- **SQLE_COMMUNICATIONS_ERROR (-85) エラーメッセージが SQLE_MOBILINK_COMMUNICATIONS_ERROR (-1305) に変わりました**　通信の問題に関するすべての情報がこのエラーに含まれます。別々のストリームエラーオブジェクトからこれらの値にアクセスする必要はなくなりましたが、これらの値を参照する既存のコードは引き続き機能します。このエラーのパラメーターはストリームエラーコード、パラメーターおよびシステムコードです。「通信エラーが発生しました。」『エラーメッセージ』を参照してください。

- **SQLE_UNKNOWN_PROPERTY エラーメッセージ**　不明なプロパティ名に対して SQLE_UNKNOWN_PROPERTY が通知されるようになりました。「'%1' は認識できないプロパティです。」『エラーメッセージ』を参照してください。

- **SQLE_MOBILINK_AUTHENTICATION_FAILED エラーメッセージ**　同期化中に SQLE_INVALID_LOGON ではなく SQLE_MOBILINK_AUTHENTICATION_FAILED が通知され、認証ステータスと値が含まれます。「Mobile Link が値 '%2' の認証ステータス '%1' を返したため、同期に失敗しました。」『エラーメッセージ』を参照してください。

管理ツールの新機能

次に、バージョン 12.0.1 で導入された管理ツールの追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **OEM での DBISQL、Sybase Central、DBConsole および ML Monitor で使用する優先ディレクトリの指定**　管理ツールとともに *OEM.ini* ファイルを配備する場合、ファイルに管理ツールの優先ディレクトリを指定する次の行が含まれるようにします。

  ```
  [preferences]
  directory=preferences_file_directory
  ```

  「管理ツールの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

Sybase Central プラグインの新機能

次に、バージョン 12.0.1 で導入された Sybase Central のプラグインの追加機能を示します。

SQL Anywhere プラグインの新機能

- **新しいアプリケーションプロファイリングの推奨内容**　**[アプリケーションプロファイリング] ウィザード**を使用してデータベースのスキーマに基づいてデータベースの全体的なパフォーマンスを確認する場合、このウィザードによって、テーブル内のカラムがチェックされます。幅の広いカラムの大半が幅の狭いカラムよりも前にある場合、ウィザードにより推奨内容が作成されます。幅の広いカラムが幅の狭いカラムよりも前にあるように並べると、応答時間に悪影響を及ぼす場合があります。幅の狭いカラムは、アクセス頻度が低い場合以外は、テーブル宣言で幅の広いカラムよりも前に定義します。

  幅の広いカラムとは、15 バイトより大きなサイズのカラム、LONG 型データのカラム (LONG VARCHAR など)、または XML として定義されたカラムを指します。「テーブル内のカラムの順序の確認」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ユーザーのパーミッションの管理および表示方法の強化**　プラグインでユーザーのパーミッションを表示する方法に多数の強化が行われました。データベースオブジェクトに明示的に設定されたパーミッションの表示と変更がより簡単になりました。データベースオブジェクトを選択すると、パーミッション情報が右ウィンドウ枠のタブに表示されるようになりました。カラムのパーミッションは対応するテーブルパーミッションと共に表示されます。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「ユーザーへのパーミッションの付与」『SQL Anywhere サーバー データベース管理』を参照してください。

Sybase Central プラグイン動作の変更

次に、バージョン 12.0.1 で導入された Sybase Central のプラグインの動作変更を示します。

SQL Anywhere プラグインの新機能

- **Sybase Central は、SQL Anywhere バージョン 10x 以降のデータベースのみサポート**　バージョン 9 のデータベースサーバーと、バージョン 9 のソフトウェアで作成されたデータベースのサポートが SQL Anywhere プラグインから削除されました。データベースをアンロードして、再ロードファイル、新しいデータベース、または既存のデータベースに再ロードする場合、バージョン 9 以降のデータベースサーバーで稼働している、バージョン 5、6、7、8 または 9 のソフトウェアで作成したデータベースには引き続き接続できます。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

Interactive SQL の新機能

次に、バージョン 12.0.1 で導入された Interactive SQL の追加機能を示します。

- **[接続] ウィンドウが Sybase IQ をサポート**　Interactive SQL を使用して、Sybase IQ データベースに接続できます。**[接続]** ウィンドウ、**[データベースタイプを変更]** をクリックし、次に **[Sybase IQ]** をクリックします。

- **テキスト補完**　次のリストは、SQL と Sybase Central のテキスト補完機能に関する変更を示しています。

  ○**[SQL 文]** ウィンドウ枠に入力すると、デフォルトでテキスト補完ウィンドウが自動的に表示されるようになりました。テキスト補完ウィンドウは、**[編集]** » **[テキスト補完を開く]** をクリックするか、[Ctrl+Space] キーを押しても開くことができます。

  ○デフォルトでは、SQL キーワードおよび補完されたデータベースオブジェクト名は二重引用符で囲まれます。

  ○SQL 文とキーワードが候補のリストに含まれるようになりました。

  ○テキスト補完ウィンドウで文字列は大文字と小文字が区別されるが、ID は区別されないデータベースを取り扱うことができるようになりました。

  ○次のキーボードショートカットが変更されました。

以前のショートカット	新しいショートカット	説明
[Tab]	[Ctrl + A]	内容に関係なく一致するデータのリストを表示します。

以前のショートカット	新しいショートカット	説明
+	[Ctrl + プラス記号 (+)]	項目をパラメーターリスト付きで **[SQL 文]** ウィンドウ枠に追加します。
*	[Ctrl + アスタリスク]	項目をパラメーターおよびタイプリスト付きで **[SQL 文]** ウィンドウ枠に追加します。
'	[Ctrl + 二重引用符 (")]	項目を引用符で囲んで **[SQL 文]** ウィンドウ枠に追加します。
なし	[Tab]	選択内容を受け入れ、テキスト補完ウィンドウを閉じます。

ショートカットの完全なリストについては、「テキスト補完キーボードショートカット」『SQL Anywhere サーバー データベース管理』を参照してください。

詳細については、「テキスト補完の使用」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい [インポート／エクスポート] オプション**　**[オプション] » [インポート／エクスポート]** をクリックすると、NULL 値を **[NULL 値のエクスポート方法]** フィールドにエクスポートする方法を指定できます。「output_nulls オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』を参照してください。

Interactive SQL の動作の変更

次に、バージョン 12.0.1 で導入された Interactive SQL の変更点を示します。

- **READ 文エンコードのアルゴリズムの変更**　Interactive SQL で READ 文を実行するとき、データの読み込みに使用されるエンコードが次の順序で決定されるようになりました。

  1. ENCODING 句で指定されたエンコード (この句が指定されている場合)。
  2. ファイル内でバイト順マークで指定されたエンコード (バイト順マークが指定されている場合)。
  3. デフォルトのテキスト形式のエンコード (この形式のエンコードが指定されている場合)。
  4. 実行しているプラットフォームのデフォルトのエンコード。英語版 Windows コンピューターでは、デフォルトのエンコードは 1252 です。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「READ 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

SQL Anywhere モニターの新機能

次に、バージョン 12.0.1 で導入された SQL Anywhere モニターの追加機能を示します。

- **モニターで SNTP サーバーを通して、電子メールで警告を送信できるようになりました** SQL Anywhere モニターが TLS 接続を必要とする SMTP サーバーを使用して、電子メールによる警告を送信できるようになりました (GMail など)。「モニターによる警告電子メールの送信の有効化」『SQL Anywhere サーバー データベース管理』を参照してください。

- **読み込み専用のスケールアウトおよびデータベースミラーリングシステムのモニターデータベース** モニターは読み込み専用のスケールアウトシステムのルートノードのデータベースサーバーおよびミラーリングシステムのプライマリデータベースサーバーのモニタリングに対応します。ミラーリングシステムで、モニターはフェールオーバーが発生しても常にプライマリデータベースサーバーをモニタリングするように設定できます。新しいウィジェットである、**[SQL Anywhere スケールアウトトポロジ]** ウィジェットは、SQL Anywhere のミラーリングおよびスケールアウトシステムのトポロジ情報を表示します。

  読み込み専用のスケールアウトシステムのルートノードおよびミラーリングシステムのプライマリデータベースのリソースを作成するには、**[リソースの追加]** ウィンドウを使用し、**[SQL Anywhere サーバー]** リソースを作成します。「リソースの追加」『SQL Anywhere サーバー データベース管理』を参照してください。

  読み込み専用のスケールアウトシステムとその他のデータベースサーバーのルートノードのリソース設定に違いはありません。リソースが設定または移行されると、モニターはそのリソースが読み込み専用のスケールアウトシステムのルートノードであることを検出します。

  ミラーリングシステムのプライマリデータベースのリソースには、特別な設定が必要です。フェールオーバーが発生してもモニターが常にプライマリデータベースサーバーをモニタリングするようにするには、次のオプションを使用してリソースを設定する必要があります。

  ○**[ホスト]** フィールドでプライマリサーバーとミラーサーバーのホスト名とポート番号をカンマ区切りのリストで指定します。(例 : my-primary-server:2638,my-mirror-server:49152)。

  ○**[ポート]** フィールドが空であることを確認します。

  ○**[サーバー]** フィールドにプライマリサーバーの代替サーバー名を入力します。つまり、データベースミラーリングシステムのプライマリサーバーとして動作するデータベースサーバーに接続するためにクライアントで使用する名前を指定します。

  [SQL Anywhere スケールアウトトポロジ] をダッシュボードに追加する必要があります。これは新しいダッシュボードの作成時に表示されるデフォルトのウィジェットの 1 つではないからです。「ウィジェット」『SQL Anywhere サーバー データベース管理』を参照してください。

  「レッスン 7 : (オプション) 読み込み専用スケールアウトシステムの一部であるデータベースをモニタリングするためのモニターの設定」『SQL Anywhere サーバー データベース管理』と

「レッスン 3：データベースミラーリングシステムのモニタリング」『SQL Anywhere サーバー データベース管理』を参照してください。

さらに、新しい警告、**[切断されたスケールアウトノードの数が指定したスレッショルドを越えた場合に警告します:]** が追加されました。「警告スレッショルドの指定」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Mobile Link サーバーの新しいモニターのメトリックと警告**　Mobile Link サーバーリソースのモニターのメトリックに **[統合を使用できますか]** というメトリックが含まれるようになりました。このメトリックは、Mobile Link サーバーが統合データベースに接続できない場合に **[True]** を示します。統合データベースを使用できなくなると、**[統合データベースを使用できません]** 警告が発効されます。

- **Mobile Link リソースのメトリックの新しい名前**　次の Mobile Link サーバーリソースの SQL Anywhere モニターのメトリックが変わりました。

以前の名前	新しい名前
バイト読み取り率	TCP バイト読み取り率
バイト書き込み率	TCP バイト書き込み率

- **[警告の通知] ウィンドウ**　**[警告の通知]** ウィンドウに日付および送信された電子メールの種類別の電子メールセットの数を制限する 2 つのオプションが含まれるようになりました。

  優先度の高い警告の場合のみ電子メール通知を送る　優先度の高い警告が発行された場合のみ、電子メール通知を受け取る場合、このオプションを設定します。

  1 日に指定した数以上の電子メールを送信しない　1 日当たりで各ユーザーに送信される通知メールの数を制限する場合にこのオプションを選択します。

  「モニターによる警告電子メールの送信の有効化」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[警告の通知] ウィンドウ**　ユーザーが警告を選択して、**[削除]** ボタンをクリックすると、**[警告の削除]** ウィンドウが表示され、ユーザーに以下のオプションのいずれかを選択するように求めます。

  選択した警告を削除する　**[警告リスト]** ウィジェットで選択した警告を削除する場合のみ、このオプションを選択します。

  次より前に受信した警告を削除する　指定した時間より前に受信したすべての警告を削除するには、このオプションを選択します。

  次のリソースの警告を削除する　選択したリソースのすべての警告を削除するには、このオプションを選択します。

  「警告の削除」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

SQL Anywhere モニターの動作の変更

次に、バージョン 12.0.1 で導入された SQL Anywhere モニターの変更点を示します。

- **エクスポートできるデータの最大量**　ファイルにエクスポートできるデータの最大量は 25 メトリックまたは 100 万ポイントです。「メトリックのエクスポート」『SQL Anywhere サーバー データベース管理』を参照してください。

マニュアルの強化

次に、バージョン 12.0.1 で SQL Anywhere マニュアルに加えられた変更を示します。

- **SQL Anywhere チュートリアルのリスト**　このマニュアルに、SQL Anywhere のすべてのチュートリアル一覧が掲載されるようになりました。「SQL Anywhere チュートリアルのリスト」『SQL Anywhere 12 紹介』を参照してください。

バージョン 12.0.0 の新機能

バージョン 10 以前の SQL Anywhere の新機能と動作変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

製品全体の新機能

次に、バージョン 12.0.0 で導入された製品全体の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **設定ファイルでサポートされているエスケープ文字**　設定ファイルの解析が強化され、¥ のエスケープシーケンスとして ¥¥、" のエスケープシーケンスとして ¥" がサポートされるようになりました。「設定ファイルのエスケープ文字」『SQL Anywhere サーバー データベース管理』を参照してください。

- **展開ウィザードでの 64 ビットの展開のサポート**　以前のバージョンの展開ウィザードは 32 ビットの展開をサポートしていました。新バージョンでは、64 ビットの展開もサポートされるようになりました。「Deployment ウィザード」『SQL Anywhere サーバー プログラミング』を参照してください。

製品全体の動作の変更

次に、バージョン 12.0.0 で導入された製品全体の動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **Sybase Replication Server 用の SQL Anywhere Replication Agent がサポートされない**　Sybase Replication Server 用の SQL Anywhere Replication Agent は、バージョン 12 ではサポートされません。Mobile Link や SQL Remote などの代替のレプリケーションまたは同期テクノロジを使用してください。「Mobile Link テクノロジ」『Mobile Link クイックスタート』と「SQL Remote システム」『SQL Remote』を参照してください。

  この変更により、次の変更がソフトウェアに追加されました。

  - **a_change_log DBTools 構造体**　ignore_ltm_trunc メンバーは、サポートされなくなりました。
  - **LTMGeneration データベースプロパティ**　このプロパティはシステムで使用するために予約されています。
  - **LTMTrunc データベースプロパティ**　このプロパティはシステムで使用するために予約されています。
  - **Log Transfer Manager ユーティリティ (dbltm)**　このユーティリティは削除されました。

- **ログ変換ユーティリティ (dbtran)**　-is オプションで、値 RepServer がサポートされなくなりました。

  -rsu オプションは削除されました。

- **サービスユーティリティ (dbsvc)**　Replication Agent 用にサービスを作成できなくなりました。SQLANYLTM サービスグループはサポートされなくなりました。

  -w オプションと -t オプションで、値 **dbltm** がサポートされなくなりました。

- **サポートユーティリティ (dbsupport)**　このユーティリティから、SQL Anywhere Replication Agent (dbltm) の情報が返されなくなりました。

- **トランザクションログユーティリティ (dblog)**　-g オプションと -il オプションは、サポートされなくなりました。「トランザクションログユーティリティ (dblog)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **replicate_all データベースオプション**　このオプションは削除されました。

- **delete_old_logs データベースオプション**　Replication Agent では、このオプションはサポートされません。

- **ALTER PROCEDURE 文**　次の構文は、サポートされなくなりました。

  ALTER PROCEDURE [*owner.*]*procedure-name*
  REPLICATE { ON | OFF }

  「ALTER PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER TABLE 文**　**REPLICATE { ON | OFF }** 句はサポートされなくなりました。「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **実行プログラムをデーモンとして実行する場合にユーザーの umask 設定が考慮される**　以前のリリースでは、UNIX で実行プログラムをデーモンとして実行 (-ud オプションを指定して起動) した場合、ユーザーの umask 設定は無視され、グループパーミッションおよび他の読み込みと書き込みのパーミッションを使用して新しいファイルを作成する、umask(0) が呼び出されていました。SQL Anywhere 12 の実行プログラムをデーモンとして起動すると、umask(0) は呼び出されず、ユーザーの umask 設定が考慮されます。現在のユーザーの umask 設定によって実行プログラムのパーミッションが制御されるため、実行プログラムを起動する前に、ユーザーの umask 値が目的のレベルに設定されていることを確認してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

この動作の変更は、次の実行プログラムに適用されます。

- dbeng12
- dbsrv12
- dbltm
- dbmlsync
- dbns12
- dbremote
- mlsrv12
- uleng12

SQL Anywhere の新機能

次に、SQL Anywhere バージョン 12.0.0 の新機能を示します。

サポートされるプラットフォームのリストに加えられた変更については、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

主な機能

次に、SQL Anywhere バージョン 12.0.0 の主な機能を示します。

- **新しい空間データサポート**　SQL Anywhere 12.0.0 の新しい空間データ機能をサポートするために、次の機能が追加されました。この機能を使用するには、データベースをアップグレードする必要があります。

  注意
  32 ビット Windows と 32 ビット Linux の空間データサポートには、SSE2 命令をサポートする CPU が必要となります。これは、Intel Pentium 4 (2001 年リリース) 以降と AMD Opteron (2003 年リリース) 以降でサポートされます。

  SQL 文　空間機能をサポートするために、次のような SQL 文の強化が行われました。

  - **SHAPEFILE 句**　FROM 句の OPENSTRING サブ句で、新しい SHAPEFILE フォーマットオプションを使用できます。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

    また、LOAD TABLE 文の FORMAT 句で新しい SHAPEFILE フォーマットオプションを使用できます。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **CREATE SPATIAL REFERENCE SYSTEM 文**　空間参照系を作成するか、置き換えます。「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **ALTER SPATIAL REFERENCE SYSTEM 文**　既存の空間参照系の設定を変更します。空間参照系を変更する前に、注意事項について「備考」の項を参照してください。「ALTER

SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DROP SPATIAL REFERENCE SYSTEM 文**　空間参照系を削除します。「DROP SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE SPATIAL UNIT OF MEASURE 文**　空間測定単位を作成するか、置き換えます。「CREATE SPATIAL UNIT OF MEASURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DROP SPATIAL UNIT OF MEASURE 文**　空間測定単位を削除します。「DROP SPATIAL UNIT OF MEASURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Interactive SQL の変更　Interactive SQL に新しいビューアーツールの **[空間ビューアー]** が追加され、空間ジオメトリを表示できるようになりました。ビューアーの上部で空間データを問い合わせると、ビューアーの下部に結果がイメージとして表示されます。「空間データのイメージとしての表示 (Interactive SQL の場合)」『SQL Anywhere サーバー 空間データサポート』を参照してください。

また、Interactive SQL で結果ローを確認する場合、新しい **[空間プレビュー]** タブを使用して、ジオメトリを SVG (Scalable Vector Graphic) としてプレビューできるようになりました。「空間データのイメージとしての表示 (Interactive SQL の場合)」『SQL Anywhere サーバー 空間データサポート』を参照してください。

新しいデータ型、メソッド、コンストラクター　新しい型、メソッド、コンストラクターが追加され、空間データにアクセスし、そのデータをモデル化および分析できるようになりました。「空間データへのアクセスとそのデータの操作」『SQL Anywhere サーバー 空間データサポート』を参照してください。

また、空間データにアクセスし、そのデータを操作するときに通常の SQL 関数と似た処理を行う、数多くの空間互換関数を使用できるようになりました。これらの関数は、他の製品との互換性を確保するために用意されました。これらの関数では SQL Anywhere の空間メソッドとコンストラクターを利用します。「空間互換関数」『SQL Anywhere サーバー 空間データサポート』を参照してください。

新しい関数とシステムプロシージャー　データベースの空間データをサポートするために、次の関数とシステムプロシージャーが追加されました。

- **TREAT 関数**　ジオメトリ式の宣言されたタイプをサブタイプに変更できます。「TREAT 関数 [データ型変換]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_describe_shapefile システムプロシージャー**　ESRI シェイプファイルに含まれるカラムの名前と型を記述します。このシステム機能は、空間機能とともに使用します。「sa_describe_shapefile システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **sa_install_feature システムプロシージャー**　SQL Anywhere のインストール時にデータベースに存在していなかった追加の機能をインストールします。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **st_geometry_dump システムプロシージャー**　入力に含まれるジオメトリオブジェクトの 1 つを表す各ローとともに、ジオメトリオブジェクトを結果セットに展開します。「st_geometry_dump システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

ウィザード　Sybase Central では、空間データ機能をサポートするために、次のウィザードが追加されました。

- **空間参照系の作成ウィザード**　**空間参照系の作成ウィザード**を使用すると、新しい空間参照系を作成できます。「空間参照系の作成 (Sybase Central の場合)」『SQL Anywhere サーバー 空間データサポート』を参照してください。

- **測定単位の作成ウィザード**　**測定単位の作成ウィザード**を使用すると、空間データで使用する新しい測定単位を作成できます。「測定単位の作成 (Sybase Central の場合)」『SQL Anywhere サーバー 空間データサポート』を参照してください。

カタログの変更　新しい空間データサポートの一環として、カタログは次のように変更されました。

- **SYSSPATIALREFERENCESYSTEM システムビュー**　SYSSPATIALREFERENCESYSTEM システムビューの各ローは、データベースに定義されている空間参照系に関する記述です。「SYSSPATIALREFERENCESYSTEM システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SYSUNITOFMEASURE システムビュー**　SYSUNITOFMEASURE システムビューの各ローは、データベースに定義されている測定単位に関する記述です。「SYSUNITOFMEASURE システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ST_GEOMETRY_COLUMNS 統合ビュー**　ST_GEOMETRY_COLUMNS システムビューの各ローは、データベースに定義されている空間カラムに関する記述です。「ST_GEOMETRY_COLUMNS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ST_SPATIAL_REFERENCE_SYSTEMS 統合ビュー**　ST_SPATIAL_REFERENCE_SYSTEMS システムビューの各ローは、データベースに定義されている空間参照系に関する記述です。「ST_SPATIAL_REFERENCE_SYSTEMS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ST_UNITS_OF_MEASURE 統合ビュー**　ST_UNITS_OF_MEASURE システムビューの各ローは、データベースに定義されている測定単位に関する記述です。「ST_UNITS_OF_MEASURE 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

データベースオプションとプロパティ　空間データ機能をサポートするために、次のデータベースオプションとプロパティが追加されました。

- **st_geometry_asbinary_format オプション**　ジオメトリからバイナリへの空間データ値の変換方法を制御します。「st_geometry_asbinary_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_astext_format オプション**　ジオメトリからテキストへの空間データ値の変換方法を制御します。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_asxml_format オプション**　ジオメトリから XML への空間データ値の変換方法を制御します。「st_geometry_asxml_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_describe_type オプション**　空間データの記述方法を制御します。「st_geometry_describe_type オプション」『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_on_invalid オプション**　ジオメトリが基本的な検証に失敗した場合の動作を制御します。「st_geometry_on_invalid オプション」『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_asbinary_format 接続プロパティ**　空間値がジオメトリからバイナリに変換される方法を示す値を返します。st_geometry_asbinary_format 接続プロパティ『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_astext_format 接続プロパティ**　空間値がジオメトリからテキストに変換される方法を示す値を返します。st_geometry_astext_format 接続プロパティ『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_asxml_format 接続プロパティ**　空間値がジオメトリから xml に変換される方法を示す値を返します。st_geometry_asxml_format 接続プロパティ『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_describe_type 接続プロパティ**　空間データ値をクライアントに対してどのような方法で記述するかを示す値を返します。st_geometry_describe_type 接続プロパティ『SQL Anywhere サーバー データベース管理』を参照してください。
- **st_geometry_on_invalid 接続プロパティ**　ジオメトリが基本的な検証に失敗した場合の動作を示す値を返します。st_geometry_on_invalid 接続プロパティ『SQL Anywhere サーバー データベース管理』を参照してください。

SYS_SPATIAL_ADMIN_ROLE グループ　このグループのメンバーシップが与えられたユーザーは、空間参照系と計測単位を作成、変更、または削除できます。「空間パーミッションの付与」『SQL Anywhere サーバー 空間データサポート』を参照してください。

SQL Anywhere の空間サポートの詳細については、「空間データの使用」『SQL Anywhere サーバー 空間データサポート』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **読み込み専用のスケールアウト**　読み込み専用のスケールアウトシステムで SQL Anywhere を使用できるようになりました。この構成では、1 つのデータベースサーバー (ルートノード) でデータベースの読み込み／書き込みコピーが実行される一方、他のデータベースサーバーでデータベース (コピーノード) の読み込み専用コピーが実行されます。読み込み専用コピーを使用すると、データベースへの読み込みアクセスを必要とするレポートなどの操作の負荷を軽減できます。読み込み専用のスケールアウトは、そのままで、またはデータベースミラーリングと共に使用できます。この機能を使用するには、既存のデータベースをアップグレードまたは再構築する必要があります。

  %SQLANYSAMP12%¥SQLAnywhere¥DBMirror に、データベースミラーリングシステムとスケールアウトシステムを組み合わせて使用するサンプルが追加されました。

  次の項を参照してください。

  - 「SQL Anywhere の読み込み専用のスケールアウト」『SQL Anywhere サーバー データベース管理』
  - 「NodeType (NODE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』
  - 「CREATE MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「ALTER MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「SET MIRROR OPTION 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
  - MIRROR SERVER 句：「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「SYSMIRROROPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』
  - 「SYSMIRRORSERVER システムビュー」『SQL Anywhere サーバー SQL リファレンス』
  - 「SYSMIRRORSERVEROPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』
  - MirrorRole データベースプロパティ『SQL Anywhere サーバー データベース管理』
  - MirrorServerState プロパティと MirrorState プロパティ：「sa_mirror_server_status システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』

- **データベースミラーリングの強化**　データベースサーバーのコマンドラインでミラーリングの設定を指定する代わりに、SQL 文を使用してデータベースミラーリングシステムを設定できるようになりました。この機能を使用するには、既存のデータベースをアップグレードまたは再構築する必要があります。

次の項を参照してください。

- 「CREATE MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
- 「ALTER MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
- 「SET MIRROR OPTION 文」『SQL Anywhere サーバー SQL リファレンス』
- 「DROP MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
- 「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』に追加された MIRROR SERVER 句
- 「SYSMIRROROPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- 「SYSMIRRORSERVER システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- 「SYSMIRRORSERVEROPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- MirrorRole データベースプロパティ『SQL Anywhere サーバー データベース管理』
- 拡張データベースプロパティ：MirrorServerState、MirrorState
- 「sa_mirror_server_status システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- データベースミラーリングの動作の変更と廃止予定機能75 ページ

- **Host 接続パラメーター**　新しい Host 接続パラメーターでは、ホスト名 (または IP アドレス) とオプションのポート番号を使用して、データベースサーバーがある場所をクライアントに知らせます。クライアントと異なるコンピューターで実行されているデータベースサーバーに接続する場合は、この接続パラメーターを使用することをおすすめします。「Host 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **自動統計管理の強化**　SQL Anywhere 12 には、データベースカラムに関する統計の自動保守を向上させる統計ガバナーが備えられています。データベース内の各統計の正常性と有用性が自動的に評価され、統計を自己モニターして自己回復できるように、必要な保守が実行されます。統計の保守はバックグラウンドで実行され、データベースサーバーのパフォーマンスに対する大幅な負荷は発生しません。「統計ガバナーが統計を保守する方法」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  sa_server_option システムプロシージャーで、統計の管理に役立つオプション DropBadStatistics、DropUnusedStatistics、StatisticsCleaner がサポートされるようになりました。「sa_server_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **シーケンス**　SQL Anywhere でシーケンスの生成がサポートされるようになりました。シーケンスは、アプリケーションによってユニークキーの値を生成する場合に使用されます。シーケンス値を使用すると、アプリケーションは同時実行性とパフォーマンスの問題を回避できます。

  Sybase Central の SQL Anywhere プラグインを使用して、シーケンスを作成、編集、管理することもできます。たとえば、**シーケンスジェネレーターの作成ウィザード**を使用して、データベースで新しいシーケンスを作成します。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

参照：

- 「ユニークな値を生成するためのシーケンスの使用」『SQL Anywhere サーバー SQL の使用法』
- 「CREATE SEQUENCE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「ALTER SEQUENCE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「DROP SEQUENCE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「SYSSEQUENCE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- 「SYSSEQUENCEPERM システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- SEQUENCE 句、「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』
- GRANT USAGE ON SEQUENCE 構文、「GRANT 文」『SQL Anywhere サーバー SQL リファレンス』
- REVOKE USAGE ON SEQUENCE 構文、「REVOKE 文」『SQL Anywhere サーバー SQL リファレンス』

シーケンスを使用するには、既存のデータベースをアップグレードまたは再構築する必要があります。

- **マルチプログラミングレベルの強化**　ネットワークデータベースサーバー (dbsrv12) では、デフォルトでマルチプログラミングレベルが自動的に制御されるようになりました。この動作により、データベースサーバーではスループットが向上し、DBA の介入なしで負荷の変更に適応できます。

  データベースサーバーは、起動時にサービス要求に使用されるワーカーのプールを作成します。ワーカーの数は、サーバーの現在のマルチプログラミングレベルです。プールには最小値と最大値があり、現在のマルチプログラミングレベルは常にこの範囲内にあります。DBA は、起動時にデータベースサーバーオプションを使用するか、データベースサーバーの実行中に sa_server_option システムプロシージャーを使用して、最小値と最大値を変更できます。

  データベースサーバーのマルチプログラミングレベルを制御できるように、次のオプションが追加されました。

データベースサーバーオプション	sa_server_option 値	説明
「-gn dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』	CurrentMultiProgrammingLevel	データベースサーバーの初期マルチプログラミングレベルを設定します。
「-gna dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』	AutoMultiProgrammingLevel	データベースサーバーのマルチプログラミングレベルの動的チューニングをオンまたはオフにします。
「-gnh dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』	MaxMultiprogrammingLevel	データベースサーバーが同時に実行できるタスクの最大数を設定します。

データベースサーバーオプション	sa_server_option 値	説明
「-gnl dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』	MinMultiProgrammingLevel	データベースサーバーが同時に実行できるタスクの最小数を設定します。
「-gns dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』	AutoMultiProgrammingLevelStatistics	マルチプログラミングレベルの自動変更に関する統計をデータベースサーバーメッセージログに出力するかどうかを制御します。

SQL Anywhere のマルチプログラミングレベルの詳細については、「マルチプログラミングレベルのデータベースサーバー設定」『SQL Anywhere サーバー データベース管理』を参照してください。

- **即時マテリアライズドビューの外部ジョインのサポート**　定義内に OUTER JOIN があるマテリアライズドビューを即時宣言できるようになりました。「マテリアライズドビューの制限」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **DML 文からの選択**　SELECT 文の FROM 句で DML 文を指定できるようになりました。この機能を使用すると、UPDATE、INSERT、DELETE、または MERGE 文によって変更されたローが移植された派生テーブルに対して SQL クエリを作成し、これらの更新されたローからアプリケーションへ値を返すことができます。

  この機能の最も一般的な用途は、アプリケーションによって変更されたローの値の確認や検査です。以前のバージョンでは、このことは、トリガーおよび複数の SQL 文を使用することによって実行できました。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』と「DML 文に対する SELECT」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **全文検索機能による外部事前フィルターライブラリと外部単語区切りライブラリのサポート**　新しい API が追加されたことによって、全文インデックスを作成および更新するときに、外部事前フィルターライブラリと外部単語区切りライブラリに接続できるようになりました。このことは、XML、PDF、Word のようなドキュメントフォーマットを使用し、内容にインデックスを付ける前に、不要なタグとメタデータを削除できることを意味します。サンプルの単語区切りライブラリを使用して、言語またはアプリケーション固有の単語区切りを実行できます。サンプルの事前フィルターライブラリと単語区切りライブラリを利用して独自のライブラリを作成したり、サードパーティのライブラリを使用したりできます。「高度：外部単語区切りライブラリと事前フィルターライブラリ」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  データベースサーバーを実行しているシステムに Microsoft Office がインストールされている場合は、Word や Excel などの Office ドキュメント用の IFilter を使用できます。サーバーに Acrobat Reader がインストールされている場合は、PDF IFilter を使用できます。

  ALTER TEXT CONFIGURATION 文に PREFILTER EXTERNAL NAME 句と TERM BREAKER EXTERNAL NAME 句が追加され、外部ライブラリの名前とロケーションを指定

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

できるようになりました。「ALTER TEXT CONFIGURATION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

ISYSTEXTCONFIG システムテーブルは、トークン化か事前フィルタリングまたはその両方に使用されるエントリポイントと外部ライブラリに関する情報を格納するように変更されました。具体的には、prefilter カラムのデータ型が LONG VARCHAR になり、外部事前フィルターライブラリのエントリポイントとライブラリ名を格納するように変更されました。外部単語区切りライブラリのエントリポイントとライブラリ名を格納するために、新しい LONG VARCHAR カラムの external_term_breaker が追加されました。「SYSTEXTCONFIG システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

外部事前フィルターライブラリと外部単語区切りライブラリを使用するには、データベースをアップグレードする必要があります。

- **チェックサムの強化**　データベースページの書き込みチェックサム (ページがディスクに書き込まれるときにのみ作成されるチェックサム) を作成するかどうかを、データベースバージョンに基づいて、データベースサーバーが決定するようになりました。デフォルトでは、バージョン 10 と 11 のデータベースでグローバルチェックサムは無効ですが、バージョン 12 のデータベースでグローバルチェックサムは有効です。バージョン 12 のデータベースサーバーで以前のバージョンのデータベースを起動すると、データベースサーバーのデフォルトの動作で書き込みチェックサムは有効です。バージョン 12 のデータベースの場合、データベースサーバーのデフォルトの動作で書き込みチェックサムは無効です。これは、バージョン 12 のデータベースでは、デフォルトでグローバルチェックサムが有効になっているためです。新しいデータベースでチェックサムがデフォルトで有効73 ページを参照してください。

  データベースまたはデータベースサーバーの起動時に START DATABASE 文の CHECKSUM 句または -wc オプションを使用して、データベースサーバーの書き込みチェックサムの動作を変更できます。「-wc dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』、「-wc dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』、「START DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ALTER DATABASE 文の CHECKSUM 句を使用して、データベースのチェックサムを無効にできます。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

データベース接続

次に、SQL Anywhere バージョン 12.0.0 でのデータベース接続の強化を示します。

- **テンポラリ接続の指定**　テンポラリ接続を使用して、バックアップの実行やデータベースの初期化などの操作が行われます。sa_conn_info システムプロシージャー、sa_conn_list システ

ムプロシージャー、Name 接続プロパティ、ParentConnection 接続プロパティを使用して、テンポラリ接続に関する情報を取得できます。次の項を参照してください。

- 「テンポラリ接続」『SQL Anywhere サーバー データベース管理』
- Name 接続プロパティ『SQL Anywhere サーバー データベース管理』
- ParentConnection 接続プロパティ『SQL Anywhere サーバー データベース管理』
- 「sa_conn_info システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「sa_conn_list システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』

- **接続プーリング**　ConnectionPool 接続パラメーターは、クライアントの接続プールの動作を制御します。「ConnectionPool (CPOOL) 接続パラメーター」『SQL Anywhere サーバー データベース管理』と「接続プーリング」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ODBC データソースの Escape 接続パラメーターのサポート**　デフォルトでは、ODBC ドライバーでチルダ (~) がエスケープ文字として使用されますが、一部のアプリケーションではエスケープ文字に円記号 (¥) が使用されることもあります。Escape 接続パラメーターを使用して、アプリケーションのエスケープ文字を指定できます。Escape 接続パラメーター (ODBC)『SQL Anywhere サーバー データベース管理』を参照してください。

  また、**[接続]** ウィンドウで Escape 接続パラメーターを指定することもできます。

バックアップとリカバリ

次に、SQL Anywhere バージョン 12.0.0 でのバックアップとリカバリの強化を示します。

- **バージョン 12 で作成されたアーカイブバックアップのリストア**　バージョン 12 では、バージョン 11 以前のデータベースサーバーで作成したアーカイブバックアップをリストアできません。

- **アーカイブバックアップでの空きページの除去**　アーカイブバックアップでは、デフォルトで空きページがスキップされるため、バックアップが小規模で高速になる可能性があります。トランザクションログファイルには空きページが含まれないため、空きページを除去してもトランザクションログファイルのバックアップには影響がありません。したがって、大規模なトランザクションログファイルがあるデータベースで空きページを除去しても、トランザクションログファイルの規模が小さいデータベースほど利点はありません。BACKUP 文の FREE PAGE ELIMINATION 句を使用するか、**データベースバックアップウィザード**を使用して、この動作を制御できます。「BACKUP 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  空きページの除去をオンにしてバックアップされた、強力に暗号化されているデータベースをリストアする場合は、データベースの暗号化キーを指定してください。RESTORE DATABASE 文の KEY 句を使用するか、**データベースリストアウィザード**を使用して、暗号化キーを指定できます。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

セキュリティ

次に、SQL Anywhere バージョン 12.0.0 でのセキュリティの強化を示します。

- **FIPS 認定のアルゴリズムが 64 ビット Windows と 32 ビットおよび 64 ビットの Linux オペレーティングシステムで新規利用可能**　64 ビット Windows と 32 ビットおよび 64 ビットの Linux オペレーティングシステムで FIPS 認定のアルゴリズムを使用できるようになりました。

  FIPS 認定のアルゴリズムをサポートするプラットフォームのリストについては、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

  FIPS 認定のアルゴリズムの使用方法の詳細については、「FIPS 認定の暗号化テクノロジ」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースのパーミッションと権限

次に、SQL Anywhere の新しい、または強化されたパーミッションと権限を示します。これらの変更点を使用するには、データベースをアップグレードまたは再構築する必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **SYS_SPATIAL_ADMIN_ROLE グループ**　このグループのメンバーシップが与えられたユーザーは、空間参照系と計測単位を作成、変更、または削除できます。「空間パーミッションの付与」『SQL Anywhere サーバー 空間データサポート』を参照してください。

データベースユーティリティ

次に、SQL Anywhere バージョン 12.0.0 でのデータベースユーティリティの強化を示します。

- **データベースのページサイズをキロバイト単位で指定できる**　抽出ユーティリティ (dbxtract)、初期化ユーティリティ (dbinit)、アンロードユーティリティ (dbunload) を使用して、データベースのページサイズをバイトまたはキロバイト単位で指定できるようになりました。以前のバージョンでは、ページサイズをバイト単位でのみ指定できました。次の項を参照してください。

  ○ 「抽出ユーティリティ (dbxtract)」『SQL Remote』
  ○ 「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』
  ○ 「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』

- **ファイル非表示ユーティリティ (dbfhide) の強化**　Windows では、ファイル非表示ユーティリティ (dbfhide) で、Windows 暗号化 API を使用した設定ファイルの難読化がサポートされるようになりました。このユーティリティでは、クワイエットモードで実行できるように -q オプションもサポートされています。「ファイル非表示ユーティリティ (dbfhide)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サーバーライセンス取得ユーティリティ (dblic) の強化**　以前のリリースでは、SQL Anywhere のエディションを変更する場合、ソフトウェアをアンインストールしてから、新しいライセンスキーを使用して再インストールする必要がありました。このリリースでは、Sybase

iAnywhere から新しいライセンスキーを受け取ったときに、サーバーライセンス取得ユーティリティ (dblic) の -k オプションを使用して、SQL Anywhere のエディションを変更できるようになりました。「サーバーライセンス取得ユーティリティ (dblic)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サービスユーティリティ (dbsvc) の強化** Windows で dbsvc ユーティリティの -rs オプションを使用してサービスの依存性を指定する場合、表示名または実際のサービス名を指定できるようになりました。「Windows 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サポートユーティリティ (dbsupport) の強化** -ac オプションを使用すると、エラーレポートに含められるコメントを追加できます。-af オプションを使用すると、エラーレポート送信にファイルを含めることができます。「サポートユーティリティ (dbsupport)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **アンロードユーティリティ (dbunload) の強化** dbunload ユーティリティで次のオプションがサポートされるようになりました。

  - -kd オプションを指定すると、データベースが 1 つの DB 領域にアンロードされます。「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。
  - -qr オプションを指定すると、テーブルのロードやインデックスの作成が実行されるときに、進行メッセージの作成と表示が行われなくなります。「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースオプション

次に、SQL Anywhere バージョン 12.0.0 でのデータベースオプションの強化を示します。

- **uuid_has_hypens オプション** uuid_has_hyphens では、ユニークな識別子の値を文字列に変換するときのフォーマットを設定します。この機能はバージョン 11 で削除されましたが、また使用できるようになりました。「uuid_has_hyphens オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **blocking_others_timeout オプション** このオプションでは、現在の接続がロールバックされる前に、現在の接続のローとテーブルのロックを他の接続からブロックできる時間を指定します。このオプションを使用すると、優先度の低いタスクが指定時間を超えて他の接続をブロックすることを防止できます。「blocking_others_timeout オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **http_connection_pool_basesize オプション** このオプションでは、HTTP によって使用される接続プールのベースラインサイズを指定します。「http_connection_pool_basesize オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **http_connection_pool_timeout オプション** このオプションでは、未使用の接続を HTTP 接続プールに保持できる時間の上限を指定します。「http_connection_pool_timeout オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **progress_messages オプション** このオプションは、データベースサーバーからクライアントに進行メッセージを送信するかどうかを制御します。ユーティリティデータベースのテン

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ポラリオプションとしてこのオプションを設定できます。「progress_messages オプション」『SQL Anywhere サーバー データベース管理』と「ユーティリティデータベースに使用できる文」『SQL Anywhere サーバー データベース管理』を参照してください。

Interactive SQL では、デフォルトで進行メッセージが **[メッセージ]** ウィンドウ枠に表示されます。また、**[ツール]** » **[オプション]** » **[SQL Anywhere]** » **[コマンド]** をクリックし、**[進行メッセージを表示]** をクリックして、SQL Anywhere データベースオプションの progress_messages を設定することもできます。このオプションを選択すると、データベースの progress_messages オプションは Formatted に設定されます。クリアすると、データベースの progress_messages オプションは Off に設定されます。デフォルトでは、**[進行メッセージを表示]** オプションは選択されています。

- **timestamp_with_time_zone_format オプション**　このオプションは、TIMESTAMP WITH TIME ZONE 値が文字列に変換される方法を制御します。「timestamp_with_time_zone_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **reserved_keywords オプション**　このオプションは、個々のキーワードをオンにします。「reserved_keywords オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **st_geometry_asbinary_format オプション**　ジオメトリからバイナリへの空間データ値の変換方法を制御します。「st_geometry_asbinary_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **st_geometry_astext_format オプション**　ジオメトリからテキストへの空間データ値の変換方法を制御します。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **st_geometry_asxml_format オプション**　ジオメトリから XML への空間データ値の変換方法を制御します。「st_geometry_asxml_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **st_geometry_describe_type オプション**　空間データの記述方法を制御します。「st_geometry_describe_type オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **st_geometry_on_invalid オプション**　ジオメトリが基本的な検証に失敗した場合の動作を制御します。「st_geometry_on_invalid オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースサーバーオプション

次に、SQL Anywhere バージョン 12.0.0 でのデータベースサーバーオプションの強化を示します。

- **-xm オプション**　-xm データベースサーバーオプションでは、データベースサーバーが新しい IP アドレスをチェックする頻度を制御します。コンピューターが新しいネットワークに接続しているか、既存のネットワークから切断しているときに -xm オプションが指定された場合、変更が検出されると、データベースサーバーは新しいネットワークで受信を開始しま

す。「-xm dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

プロパティとパフォーマンスモニターの統計値

次に、SQL Anywhere バージョン 12.0.0 でのプロパティとパフォーマンスモニターの統計の強化を示します。

- **接続プロパティ**　このリリースには、次の接続プロパティが追加されています。
  - blocking_others_timeout
  - http_connection_pool_basesize
  - http_connection_pool_timeout
  - ParentConnection
  - Progress
  - progress_messages
  - QueryRowsFetched
  - reserved_keywords
  - st_geometry_asbinary_format
  - st_geometry_astext_format
  - st_geometry_asxml_format
  - st_geometry_describe_type
  - st_geometry_on_invalid
  - timestamp_with_time_zone_format

  これらのプロパティの詳細については、「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースプロパティ**　このリリースには、次のデータベースプロパティが追加されています。
  - ConnPoolCachedCount
  - ConnPoolHits
  - ConnPoolMisses
  - DriveBus
  - DriveModel
  - HasTornWriteFix
  - HttpConnPoolCachedCount
  - HttpConnPoolHits
  - HttpConnPoolMisses
  - HttpConnPoolSteals
  - LastCheckpointTime
  - MirrorRole
  - QueryRowsFetched
  - SynchronizationSchemaChangeActive
  - WriteChecksums

  これらのプロパティの詳細については、「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **データベースサーバープロパティ**　このリリースには、次のデータベースサーバープロパティが追加されています。

  - AutoMultiProgrammingLevel
  - AutoMultiProgrammingLevelStatistics
  - CurrentMultiProgrammingLevel
  - IPAddressMonitorPeriod
  - IsPortableDevice
  - MaxMultiProgrammingLevel
  - MinMultiProgrammingLevel
  - ObjectType
  - ThreadDeadlocksAvoided
  - ThreadDeadlocksReported

  これらのプロパティの詳細については、「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

システムプロシージャーとファンクション

次に、SQL Anywhere バージョン 12.0.0 で追加されたシステムプロシージャーとファンクションの強化を示します。

- **新しい sa_text_index_vocab_nchar システムプロシージャー**　この新しいシステムプロシージャーは、NCHAR テキストインデックスで使用され、sa_text_index_vocab と同じです。「sa_text_index_vocab_nchar システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_copy_cursor_to_temp_table システムプロシージャー**　カーソルの内容をテンポラリテーブルにコピーします。「sa_copy_cursor_to_temp_table システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_describe_cursor システムプロシージャー**　カーソル内のカラムの名前とデータ型を返します。この情報は、さまざまなクライアントプログラミングインターフェイスから取得できますが、以前はストアドプロシージャー内でアクセスできませんでした。「sa_describe_cursor システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_install_feature システムプロシージャー**　SQL Anywhere のインストール時にデータベースに存在していなかった追加の機能をインストールします。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_list_cursors システムプロシージャー**　現在の接続についてデータベースサーバーが保持している各カーソルの 1 行を含む結果セットを返します。結果セットは、カーソル名、カーソルが現在開いているかどうかを示す値、その他のメタ情報を示します。「sa_list_cursors システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_mirror_server_status システムプロシージャー**　このプロシージャーは、プロシージャーが実行されているデータベースサーバーの下にあるツリー内のコピーノードのステー

タスをレポートします。「sa_mirror_server_status システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_reserved_words システムプロシージャー** このプロシージャーは、SQL Anywhere の予約語のリストを返します。「sa_reserved_words システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_server_option システムプロシージャーの強化** sa_server_option システムプロシージャーに次のオプションが追加されました。

  - AutoMultiProgrammingLevel
  - AutoMultiProgrammingLevelStatistics
  - CurrentMultiProgrammingLevel
  - DropBadStatistics
  - DropUnusedStatistics
  - IPAddressMonitorPeriod
  - MaxMultiProgrammingLevel
  - MinMultiProgrammingLevel
  - StatisticsCleaner

  「sa_server_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_table_page_usage システムプロシージャーの強化** progress_messages データベースオプションが Raw または Formatted に設定されている場合、このシステムプロシージャーの実行中にデータベースサーバーからクライアントに進行状況メッセージが送信されます。「sa_table_page_usage システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **xp_read_file システムプロシージャーの強化** xp_read_file システムプロシージャーに、遅延読み込みを指定するための任意のパラメーターが含まれるようになりました。この任意のパラメーターを指定し、その値が 0 以外の場合、ファイルは読み込まれ、直ちにロックが解除されます。「xp_read_file システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **xp_startsmtp システムプロシージャーの強化** xp_startsmtp システムプロシージャーで、4 つの新しいパラメーター trusted_certificates、certificate_company、certificate_unit、certificate_name がサポートされます。これらのパラメーターにより、セキュア SMTP を使用してメールを送信できます。この機能を使用するには、データベースをアップグレードする必要があります。「xp_startsmtp システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **openxml システムプロシージャーの強化** openxml システムプロシージャーで、USING FILE 句と USING VALUE 句がサポートされるようになりました。これらの句を使用すると、CHAR、NCHAR、BINARY、または LONG BINARY 型のファイルや式、または BLOB 文字列からデータをロードできます。「openxml システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい MEDIAN 関数** ローのセットの数値式の中央値を計算します。「MEDIAN 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **HASH 関数の強化**　HASH 関数で CRC32 アルゴリズム型を使用できるようになりました。「HASH 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **BIT_OR、BIT_AND、BIT_XOR 関数の強化**　BIT_OR、BIT_AND、BIT_XOR 関数で、整数値とバイナリ値がサポートされるようになりました。また、並列実行プランで BIT_OR、BIT_AND、BIT_XOR 関数を使用できるようになりました。

  次の項を参照してください。

  ○「BIT_OR 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』
  ○「BIT_AND 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』
  ○「BIT_XOR 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』

- **DATEADD、DATEDIFF、DATEPART、DATENAME 関数の強化**　DATEADD、DATEDIFF、DATEPART、DATENAME 関数で、マイクロ秒日付単位と TIMESTAMP WITH TIME ZONE データ型がサポートされるようになりました。次の項を参照してください。

  ○「DATEADD 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』
  ○「DATEDIFF 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』
  ○「DATEPART 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』
  ○「DATENAME 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』

- **DB_EXTENDED_PROPERTY 関数の強化**　DB_EXTENDED_PROPERTY 関数を MirrorServerState プロパティおよび MirrorState プロパティとともに使用して、ミラーサーバーの同期ステータスと接続ステータスを確認できるようになりました。「DB_EXTENDED_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい HTTP_RESPONSE_HEADER 関数**　HTTP 応答ヘッダーの値を返します。「HTTP_RESPONSE_HEADER 関数 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **HTTP_VARIABLE 関数の強化**　@BINARY 属性を使用して、x-www-form-urlencoded バイナリデータ値を返せるようになりました。「HTTP_VARIABLE 関数 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい ISENCRYPTED 関数**　文字列が暗号化されているかどうかを確認します。「ISENCRYPTED 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい NEXT_HTTP_RESPONSE_HEADER 関数**　次の HTTP 応答ヘッダーの名前を取得します。「NEXT_HTTP_RESPONSE_HEADER 関数 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい SWITCHOFFSET 関数**　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。「SWITCHOFFSET 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい SYSDATETIMEOFFSET 関数**　データベースサーバーの現在の日付、時刻、タイムゾーンオフセットを返します。「SYSDATETIMEOFFSET 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい TODATETIMEOFFSET 関数**　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。「TODATETIMEOFFSET 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい COUNT_BIG 関数**　指定されたパラメーターに従って、グループのロー数をカウントします。「COUNT_BIG 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

SQL 文

次に、SQL Anywhere バージョン 12.0.0 での SQL の強化を示します。

- **名前付きインデックスヒントの修飾の改善**　1 つのテーブルの PRIMARY KEY インデックスと FOREIGN KEY インデックスは名前が同じである場合があります。このことが当てはまる場合は、名前付きインデックスヒントを明確に指定できません。名前付きインデックスヒントの指定が拡張され、ヒントが指定されたインデックスを PRIMARY KEY インデックスまたは FOREIGN KEY インデックスとして修飾できるようになりました。INDEX 句、FROM 句『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい IS DISTINCT FROM および IS NOT DISTINCT FROM 探索条件**　IS DISTINCT FROM および IS NOT DISTINCT FROM 探索条件を使用すると、NULL の場合の等価性の評価を制御できます。「IS DISTINCT FROM および IS NOT DISTINCT FROM 探索条件」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE SYNCHRONIZATION PROFILE 文**　この文は、SQL Anywhere の同期プロファイルを作成します。同期プロファイルによって、SQL Anywhere データベースが Mobile Link サーバーと同期する方法を定義します。「CREATE SYNCHRONIZATION PROFILE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい CREATE INDEX 文の WITH NULL NOT DISTINCT 句**　CREATE INDEX 文に、UNIQUE インデックスの作成時に使用する新しい句 WITH NULLS NOT DISTINCT が追加されました。この句を使用すると、インデックスキー内の NULL がユニークでないことを指定できます。この機能を使用するには、既存のデータベースをアップグレードまたは再構築する必要があります。「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』の UNIQUE 句の説明を参照してください。

- **逆引用符識別子デリミター**　逆引用符を識別子デリミターとして使用できるようになりました。「識別子」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER TEXT CONFIGURATION 文の新しい SAVE OPTION VALUES**　この新しい句を使用して、テキスト設定オブジェクトに保存された date_format、time_format、timestamp_format、timestamp_with_time_zone_format オプションの値を変更します。「ALTER TEXT CONFIGURATION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER SERVER 文と CREATE SERVER 文**　新しい IQODBC サーバークラスと IQJDBC サーバークラスを使用すると、Sybase IQ サーバーをリモート接続として指定できます。「ALTER SERVER 文」『SQL Anywhere サーバー SQL リファレンス』と「CREATE SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい LOCK FEATURE 文**　この文は、他の同時接続でデータベースサーバーの機能が使用されないようにします。「LOCK FEATURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **BEGIN、CREATE VARIABLE、DECLARE 文の強化**　変数宣言に変数の初期開始値を含めることができるようになりました。次の項を参照してください。

  - 「BEGIN 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE VARIABLE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DECLARE 文」『SQL Anywhere サーバー SQL リファレンス』

- **LOAD TABLE 文の強化**　LOAD TABLE 文で load-option 句がサポートされるようになりました。この句を使用すると、データのロード方法を制御できます。また、FORMAT 句の値 XML もサポートされます。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい IF NOT EXISTS 句**　新しい IF NOT EXISTS 句を指定し、指定したオブジェクトがすでに存在する場合、変更は行われず、エラーは返されません。次の項を参照してください。

  - 「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PUBLICATION 文 [Mobile Link] [SQL Remote]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE TEXT INDEX 文」『SQL Anywhere サーバー SQL リファレンス』

- **新しい IF EXISTS 句**　新しい IF EXISTS 句により、存在しないデータベースオブジェクトの削除を DROP 文で試みてもエラーを返さないように指定できます。次の項を参照してください。

  - 「DROP EVENT 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP FUNCTION 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP INDEX 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP PUBLICATION 文 [Mobile Link] [SQL Remote]」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP SPATIAL UNIT OF MEASURE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP SYNCHRONIZATION PROFILE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP TRIGGER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP VARIABLE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP VIEW 文」『SQL Anywhere サーバー SQL リファレンス』

- **CASE 文の強化**　Transact-SQL のプロシージャーとバッチで CASE 文がサポートされるようになりました。

- **新しい OR REPLACE 句**　新しい OR REPLACE 句を使用すると、プロファイルまたは変数を作成したり、同じ名前のオブジェクトが存在する場合は、置換したりできます。次の項を参照してください。
  - 「CREATE FUNCTION 文 [外部呼び出し]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE FUNCTION 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PROCEDURE 文 [外部呼び出し]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PROCEDURE 文 [T-SQL]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE SEQUENCE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE SYNCHRONIZATION PROFILE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE TRIGGER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE VARIABLE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE VIEW 文」『SQL Anywhere サーバー SQL リファレンス』
- **SELECT 文の新しい LIMIT 句のサポート**　SELECT 文で新しい LIMIT 句を使用してローカウントとオフセットを指定できるようになりました。「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **SET OPTION 文の強化**　SET OPTION 文の構文 1 で、変数の内容を使用したオプションの設定がサポートされるようになりました。「SET OPTION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **新しい IF [NOT] OF 探索条件**　IF [NOT] OF *type-expression* 探索条件が追加されました。「探索条件」『SQL Anywhere サーバー SQL リファレンス』と「空間データ型構文の理解」『SQL Anywhere サーバー 空間データサポート』を参照してください。
- **INSERT 文の強化**　INSERT 文には、次のような強化が行われています。「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **複数の値のリストのサポート**　INSERT 文に複数の値のリストを含めることが可能になったことによって、一度に複数のローを挿入できるようになりました。次に例を示します。

    ```
    INSERT INTO T (c1,c2,c3)
    VALUES (1,10,100), (2,20,200), (3,30,300);
    ```

  - **すべてのデフォルト値を含むローの挿入のサポート**　SQL Anywhere では、テーブル内のカラムのサブセットに対して指定された値を VALUES 句に含めることができます。指定されていないすべてのカラムには、CREATE TABLE 文の DEFAULT 句、NULL 句、COMPUTE 句によってカラムごとに指定されたデフォルト値が適用されます。以前のバージョンでは、データベースサーバーには、テーブル内の少なくとも 1 つのカラムの入力値を指定する必要がありました。

    今回のバージョンでは、次のいずれかの構文拡張によって、すべてのカラムにデフォルトが適用されるようになりました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

INSERT [INTO] *table-name options* **DEFAULT VALUES** ...

INSERT [INTO] *table-name* **( )** *options* **VALUES ( ) [, ( ) ...]**

DEFAULT VALUES または VALUES を指定することは、セマンティック上、次の構文を使用することと同じです。デフォルトエントリの数は、テーブル内のカラムの数と同じになります。

INSERT [INTO] *table-name* **VALUES(default, default, ..., default)**

DEFAULT VALUES 句は SQL/2008 標準の一部ですが、VALUES 句はベンダー拡張です。

- **新しい START SERVER 文**　START SERVER 文が追加されました。この文は、廃止される START ENGINE 文の代わりに使用してください。「START SERVER 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **新しい STOP SERVER 文**　STOP SERVER 文が追加されました。この文は、廃止される STOP ENGINE 文の代わりに使用してください。「STOP SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **ALTER DATABASE 文の新しい CHECKSUM 句**　CHECKSUM OFF 句を使用すると、データベースのグローバルチェックサムを無効にすることができます。新しいバージョン 12 のデータベースでは、デフォルトでグローバルチェックサムが有効になっています。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **DELETE 文の強化**　DELETE 文で相関名がサポートされるようになりました。「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

データ型

次に、SQL Anywhere バージョン 12.0.0 でのデータ型の強化を示します。

- **TIMESTAMP WITH TIME ZONE データ型**　タイムゾーンオフセット付きの時刻を格納します。「TIMESTAMP WITH TIME ZONE データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  エイリアス DATETIMEOFFSET もサポートされます。「DATETIMEOFFSET データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CHAR、NCHAR、NVARCHAR、VARCHAR データ型の 32767 文字長のサポート**　NCHAR および NVARCHAR データ型は、最大 32767 文字まで宣言できるようになりました。CHAR および VARCHAR データ型は、文字長セマンティックを使用して最大 32767 文字まで宣言できるようになりました。クライアントの文字セットで、NCHAR、NVARCHAR、CHAR、または VARCHAR データ型の、文字長セマンティックを使用して記述されたバイト長が 32767 を超える場合、これらのデータ型は、LONG NVARCHAR または LONG VARCHAR データ型として記述されます。CHAR および VARCHAR データ型のバイト長が、バイト長セマンティックを使用して記述されている場合は、値をクライアントの文字セットに変換するときに、最大可能文字セット拡張が行われます。最大拡張を含む結果長が 32767 を超える場合、そのデータ型は LONG VARCHAR として記述されます。クライアントでシングルバイト文字

セットが使用されているか、クライアントの文字セットとデータベースの文字セットのエンコードが同じ場合、拡張は行われません。次の項を参照してください。

○「CHAR データ型」『SQL Anywhere サーバー SQL リファレンス』
○「NCHAR データ型」『SQL Anywhere サーバー SQL リファレンス』
○「NVARCHAR データ型」『SQL Anywhere サーバー SQL リファレンス』
○「VARCHAR データ型」『SQL Anywhere サーバー SQL リファレンス』

- **空間データのデータ型**　空間データをサポートするために、複数の新しいデータ型が追加されました。「空間データへのアクセスとそのデータの操作」『SQL Anywhere サーバー 空間データサポート』を参照してください。

プログラミングインターフェイス

次に、SQL Anywhere バージョン 12.0.0 でのプログラミングインターフェイスの強化を示します。

- **Web サービスパフォーマンスの強化**　プランのキャッシュの効果と潜在的なパフォーマンスの向上による利点を最大化するために、HTTP サービスで自動接続プールがサポートされるようになりました。「HTTP Web サーバーでの Web サービスアプリケーションの開発」『SQL Anywhere サーバー プログラミング』を参照してください。

- **カスタマイズされた ODBC ドライバー名のサポート**　SQL Anywhere ODBC ドライバーの複数の独立したコピーをクライアントシステムに容易にインストールおよび登録するため、ODBC ドライバーにカスタマイズされた名前を割り当てることができるようになりました。「ODBC ドライバーの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

- **UNIX の ANSI 専用 ODBC ドライバー**　SQLWCHAR を 32 ビット (UTF-32) 数として定義するバージョンの UNIX ODBC ドライバーマネージャーは、ワイド呼び出しをサポートする SQL Anywhere ODBC ドライバーに使用できません。これは、このドライバーは 16 ビット SQLWCHAR 用に作成されているためです。このような場合に備えて ANSI 専用バージョンの SQL Anywhere ODBC ドライバーが用意されています。このバージョンの ODBC ドライバーでは、ワイド呼び出しインターフェイス (SQLConnectW など) はサポートされていません。「UNIX 用 UTF-32ODBC ドライバーマネージャー」『SQL Anywhere サーバー プログラミング』を参照してください。

- **DBTools: no_reload_status ビットフィールドの an_unload_db 構造への追加**　an_unload_db 構造に新しいビットフィールド no_reload_status が追加されました。no_reload_status を使用して、テーブルとインデックスの進行メッセージの再ロードを抑制できます。an_unload_db 構造体 [データベースツール]『SQL Anywhere サーバー プログラミング』を参照してください。

- **新しい SQL Anywhere TYPE-2 JDBC ドライバー**　SQL Anywhere への接続時に、新しい TYPE-2 JDBC ドライバーを JDBC アプリケーションで使用できるようになりました。ODBC の最上部に位置し、ODBC 経由でさまざまなサーバーに接続するために使用できる iAnywhere JDBC ドライバーとは異なり、SQL Anywhere JDBC ドライバーは SQL Anywhere のみに接続し、SQL Anywhere ODBC ドライバーのインストールや登録を必要としません。

  SQL Anywhere JDBC ドライバーは、JDBC 3.0 ドライバーと JDBC 4.0 ドライバーに付属しています。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

注意
現在 iAnywhere JDBC ドライバーを使用している場合は、新しい SQL Anywhere JDBC ドライバーに変更することを強くおすすめします。

JDBC 4.0 ドライバーは自動的にロードおよび登録されます。

バージョン 3.0 の SQL Anywhere JDBC ドライバーを使用するには、java.sql.Driver インターフェイスを実装する sybase.jdbc.sqlanywhere.IDriver クラスをロードし、JDBC ドライバーマネージャーを使用して SQL Anywhere JDBC ドライバーを登録する必要があります。ロードすると、URL jdbc:sqlanywhere:*connection-string-parameters* を使用して、SQL Anywhere JDBC ドライバーを使用する接続を確立できます。*connection-string-parameters* は、SQL Anywhere に接続するときに必要な標準の接続パラメーターです。SQL Anywhere JDBC ドライバーを使用する場合、アプリケーションで *connection-string-parameters* に DRIVER= または DSN= を指定する必要はなくなりました。「JDBC サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **JDBC statement クラスメソッドの新しいサポート**　以前のバージョンでは、PreparedStatement クラスの addBatch メソッドと executeBatch メソッドのみが JDBC ドライバーによってサポートされていました。今回のバージョンでは、Statement クラスの addBatch メソッド、clearBatch メソッド、および executeBatch メソッドも、JDBC ドライバーによってサポートされるようになりました。Statement クラスの executeBatch メソッドの動作に関する JDBC 仕様は明確でないため、このメソッドを SQL Anywhere JDBC ドライバーで使用する場合は、次のことに注意する必要があります。

  1. SQL 例外または結果セットが発生すると、バッチの処理はすぐに停止します。バッチの処理が停止すると、executeBatch メソッドによって BatchUpdateException がスローされます。BatchUpdateException で getUpdateCounts メソッドが呼び出されると、ローカウントの整数配列が返されます。この配列では、バッチエラーが発生する前のカウントセットには有効な負でない更新カウントが含まれますが、バッチエラーが発生した時点以降のすべてのカウントには -1 の値が含まれます。BatchUpdateCount を SQLException にキャストすると、バッチ処理が停止した理由に関する追加の詳細が提供されます。

  2. バッチは、clearBatch メソッドが明示的に呼び出された場合にのみクリアされます。その結果、executeBatch メソッドを繰り返し呼び出すと、バッチが繰り返し再実行されます。また、execute(sql_query) または executeQuery(sql_query) を正しく呼び出すと、指定された SQL クエリが実行されますが、基本となるバッチはクリアされません。したがって、executeBatch メソッド、execute(sql_query)、executeBatch メソッドの順に呼び出すと、バッチ文のセットが実行され、次に、指定された SQL クエリが実行され、次に、再びバッチ文のセットが実行されます。

  「JDBC サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **RESUME 文はローカウントまたはロー推定値を返す**　RESUME 文は、プロシージャーコールの結果セット内のローの数、または次の結果セットのロー推定値を返すようになりました。以前のバージョンでは、このローカウントは RESUME 文の後に使用できませんでした。RESUME ローカウントを取得するには、クライアントアプリケーションとデバイスサーバーの両方が SQL Anywhere 12 である必要があります。

  この変更は、次のクライアント API に影響を及ぼします。

API	影響を受ける関数呼び出しまたは文	ローカウントを返す
Embedded SQL	RESUME 文	SQLCOUNT フィールド
ODBC	SQLMoreResults 関数	SQLRowCount 関数
OLE DB	IMultipleResults::GetResult メソッド	pcRowsAffected 出力パラメーター
PHP	sasql_next_result 関数	sasql_num_rows 関数
PHP	sasql_stmt_next_result 関数	sasql_stmt_num_rows 関数
SQL Anywhere C API	sqlany_get_next_result 関数	sqlany_num_rows 関数
SQL Anywhere Ruby API	sqlany_get_next_result 関数	sqlany_num_rows 関数
SQL Anywhere Python API	nextset メソッド	rowcount 属性

- **CGI/1.1 に必要なすべての変数が PHP $_SERVER 変数に含まれるようになった**　HTTP 要求を実行するときに CGI/1.1 に必要なすべての変数が PHP の $_SERVER 変数に含まれるようになりました。「PHP 外部環境」『SQL Anywhere サーバー プログラミング』を参照してください。

  CGI/1.1 に必要な変数の詳細については、http://www.ietf.org/rfc/rfc3875 を参照してください (RFC3875 の Section 4)。

- **sqlanydb への Python データ型マッピングに対する詳細な制御の追加**　変換コールバックを登録すると、データベースサーバーから結果がフェッチされるときに、Python オブジェクトへデータベースの型がマッピングされる方法を制御できます。「データベースタイプの変換」『SQL Anywhere サーバー プログラミング』を参照してください。

- **新しい DBLib fill_sqlda_ex 関数**　fill_sqlda_ex 関数には、fill_sqlda に対する追加機能があります。特に、記述子を埋めるときに、DT_LONGVARCHAR、DT_LONGNVARCHAR、DT_LONGBINARY 型を保持するためにフラグを提供できます。「fill_sqlda_ex 関数」『SQL Anywhere サーバー プログラミング』を参照してください。

- **サイレントインストール機能を選択できるようになった**　Windows では、コマンドラインから SQL Anywhere のコンポーネントや機能をインストール用に選択または省略できます。たとえば、32 ビットの管理ツールを選択するには、AT32=1 を指定します。「SQL Anywhere インストーラーを使用したサイレントインストール」『SQL Anywhere サーバー プログラミング』を参照してください。

- **TDS による時刻および日時データの 6 桁精度のサポート**　jConnect 7 以降または Open Client 15.5 以降を使用して SQL Anywhere に接続するアプリケーションでは、時刻および日時データを問い合わせるときに 6 桁の精度を取得できるようになりました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

カタログの変更

次に、SQL Anywhere バージョン 12.0.0 でのカタログの変更を示します。

- **新しい ISYSSEQUENCE システムテーブルと SYSSEQUENCE システムビュー**　ISYSSEQUENCE システムテーブルには、各ユーザー定義シーケンスごとに 1 つのローがあります。「SYSSEQUENCE システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい ISYSSEQUENCEPERM システムテーブルと SYSSEQUENCEPERM システムビュー**　ISYSSEQUENCEPERM システムテーブルには、ユーザーまたはグループがシーケンスに対して保持する権限が記録されます。「SYSSEQUENCEPERM システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ISYSTEXTCONFIG システムテーブル**　ISYSTEXTCONFIG システムテーブルは、トークン化か事前フィルタリングまたはその両方に使用されるエントリポイントと外部ライブラリに関する情報を格納するように変更されました。具体的には、prefilter カラムのデータ型が LONG VARCHAR になり、外部事前フィルターライブラリのエントリポイントとライブラリ名を格納するように変更されました。外部単語区切りライブラリのエントリポイントとライブラリ名を格納するために、新しい LONG VARCHAR カラムの external_term_breaker が追加されました。「SYSTEXTCONFIG システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ISYSTABCOL システムテーブル：新しい base_type_str カラム**　このカラムには、カラムの物理的な型を表す注釈付きの型文字列が格納されます。

- **ISYSPROCPARM システムテーブル：新しい base_type_str カラム**　このカラムには、パラメーターの物理的な型を表す注釈付きの型文字列が格納されます。

- **ISYSUSERTYPE システムテーブル:新しい base_type_str カラム**　このカラムには、ユーザータイプの物理的な型を表す注釈付きの型文字列が格納されます。

- **ISYSOBJECT システムテーブル：新しい object_type_str カラム**　このカラムには、SYSOBJECT.object_type 内の値の単語説明 (MAT VIEW など) が格納されます。

  以前のバージョンでは、SYSOBJECT ビューには TINYINT として表されるオブジェクトタイプを示す object_type のみがありました。つまり、整数を単語説明に変換するには、マニュアルまたはビュー定義のいずれかにアクセスする必要がありました。このバージョンでは、object_type_str カラムを問い合わせて、オブジェクトの単語説明を確認できます。

Windows Mobile の強化

次に、SQL Anywhere バージョン 12.0.0 での Windows Mobile に関する強化を示します。

- **Windows Mobile の内部実行スレッドのデフォルトスタックサイズの変更**　Windows Mobile のデフォルトのスタックサイズは 96 KB、最小スタックサイズは 64 KB、最大スタックサイズは 512 KB になりました。「-gss dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

UNIX/Linux の強化

次に、SQL Anywhere バージョン 12.0.0 での UNIX と Linux に関する強化を示します。

- **パフォーマンス統計ユーティリティ (dbstats)** dbstats ユーティリティは、UNIX コンピューターに関するパフォーマンス統計の値を返します。このユーティリティの動作は、Windows パフォーマンスモニターの動作と似ています。「パフォーマンス統計値ユーティリティ (dbstats) (UNIX)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Linux サービス用に作成される PID ファイル** SQL Anywhere サービスが Linux で実行されている場合、PID ファイルは */var/run* ディレクトリで作成されます。「Linux 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

パフォーマンスの強化

次に、パフォーマンスの向上以外にユーザーに見える変更がないバージョン 12.0.0 でのパフォーマンスの強化を示します。

- **プライマリローの更新時のロックの向上** プライマリローの非キーカラムの更新と、そのローを参照する外部ローの変更は、相互に干渉しなくなりました。たとえば、サンプルデータベースで、一方の処理が他方を待機する必要なしに、対応する顧客アドレスの更新中に sales_order ローを追加できます。「sa_locks システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』と「ロックできるオブジェクト」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **独立性レベル 2 および 3 でのロックの減少** テーブルが共有モード (LOCK TABLE...IN SHARE MODE 文) でロックされている場合、個々のローに対する読み込みロックは取得されなくなりました。これにより、独立性レベル 2 および 3 でのロックのオーバーヘッドが減少する可能性があります。

- **インデックスのパフォーマンス向上** SQL Anywhere 12 では、多数のクラスター化された連続値をインデックスから削除する場合のパフォーマンスを改善するために、アルゴリズムが強化され、新しいディスク上レイアウトが用意されました。

- **検証パフォーマンスの改善** SQL Anywhere 12 では、大規模なデータベースの検証を改善するために、大幅な強化が加えられました。

- **要求の優先順位付けの向上** SQL Anywhere 12 は、I/O バウンド要求の優先順位を上げるように強化されました。これにより、ハードウェアリソースの使用率が向上します。

- **リモートデータアクセスパフォーマンスの向上** SQL Anywhere 12 では、プロキシテーブルのパフォーマンスの向上を含め、リモートデータアクセスパフォーマンスを向上させるための大幅な強化が行われています。

- **新しいコストモデル** SQL Anywhere 12 には、最新のハードウェアのクエリ実行コストをより正確に推定する CPU コストモデルが用意されています。この動作により、一部のアクセスプランが変更される可能性があります。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **ユーザー定義関数に埋め込まれたクエリの強化**　ユーザー定義関数に埋め込まれた SQL クエリをクエリオプティマイザーによってインライン化できるようになりました。これにより、呼び出しごとのプロシージャーを介した切り替えが回避され、オプティマイザーが文を最適化するときの自由度が大きくなります。

- **式の異なるデータ型への変換の改善**　式を異なるデータ型に変換するときにデータベースサーバーが使用する評価ルールが改善されました。新しい評価ルールにより、変換がより効率的に実行されるようになります。

その他

次に、SQL Anywhere バージョン 12.0.0 でのその他の強化を示します。

- **サンプルデータベースのコピーを作成するための新しいスクリプト**　SQL Anywhere インストールディレクトリの *bin32* または *bin64* ディレクトリに *newdemo.bat* ファイルと *newdemo.sh* ファイルが配置されました。これらのファイルを使用して、サンプルデータベースのすべてのデータを含むサンプルデータベースのコピーを作成できます。このスクリプトを使用すると、サンプルデータベースを再作成したり、異なる名前でサンプルデータベースの新しいコピーを作成したりできます。「サンプルデータベースの再作成 (demo.db)」『SQL Anywhere 12 紹介』を参照してください。

- **クエリオプティマイザーの新しい選択性推定ソースタイプ**　新しい選択性推定ソースタイプ JOIN が追加されました。この新しいソースタイプは、クエリオプティマイザーによって、フォーム T.X = R.X のアトミックな述部の選択性推定に使用されます。「ESTIMATE_SOURCE 関数 [その他]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **オーバーフローエラーの処理の改善**　算術演算 (+, -, *, /, SUM, AVG) は、演算の結果をそのデータ型で表現できないことによってオーバーフローする場合があります。以前のバージョンでは、データ型が INT の式でこのオーバーフローが発生した場合にエラーが返されましたが、その他のすべてのデータ型ではオーバーフローが発生した場合に未定義の値が返されました。今回のバージョンでは、すべてのデータ型に対するすべての算術計算でオーバーフローが検出され、結果をそのデータ型で表現できない場合に、エラーが返されます。

- **ALTER EXTERNAL ENVIRONMENT 文を使用した dbmlsync のロケーションの設定**　同期中に、データベースサーバーが使用している PATH 環境変数を使用して dbmlsync 実行プログラムを検出できない場合、ALTER EXTERNAL ENVIRONMENT 文を使用してデータベースサーバーに dbmlsync 実行プログラムのロケーションを通知できます。「ALTER EXTERNAL ENVIRONMENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **逆引用符のデリミターとしてのサポート**　逆引用符 (`) を使用して、SQL Anywhere の識別子を区切ることができるようになりました。「識別子」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **日本語の Unicode 照合アルゴリズム (UCA) 照合の適合化オプション**　新しい日本語の UCA 照合の適合化オプションを使用できるようになりました。このオプションを使用して、すべてのひらがなとカタカナの文字間におけるレベル 1 の違いを定義できます。この新しい適合化オプションを使用すると、大文字と小文字が区別されない照合において、ひらがなとカタ

カナの文字を正確に等号比較できます。「照合の適合化オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サーバーメッセージウィンドウと Windows システムトレイアイコンの変更** データベースサーバーメッセージウィンドウのタイトルバーで、パーソナルサーバーを実行しているか、ネットワークサーバーを実行しているかを表示できるようになりました。Windows システムトレイアイコンのツールチップにもデータベースサーバーのタイプが表示されます。また、データベースサーバーの **[バージョン情報]** ウィンドウには、実行中の SQL Anywhere のエディションが表示されます。

- **情報ユーティリティ (dbinfo) の強化** dbinfo ユーティリティで、データベースの CHAR 照合指定、CHAR エンコード、NCHAR 照合指定、または NCHAR エンコードに関する情報が返されるようになりました。「情報ユーティリティ (dbinfo)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **キャッシュ以外の使用に確保されるアドレス領域のサイズの制御** -ch オプションによって、より多くのアドレス領域がキャッシュ以外の用途に残されるようになり、32 ビットオペレーティングシステムの最大非 AWE キャッシュサイズは縮小されました。「-ch dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』と「-chx dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **CPU バウンド要求と比較される I/O バウンドの優先度管理の強化** ディスクスループットおよびハードウェアリソースの使用を改善するため、データベースサーバーによって I/O バウンドの要求が動的に検出され、CPU バウンドタスクに対する優先度が上げられるようになりました。

- **Windows での停電に対する堅牢性の向上** SQL Anywhere を配備するとき、Windows レジストリエントリを設定して、特定の Intel ストレージドライバーを使用するシステムの停電に対する堅牢性を高めることができます。このパラメーターを設定しないと、停電発生時にデータが失われたり、データベースが破損したりする可能性があります。配備の一部としてこれらのエントリが必要かどうかを確認するには、「Windows レジストリエントリ」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL FLAGGER の強化** SQL FLAGGER で SQL/2008 標準がサポートされるようになりました。「SQL Flagger を使用した SQL 準拠のテスト」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **進行メッセージ** 一部の SQL 文で、データベースサーバーからクライアントへの進行メッセージの送信がサポートされるようになりました。新しい progress_messages オプション 56 ページを参照してください。

SQL Anywhere の動作の変更

次に、バージョン 12.0.0 で導入された SQL Anywhere の動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しいデータベースでチェックサムがデフォルトで有効**　新しいデータベースを作成すると、グローバルチェックサムはデフォルトで有効になっています。グローバルチェックサムは、データベースページの読み込みまたは書き込みが行われるたびに計算、検証され、データベースページがディスク上で変更されているかどうかを確認するために使用されます。初期化ユーティリティの -s[+ | -] オプションを使用するか、CREATE DATABASE 文の CHECKSUM 句を使用して、データベースでグローバルチェックサムを有効にするかどうかを制御できます。次の項を参照してください。

  ○ チェックサムの強化 53 ページ
  ○ 「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』
  ○ 「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』
  ○ 「START DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』

- **チェックポイントログの変更**　以前のリリースでは、データベースを停止すると、SQL Anywhere はチェックポイントのログを完全にトランケートしました。バージョン 12 では、チェックポイントログの使用履歴がデータベースに格納され、次のセッションのチェックポイントログに適切なサイズを決定するために使用されます。

  セッションにわたってチェックポイントログを格納すると、次回データベースを起動するときにチェックポイント割り当てのオーバーヘッドを防ぎ、ファイルの拡張時に発生する可能性があるファイルの断片化を回避できます。今回のリリースでは、データベース停止時のデータベースファイルは以前のリリースよりも大きくなりますが、次回データベースを再起動するときにチェックポイントログに追加のスペースが再利用されます。「チェックポイントログ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ネットワークデータベースサーバーがワーカーの最大数を割り当てるようになった**　以前のリリースでは、ネットワークデータベースサーバーが起動するとき、データベースサーバーのマルチプログラミングレベルに対応する数のワーカーが割り当てられました。バージョン 12 のネットワークサーバーでは、サーバーの最大マルチプログラミングレベルに対応する数のワーカーが割り当てられます。このようにワーカーの数が増加することによって、ネットワークデータベースサーバーでワーカースタック用に必要となるアドレス領域が増加し、データベースサーバーが割り当てることができるデータベースキャッシュの量に影響を及ぼすことがあります。

  たとえば、32 ビットの Windows プラットフォームでは、デフォルトで、各ワーカーにはスタック用に 1 MB のアドレス領域が必要です。Windows で起動するバージョン 11 のネットワークサーバーでは、デフォルトのマルチプログラミングレベルが 20 であるため、ワーカースタック用に 20 MB のアドレス領域が必要となります。ただし、Windows で起動するバージョン 12 のネットワークサーバーでは、デフォルトのワーカーの最大数が 80 であるため、80 MB のアドレス領域が必要となります。この変更は、パーソナルサーバーや Windows Mobile には影響を及ぼしません。「SQL Anywhere のスレッド化」『SQL Anywhere サーバー データベース管理』を参照してください。

- **古い統計はデータベースの再構築時にロードされない**　バージョン 11 以前のデータベースを再構築する場合、LOAD STATISTICS 文は古い文字列統計の新しいデータベースへのロードを通知しないで省略しますが、文字列統計のバージョンはアップグレードされます。「LOAD STATISTICS 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  アップグレードユーティリティを使用してデータベースをアップグレードしても、文字列統計のバージョンはアップグレードされません。

- **位置付け DELETE 文と UPDATE 文**　以前のバージョンでは、位置付け UPDATE 文または DELETE 文で TOP 句または FIRST 句を指定できましたが、これらの句は無視されました。今回のバージョンでは、位置付け UPDATE 文または DELETE 文で TOP または FIRST を指定すると、構文エラーが返されるようになりました。「UPDATE (位置付け) 文 [ESQL] [SP]」『SQL Anywhere サーバー SQL リファレンス』と「DELETE 文 (位置付け) [ESQL] [SP]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **JDBC ドライバーは CHAR だけでなく CHAR と VARCHAR をレポートするようになった**　以前のバージョンでは、iAnywhere JDBC ドライバーを使用してアプリケーションを接続し、CHAR カラムを含むテーブルまたは結果セットのメタデータを記述しようとすると、メタデータではカラムのタイプ名が CHAR と返されましたが、SQL タイプでは Types.VARCHAR として返されました。JDBC アプリケーションで JDBC ドライバーが CHAR カラムの SQL タイプを Types.CHAR と返すようにするには、アプリケーションで odbc_distinguish_char_and_varchar データベースオプションを設定する必要がありました。今回のバージョンでは、新しい SQL Anywhere JDBC ドライバーと非推奨の iAnywhere JDBC ドライバーで、データベースオプションの設定に関係なく、タイプ CHAR のテーブルと結果セットのカラムに対してタイプ名 CHAR および SQL タイプ Types.CHAR が返され、タイプ VARCHAR のカラムに対してタイプ名 VARCHAR および SQL タイプ Types.VARCHAR が返されるようになりました。

- **CURRENT UTC TIMESTAMP および UTC TIMESTAMP 空間値の変更**　CURRENT UTC TIMESTAMP 空間値、およびデフォルト値 UTC TIMESTAMP 空間値の基本となるデータ型は、TIMESTAMP WITH TIME ZONE になりました。これらの値が TIMESTAMP として定義されたカラムで使用された場合、タイムゾーンオフセットはトランケートされ、動作に目立った違いはありません。ただし、これらの値が CHAR または VARCHAR カラムで使用された場合、オフセットは以前とは異なる値が生成されます。「CURRENT UTC TIMESTAMP 特別値」『SQL Anywhere サーバー SQL リファレンス』と「UTC TIMESTAMP 特別値」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **パーソナルサーバーはデフォルトで TCP/IP を起動しない**　デフォルトでは、パーソナルデータベースサーバーは共有メモリプロトコルのみを起動します。TCP/IP プロトコルを使用する場合は、パーソナルデータベースサーバーを起動するとき、-x サーバーオプションを使用して TCP/IP プロトコルを指定する必要があります。「-x dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』と「通信プロトコルオプションの考慮事項」『SQL Anywhere サーバー データベース管理』を参照してください。

- **TCP/IP 接続**　TCP/IP を介して接続する場合、HOST 接続パラメーターでホスト名が指定されていれば、データベースサーバー名 (ServerName (SERVER) 接続パラメーターで指定) は必須ではなくなりました。「Host 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Host (IP) プロトコルオプション**　以前のリリースでは、Host プロトコルオプションによって、データベースサーバーを実行する可能性がある 1 つ以上のホストが示されましたが、このことは、クライアントライブラリへのヒントと見なされました。これらのホストでデータベースサーバーが見つからない場合、データベースサーバーを検出するためにネットワークブロードキャストが実行されました。今回のリリースでは、Host オプションを指定すると、データベースサーバーについて指定したホストのみが検索され、デフォルトでは、クライアントはデータベースサーバーを検出するためにブロードキャストを実行しません。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

この動作は、DoBroadcast プロトコルオプションを Direct に設定した場合と同じです。HOST プロトコルオプションで指定したコンピューター以外のコンピューターでデータベースサーバーが実行されている場合、データベースサーバーは検出されません。前回のリリースと同じように動作させる場合は、接続文字列で DoBroadcast=All を指定します。「Host (IP) プロトコルオプション (クライアント側のみ)」『SQL Anywhere サーバー データベース管理』と「DoBroadcast (DOBROAD) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ServerPort (PORT) プロトコルオプション**　以前のリリースでは、PORT プロトコルオプションによって、データベースサーバーが受信する可能性がある 1 つ以上のポート番号が示されましたが、このことは、クライアントライブラリへのヒントと見なされました。クライアントライブラリはブロードキャストを送信する場合、PORT プロトコルオプションで指定されたポート番号とデフォルトのポート番号 2638 を使用しました。今回のリリースでは、PORT オプションを指定すると、データベースサーバーを検出するために、指定したポートのみが使用されます。「ServerPort (PORT) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **共有メモリ接続での Idle 接続パラメーターの尊重**　Idle 接続パラメーターでは、接続のアイドルタイムアウト時間を指定します。共有メモリ接続でこの接続パラメーターが尊重されるようになりました。共有メモリ接続のデフォルトのアイドルタイムアウトは 0 (アイドルタイムアウトなし) です。「Idle 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースサーバー名は ServerName (Server) 接続パラメーターで指定した名前と同じでない場合がある**　以前のリリースでは、Name データベースプロパティの値は ServerName (Server) 接続パラメーターで指定した値と常に一致していました。ただし、クライアントがデータベースに接続して新しい NodeType (NODE) 接続パラメーターを使用し、異なるデータベースサーバーに接続するようにリダイレクトされた場合、名前は一致しません。「NodeType (NODE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **代替サーバー名を使用した接続時に一部の操作がサポートされない**　以前のリリースでは、クライアントが代替サーバー名を使用してデータベースに接続した場合、クライアントは同じデータベースサーバー上の他のデータベースを作成、停止、削除できました。代替サーバー名を使用して接続した場合、これらの操作はサポートされなくなりました。

- **データベースミラーリングの動作の変更と廃止予定機能**　-xp オプションによる監視サーバーの名前、認証文字列、同期実行モードなどのデータベースミラーリングオプションの定義は廃止される予定です。ただし、ミラーリングシステムでデータベースサーバーを使用する場合は、引き続き **-xp on** を指定する必要があります。「-xp dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

  次の SQL 文を使用してデータベースミラーリング設定を定義できるようになりました。

  ○ 「CREATE MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
  ○ 「ALTER MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』
  ○ 「SET MIRROR OPTION 文」『SQL Anywhere サーバー SQL リファレンス』

今回リリースでは、データベースミラーリングについて、次の動作の変更が導入されています。

- 以前のリリースでは、ミラーサーバーのステータス情報ファイルの名前は、デフォルトでサーバー名に基づいた名前でした。今回のリリースでは、ステータス情報ファイルの名前を指定する必要があります。
- 以前のリリースでは、ミラーサーバーに送信された Web サービス要求はプライマリサーバーにリダイレクトされました。今回のリリースでは、Web サービス要求は、受け取ったサーバーによって処理されます。

今回のリリースでのデータベースミラーリングの強化については、データベースミラーリングの強化 49 ページを参照してください。

- **Embedded SQL カーソル動作の変更**　Embedded SQL カーソルは、READ ONLY のデフォルトになりました。明示的な FOR READ ONLY 句または FOR UPDATE 句は、DECLARE 文ではなく、PREPARE 文で指定することが必要になりました。次の項を参照してください。

  - 「DECLARE CURSOR 文 [ESQL] [SP]」『SQL Anywhere サーバー SQL リファレンス』
  - 「PREPARE 文 [ESQL]」『SQL Anywhere サーバー SQL リファレンス』

- **CREATE OR REPLACE PROCEDURE 文のカーソルは閉じられる**　CREATE OR REPLACE PROCEDURE 文を実行する場合、接続が開いた他のカーソルはすべて閉じられます。「CREATE PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **逆引用符識別子デリミター**　逆引用符を識別子デリミターとして使用できるようになりました。「識別子」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ロックの動作の変更**　同時実行性を最大化するために、ローのキーとキーではない部分を別々にロックできるようになりました。ローを参照する外部ローの挿入と削除を妨げずに、そのローのキーではないカラムを更新できます。「ロックの仕組み」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ODBC ドライバーの動作の変更**　ODBC 関数 SQLTables を使用すると、SQL_ALL_SCHEMAS 引数で関数を呼び出して、すべてのスキーマ (ユーザー) のリストを取得できます。以前のバージョンでは、この関数によって返されるユーザーのリストには、テーブルを所有するユーザーのみが含まれていました。新しく初期化したデータベースでは、一部のユーザー、特に DBA ユーザーが除外されました。今回のリリースでは、テーブルを所有しないユーザーを含む、スキーマの完全なリストが返されます。

- **divide_by_zero_error オプション**　divide_by_zero_error オプションは、バージョン 11 ではサポートされていませんでした。このオプションはバージョン 12 でサポートされます。マテリアライズドビューを使用している場合は、このオプションを On (デフォルト) に設定して、新しいマテリアライズドビューを作成してください。「divide_by_zero_error オプション」『SQL Anywhere サーバー データベース管理』と「マテリアライズドビューの制限」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **PrefetchBuffer (PBUF) 接続パラメーター**　以前のリリースでは、PrefetchBuffer (PBUF) 接続パラメーターで指定したローをバッファーするためのメモリ量は、すべての接続間で共有されました。今回のリリースでは、この接続パラメーターで指定したメモリ量は、各接続で使

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

用できます。「PrefetchBuffer (PBUF) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **削除されたユーザーの外部ログインは自動的に削除される**　データベースからユーザーを削除すると、そのユーザーのすべての外部ログインは自動的に削除されるようになりました。以前のリリースでは、外部ログインを個別に削除する必要がありました。「ユーザーをデータベースから削除する」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースクリーナーとデータベース検証は同時に実行されなくなった**　以前のリリースでは、データベース検証とデータベースクリーナーを同時に実行し、データベースページへの同時アクセスによるエラーをレポートできました。データベース検証とデータベースクリーナーは、同じデータベース上で同時に実行されなくなりました。検証はデータベースクリーナーの終了を待機し、データベースクリーナーが sa_clean_database を呼び出して開始された場合は、データベースクリーナーが検証の終了を待機します。データベースクリーナーは、テーブル検証またはインデックス検証と同時に実行できます。

- **sa_index_density システムプロシージャー内のカラムに対するデータ型の変更**　密度カラムとスキューカラムのデータ型は、numeric(8,6) から double に変更されました。「sa_index_density システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DATEDIFF 関数**　以前のリリースでは、DATEDIFF 関数は時間とそれ以下の日付の単位に INTEGER を返しました。今回のリリースでは、DATEDIFF はこれらの日付の単位に BIGINT を返すようになりました。「DATEDIFF 関数 [日付と時刻]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **openxml システムプロシージャー**　openxml システムプロシージャーは、dbo データベースユーザーによって所有されなくなりました。**dbo.**でプロシージャー名を修飾する openxml システムプロシージャーを使用するクエリは **dbo.** を削除するように変更する必要があります。次のような文を実行すると、エラーが返されます。

  ```
  SELECT ... FROM dbo.openxml(...)
  ```

  以前のリリースでは、openxml システムプロシージャーは NCHAR データを CHAR エンコードに変換し、CHAR フォーマットでデータを解析しました。今回のリリースでは、出力に NCHAR カラムがある場合、NCHAR データは NCHAR エンコードで解析されます。

  以前のリリースでは、WITH 句内の xpath 引数には、リテラル文字列のみを指定できました。今回のリリースでは、リテラル文字列と変数を指定できるようになりました。「openxml システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  バージョン 11 以前からアップグレードされたデータベースには、SYS.SYSPROCEDURE システムビューに openxml のローが含まれますが、その定義はバージョン 12 のデータベースでは使用できません。プロシージャーを次のように使用すると、構文エラーが返されます。

  ```
  SELECT * FROM dbo."openxml"(...)
  ```

- **sa_text_index_vocab システムプロシージャー**　NCHAR テキストインデックスで sa_text_index_vocab を呼び出そうとするとエラーが返されるようになりました。代わりに新しい sa_text_index_vocab_nchar システムプロシージャーを使用します。

「sa_text_index_vocab_nchar システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

さらに、次の点が変更されました。

○tab_owner パラメーターはオプションになりました。

○CALL 文で sa_text_index_vocab を使用できるようになりました。

○プロシージャー内の文で sa_text_index_vocab を使用できるようになりました。

○パラメーター値をホスト変数または式にできるようになりました。

- **Mac OS X のデフォルト照合の変更**　以前のリリースでは、Mac OS X に環境変数がない場合、初期化ユーティリティ (dbinit) はデフォルトの CHAR 文字セットに ISO_8859-1:1987 を使用し、CHAR 照合に ISO1LATIN1 を使用しました (Linux と同じ動作)。今回のリリースでは、UTF-8 と UTF8BIN 照合が選択されます。「推奨文字セットと照合」『SQL Anywhere サーバー データベース管理』を参照してください。

- **予約語**　次に、SQL Anywhere バージョン 12.0.0 でデータベースに追加された予約語を示します。

  ○datetimeoffset

  ○inner

  ○openxml

  ○spatial

  ○treat

  次に、SQL Anywhere バージョン 12.0.0 でデータベースから削除された予約語を示します。

  ○index_lparen

  ○lock

  ○with_cube

  ○with_lparen

  ○syntax_error

  ○with_rollup

- **WITH (*index-hint*) 句の動作変更**　以前は、テーブルのプライマリキーインデックス、外部キーインデックス、または標準インデックスの名前が同じであった場合、オプティマイザーによってプライマリーまたは外部キーインデックスの名前に解決されていました。現在では、オプティマイザーによって標準のインデックスの名前に解決されるようになりました。それが存在しない場合、オプティマイザーはまずプライマリーキーの名前に解決し、次いで

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

外部キーに解決します。WITH table-hint 句、FROM 句『SQL Anywhere サーバー SQL リファレンス』を参照してください。

SQL Anywhere の廃止予定機能とサポート終了機能

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

- **Address Windowing Extensions (AWE) の廃止** 32 ビット Windows の Address Windowing Extensions は使用されなくなりました。サイズの大きいキャッシュが必要な場合は、64 ビット版のオペレーティングシステムで 64 ビット版の SQL Anywhere データベースサーバーを使用することをおすすめします。「-cw dbeng12/dbsrv12 サーバーオプション (旧式)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **CALL 文** この文を使用した関数の呼び出しは廃止される予定です。呼び出す関数がある場合は、代入文を使用して関数を呼び出し、その結果を変数に代入することを検討してください。次に例を示します。

  ```
  DECLARE varname INT; SET varname=test( );
  ```

  「CALL 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **STOP ENGINE 文** STOP ENGINE 文は使用されなくなりました。代わりに STOP SERVER 文を使用します。「STOP SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **Windows 2000 サポートの削除** SQL Anywhere は、バージョン 12.0.0 からは Windows 2000 でサポートされなくなりました。

- **再構築ユーティリティの削除** このリリースでは、SQL Anywhere データベースを再構築する場合に再構築ユーティリティはサポートされなくなりました。アンロードユーティリティ (dbunload) を使用してデータベースを再構築できます。「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サポート対象外のデータベースプロパティ** このリリースでは、次のプロパティが削除されています。

  - CheckpointLogBitmapPagesWritten データベースプロパティ
  - CheckpointLogBitmapSize データベースプロパティ
  - java_main_userid 接続プロパティ
  - QueryRowsBufferFetch 接続プロパティ
  - QueryRowsBufferFetch データベースプロパティ

- **JDBC ベースのサーバークラスの廃止**　次の JDBC ベースのサーバークラスのサポートは廃止されました。

  ○ asejdbc
  ○ iqjdbc
  ○ sajdbc

  ODBC ベースのサーバークラスを使用するようにアプリケーションを更新してください。「ODBC 外部サーバー定義」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **SQL Anywhere Explorer のサポート終了**　Visual Studio 用の SQL Anywhere Explorer と SQL Anywhere ツールバーはサポートされなくなりました。代わりに Microsoft のサーバーエクスプローラーを使用します。

- **short int embedded SQL インジケーター変数の廃止**　32 ビット長および 64 ビット長とインジケーターを今後使用できるように、Embedded SQL インジケーター変数での short int の使用は推奨されなくなりました。a_sql_len を代わりに使用します。「インジケーター変数」『SQL Anywhere サーバー プログラミング』を参照してください。

- **EngineName (ENG) 接続パラメーターの廃止**　EngineName (ENG) 接続パラメーターは使用されなくなりました。代わりに、Use the ServerName (Server) 接続パラメーターを使用します。ServerName 接続パラメーターの短縮形は ENG から Server に変更されました。「ServerName (Server) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **iAnywhere JDBC ドライバーの廃止**　Type 1 iAnywhere JDBC ドライバーは使用されなくされました。代わりに Type 2 SQL Anywhere JDBC ドライバーを使用します。新しい SQL Anywhere TYPE-2 JDBC ドライバー 66 ページを参照してください。

- **-gu all データベースサーバーオプションの廃止**　-gu データベースサーバーオプションの値 **all** は使用されなくなりました。「-gu dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-sm dbsrv12 データベースオプション (旧式)**　-sm データベースオプションは廃止されます。代わりに CREATE MIRROR SERVER 文を使用します。「CREATE MIRROR SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SET OPTION 文**　識別子を SET OPTION 文で文字列リテラルではなくオプション値として指定する機能は、使用されなくなりました。

- **サービスユーティリティ (dbsvc ユーティリティ)**　-t オプションの値 Standalone は使用されなくなりました。代わりに -t Personal を使用して、パーソナルデータベースサーバーのサービスを作成します。「Linux 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』と「Windows 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サーバーからの Host (IP) プロトコルオプション**　サーバーコマンドでの Host (IP) プロトコルオプションのサポートが削除されました。次のコマンドはサポートされなくなり、エラーが返されます。「dbeng12 -x tcpip(host=host-name) "%SQLANY12%¥demo.db"」ただし、クライアント側から CommLinks(LINKS) 接続パラメーターで Host (IP) プロトコルを指定するこ

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

とはできます。「Host (IP) プロトコルオプション (クライアント側のみ)」『SQL Anywhere サーバー データベース管理』を参照してください。

注意
Host プロトコルオプションは、Host 接続パラメーターとは別のものです。Host プロトコルオプションは、CommLinks 接続パラメーターによって使用されます。CommLinks (LINKS) 接続パラメーターは、Host および ServerPort (PORT) 以外の TCP/IP オプションを指定する必要がある場合にのみ使用してください。接続文字列に CommLinks と Host の両方は指定できません。「CommLinks (LINKS) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

ほとんどの場合、HOST 接続パラメーターを使用してください。「Host 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

Mobile Link の新機能

次に、バージョン 12.0.0 で導入された Mobile Link の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **リモートデータベースの集中管理**　リモートデータベースの集中管理を使用して、次のことができます。

  ○リモートデータベースが Mobile Link と同期するときに集中管理します。以前は、dbmlsync オプションを使用してクライアントで設定するか、アプリケーションにハードコードしていました。

  ○リモートデータベースにスキーマの変更を適用します。

  ○特定のリモートデータベースや一般的な同期システムに関する問題を診断します。

  リモートデータベースの集中管理については、「リモートデータベースの集中管理」『Mobile Link サーバー管理』を参照してください。

- **新しい Mobile Link Replay ユーティリティ (mlreplay)**　Mobile Link Replay ユーティリティは、Mobile Link サーバーによって記録された Mobile Link プロトコルのリプレイに使用されるツールです。「Mobile Link Replay ユーティリティ (mlreplay)」『Mobile Link サーバー管理』を参照してください。

- **Sybase Central の Mobile Link 12 プラグインの再設計**　Mobile Link プラグインはバージョン 12 で再設計され、リモートデータベースの集中管理がサポートされるようになりました。新しいプラグインには、モデルと管理の 2 つの Mobile Link モードが組み込まれました。今回のリリースでは、Mobile Link プラグインを使用して、統合データベース、グループ、同期モデル、リモートタスクを含む同期プロジェクトを作成できるようになりました。同期プロジェクトに古い同期モデルをインポートできます。「リモートデータベースの集中管理」『Mobile Link サーバー管理』を参照してください。

- **特別値への統合カラムのマッピング**　特別値 (ML ユーザーなど) にリモートカラムの代わりに統合カラムをマッピングできるようになりました。

- **ダウンロード削除サブセットのサポート**　ダウンロード削除サブセットを指定できるようになりました。ダウンロード削除サブセットは、デフォルトでダウンロードサブセットと同じです。

- **新しいツリービューを使用したカラムとテーブルの選択**　統合データベースを作成したり更新したりするとき、または既存のリモートデータベースで同期対象のテーブルを選択するときに、カラムとテーブルを選択できるようになりました。この選択は、テーブルとカラムのツリービューで、同期するテーブルやカラムの横にチェックマークを付けることによって簡単に実行できます。

- **空間データ型のサポート**　同期モデルで空間データ型および TIMESTAMP WITH TIME ZONE データ型がサポートされるようになりました。

- **シャドウテーブルにインデックスを用意**　同期モデルによって作成されるシャドウテーブルにインデックスが作成されるようになりました。これにより、シャドウテーブルを使用する download_delete_cursor スクリプトと download_cursor スクリプトを高速化できます。

- **SQL Anywhere モニターで Mobile Link サーバーファームと Relay Server ファームをサポート**　今回のリリースでは、SQL Anywhere モニターを使用して、SQL Anywhere データベースと Mobile Link サーバーだけでなく Mobile Link サーバーファームと Relay Server ファームもモニタリングできるようになりました。「SQL Anywhere モニター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サーバーファーム用の新しい Mobile Link 監視サーバーユーティリティ**　サーバー起動同期および QAnywhere について、Mobile Link 監視サーバーは、サーバーファーム内の 1 つの Mobile Link サーバーだけがプライマリサーバーとして動作するようにします。「アーキテクチャー」『Mobile Link クイックスタート』を参照してください。

- **サーバーファームの強化サポート**　新しいリモート ID ロックロジックを使用すると、Mobile Link 高可用性の同じリモート ID からの冗長な同期を防止できます。-ss オプションを設定する必要がなくなりました。サーバー起動同期または QAnywhere を Mobile Link サーバーファームと使用する場合は、監視サーバーが必要です。

  注意
  サーバーファームでの Mobile Link サーバーの実行は、Mobile Link の高可用性オプションの機能であり、別途ライセンスが必要です。「別途ライセンスが必要なコンポーネント」『SQL Anywhere 12 紹介』を参照してください。

- **TLS 暗号パッケージプログラムのサポートの変更**　Mobile Link サーバーおよびクライアントで、RSA と ECC の両方の 256 ビット AES 暗号パッケージプログラムがサポートされるようになりました。また、ECC 暗号パッケージプログラムの RFC 4492 バージョンのサポートも追加されました。

- **動的メモリキャッシュ**　動的メモリキャッシュでは、メモリキャッシュの使用量が増えるため、スワップを防ぐためにより大きいキャッシュサイズが必要になる場合があります。全体的なメモリの使用量はほぼ同じです。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい統合化 Outbound Enabler**　mlsrv12 の -x オプションの新しい OE プロトコルを使用して、rsoe コマンドで呼び出されるスタンドアロンの Outbound Enabler の代わりに統合化 Outbound Enabler を使用します。「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **Sybase Central 用の新しい Relay Server プラグイン**　Sybase Central 用の新しい Relay Server プラグインを使用すると、バックエンドファームとバックエンドサーバーを設定できます。Relay Server プラグインは、Windows と Linux のみでサポートされています。「Sybase Central 用の Relay Server プラグイン」『Relay Server』を参照してください。

- **アップロードおよびダウンロードデータスクリプトの要件**　ml_add_missing_dnld_scripts ストアドプロシージャーを使用して、欠落している download_cursor スクリプトか download_delete_cursor スクリプトまたはその両方を修正します。

SQL 文

次に、SQL Anywhere バージョン 12.0.0 での Mobile Link 用の SQL の強化を示します。

- **新しい SYNCHRONIZE 文 [Mobile Link]**　この新しい文を使用すると、SQL Anywhere リモートデータベースを Mobile Link サーバーと同期できます。「SYNCHRONIZE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい START SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]**　この新しい文を使用すると、SQL Anywhere リモートデータベースで Mobile Link の同期スキーマ変更を開始できます。「START SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい STOP SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]**　この新しい文を使用すると、SQL Anywhere リモートデータベースで Mobile Link の同期スキーマ変更を停止できます。「STOP SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link] の強化**　スクリプトバージョンとスクリプト名を指定できるようになりました。「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link] の強化**　新しい SET SCRIPT VERSION 句を使用して、同期中に使用するスクリプトバージョンを指定できます。RENAME 句を使用してサブスクリプションの名前を変更することもできます。「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

統合データベース

次に、SQL Anywhere バージョン 12.0.0 での Mobile Link 用の統合データベースサポートの強化を示します。

- **新しくサポートされる統合データベース**　Mobile Link バージョン 12.0.0 では、次の新しい統合データベースがサポートされます。

  - IBM DB2 LUW 9.7
  - SQL Anywhere 12

- **空間データの同期**　次の統合データベースの空間データ型の同期がサポートされるようになりました。SQL Anywhere、Oracle、Microsoft SQL Server、IBM DB2、MySQL。「空間データの同期」『Mobile Link サーバー管理』を参照してください。

- **より長い CHAR カラムのサポート**　Mobile Link サーバーで、バイト長が 32767 を超える CHAR、VARCHAR、NCHAR、NVARCHAR リモートデータベースカラムの同期がサポートされるようになりました。

- **TIMESTAMP WITH TIME ZONE データ型を持つカラムの同期のサポート**　Mobile Link サーバーで、TIMESTAMP WITH TIME ZONE データ型を持つリモートデータベースカラムの同期がサポートされるようになりました。

iAnywhere Solutions 12 - Oracle ODBC ドライバー

次に、SQL Anywhere バージョン 12.0.0 に導入された iAnywhere Solutions 12 - Oracle ODBC ドライバーの強化点を示します。

- **パッケージ化されたプロシージャーでの VARRAY のサポート**　iAnywhere Solutions 12 - Oracle ODBC ドライバーで、パッケージ化されたプロシージャーでの VARRAY の使用がサポートされるようになりました。

Mobile Link サーバー

次に、SQL Anywhere バージョン 12.0.0 での Mobile Link サーバーの強化を示します。

mlsrv12 の新機能

- **サーバー名を mlsrv12 に変更**　Mobile Link サーバーは mlsrv11 から mlsrv12 に変更されました。

- **mlsrv12 の新しい -cmax オプション**　新しいオプションは、動的キャッシュサイズ決定機能の一部です。サーバーのメモリキャッシュの最大サイズを設定します。「-cmax mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -cinit オプション**　新しいオプションは、動的キャッシュサイズ決定機能の一部です。サーバーのメモリキャッシュの初期サイズを設定します。「-cinit mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -cmin オプション**　新しいオプションは、動的キャッシュサイズ決定機能の一部です。サーバーのメモリキャッシュの最小サイズを設定します。「-cmin mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の -x オプションの新しいプロトコルオプション**　mlsrv12 の -x オプションで、新しい oe プロトコルオプションがサポートされるようになりました。このプロトコルオプショ

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ンを使用すると、Relay Server の使用時に新しい統合化 Outbound Enabler を使用できます。「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の -sl java オプションの新しいデフォルト**　デフォルトで、Mobile Link サーバーでは、サーバー Java VM が存在する場合、クライアント VM ではなくサーバー Java VM のロードを試みるようになりました。この新しいデフォルトを上書きするには、-sl java オプションに -client を追加して、クライアント VM を明示的に要求します。「-sl java mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -ca オプション**　QAnywhere で複数のサーバーを使用する場合、またはライトウェイトポーリングなしでサーバー起動同期を行う場合は、Mobile Link 監視サーバーを実行するホストの名前を入力します。「-ca mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -ftru オプション**　このオプションは、Mobile Link サーバーでファイルアップロードサポートを有効にするために追加されました。このオプションを使用すると、mlfiletransfer ユーティリティによって、アップロードするファイルのルートディレクトリを設定できます。「-ftru mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の -ppv オプションの新しいメトリック**　mlsrv12 の -ppv オプションに多数の新しいメトリックが追加されました。「-ppv mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -rrp オプション**　mlsrv12 の新しい -rrp オプションが Mobile Link Replay ユーティリティ (mlreplay) と組み合わせて追加されました。このオプションを指定すると、Mobile Link サーバーの起動時に mlreplay ユーティリティが実行され、指定したディレクトリ内の記録されたセッションがすべてリプレイされます。「-rrp mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -rp オプション**　mlsrv12 の新しい -rp オプションが Mobile Link Replay ユーティリティ (mlreplay) と組み合わせて追加されました。このオプションは、mlreplay ユーティリティよるプレイバック用の同期を記録するディレクトリを指定するために使用します。「-rp mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -vR オプション**　mlsrv12 の新しい -vR オプションを使用すると、同期の各ログメッセージ内のリモート ID が表示されます。「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -vU オプション**　mlsrv12 の新しい -vU オプションを使用すると、同期の各ログメッセージ内の Mobile Link ユーザー名が表示されます。「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の新しい -vk オプション**　新しいオプションは、動的キャッシュサイズ決定機能の一部です。このオプションでは、キャッシュサイズが増大または縮小すると、ログに 1 行が追加されます。「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **Mobile Link サーバーでのメモリ不足パフォーマンスの改善**　メモリ不足状態における動作と堅牢性が改善されました。

- **2 つの新しい競合検出方法**　upload_fetch、upload_fetch_column_conflict、upload_new_row_insert、upload_old_row_insert の各スクリプトを使用して、競合を検出できます。「競合検出」『Mobile Link サーバー管理』を参照してください。

- **新しい ml_add_missing_dnld_scripts システムプロシージャー**　新しいシステムプロシージャーを使用して、欠落した download_cursor スクリプトと download_delete_cursor スクリプトを追加できます。「ml_add_missing_dnld_scripts システムプロシージャー」『Mobile Link サーバー管理』を参照してください。

Mobile Link サーバーの新しい API 機能

- **新しい TimestampWithTimeZone クラスの追加 (Java)**　このクラスは、Timestamp 値のタイムゾーンオフセットを取得および指定できるメソッドを提供します。TimestampWithTimeZone クラス [Mobile Link サーバー Java]『Mobile Link サーバー管理』を参照してください。

- **新しい DateTimeWithTimeZone クラスの追加 (.NET)**　このクラスは、DateTime 値のタイムゾーンオフセットを取得および指定できるメソッドを提供します。DateTimeWithTimeZone クラス [Mobile Link サーバー .NET]『Mobile Link サーバー管理』を参照してください。

- **新しい SpatialUtilities クラスの追加 (.NET および Java)**　このクラスは、空間データを操作する場合に役立つメソッドを提供します。SpatialUtilities クラス [Mobile Link サーバー .NET]『Mobile Link サーバー管理』 (.NET) と SpatialUtilities クラス [Mobile Link サーバー Java]『Mobile Link サーバー管理』 (Java) を参照してください。

Mobile Link の新しいスクリプト機能

- **upload_fetch スクリプトと upload_fetch_column_conflict スクリプトに使用できる新しいシステムパラメーター**　Mobile Link サーバーのシステムパラメーター「remote_id」と「username」を upload_fetch スクリプトと upload_fetch_column_conflict スクリプトに使用できるようになりました。

- **authenticate_file_transfer スクリプトに使用できる新しいパラメーター**　authenticate_file_transfer スクリプトに新しいパラメーターを使用できます。「authenticate_file_transfer 接続イベント」『Mobile Link サーバー管理』を参照してください。

- **新しい authenticate_file_upload 接続イベント**　ファイルアップロードのサポートに新しい接続イベントを使用できます。「authenticate_file_upload 接続イベント」『Mobile Link サーバー管理』を参照してください。

- **新しい generate_next_last_download_timestamp スクリプト**　次の最終ダウンロード時間を決定する Mobile Link のデフォルトアルゴリズムを抑制する場合は、modify_next_last_download_timestamp の代わりにこのスクリプトを使用します。「generate_next_last_download_timestamp イベント」『Mobile Link サーバー管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

サーバーユーティリティ

次に、SQL Anywhere バージョン 12.0.0 での Mobile Link サーバーユーティリティの強化を示します。

- **新しい mlarbiter ユーティリティ**　新しい mlarbiter ユーティリティを使用して、Mobile Link 監視サーバーを起動します。サーバー起動同期または QAnywhere を Mobile Link サーバーファームと使用する場合は、監視サーバーが必要です。「Windows 用 Mobile Link 監視サーバーユーティリティ (mlarbiter)」『Mobile Link サーバー管理』を参照してください。

Mobile Link モニター

次に、SQL Anywhere バージョン 12.0.0 での Mobile Link モニターの強化を示します。

- **グラフウィンドウ枠の新しいエントリ**　Mobile Link モニターのグラフウィンドウ枠に次のエントリが追加されました。OE ワークキュー、Notifier ワークキュー、動的キャッシュワークキュー。「使用率グラフの使用」『Mobile Link サーバー管理』を参照してください。

- **新しい [信頼できる証明書ファイル] オプション**　新しい **[信頼できる証明書ファイル]** オプションと **[参照]** ボタンが **[Mobile Link サーバーへの接続]** ウィンドウに追加されました。

Mobile Link クライアント

次に、SQL Anywhere バージョン 12.0.0 での Mobile Link クライアントの強化を示します。

- **dbmlsync によるバックグラウンド同期のサポートの強化**　dbmlsync がロックしたデータベースリソースへのアクセスを別の接続が待機している場合、データベースエンジンでリモートデータベースへの dbmlsync 接続を削除 (および dbmlsync のコミットされていない操作をロールバック) できるようになりました。これにより、他の接続は同期の完了を待機しないで先に進むことができます。

  バックグラウンド同期に次のオプションを使用します。

  - 「-bk dbmlsync オプション」『Mobile Link クライアント管理』
  - 「-bkr dbmlsync オプション」『Mobile Link クライアント管理』
  - 「Mobile Link 同期プロファイル」『Mobile Link クライアント管理』の Background オプションと BackgroundRetry オプション

- **dbmlsync での名前付きサブスクリプションのサポート**　dbmlsync の新しい -s オプションを使用すると、同期させるサブスクリプションの名前を指定できます。「-s dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **HTTP 応答バッファリング**　新しいネットワークプロトコルオプション http_buffer_responses を使用すると、Mobile Link からの HTTP パケットを、ネットワークから読み込むと同時に処理するのではなく、処理前に、中間バッファーに完全にストリーミングすることができます。「http_buffer_responses」『Mobile Link クライアント管理』を参照してください。

- **クライアント側の証明書を使用したサードパーティのサーバーとプロキシへの Mobile Link クライアントの認証**　この機能をサポートするために、identity 同期パラメーターと identity_password 同期パラメーターが追加されました。

- **リモートデータベースでのスキーマ変更の実装に同期が不要になった (SQL Anywhere クライアントのみ)**　START SYNCHRONIZATION SCHEMA CHANGE 文と STOP SYNCHRONIZATION SCHEMA CHANGE 文を使用してスキーマの変更を区切ることにより、リモートデータベースでのスキーマの変更を簡略化できます。次の項を参照してください。

  ○「START SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』
  ○「STOP SYNCHRONIZATION SCHEMA CHANGE 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』

  CREATE SYNCHRONIZATION SUBSCRIPTION 文または ALTER SYNCHRONIZATION SUBSCRIPTION 文を使用して、各同期サブスクリプションのスクリプトバージョンを指定します。次の項を参照してください。

  ○「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』
  ○「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』

- **Ultra Light の新しい mlfileupload メソッド**　Ultra Light クライアントの MLFileUpload メソッドが追加されました。

- **dbmlsync オプションウィンドウの変更**　dbmlsync オプションウィンドウは、次のように変更されました。

  ○ **[リモートプログレスに対してリトライする]**、**[リモートのオフセットが後]**、**[リモートのオフセットが前]**、**[競合する接続の削除]**、**[サイトスクリプト]**、**[コマンドラインヘルプ]** は削除されました。
  ○ **[パブリケーション]** オプションは **[サブスクリプション]** オプションに置き換えられました。

新しい dbmlsync オプション

- **dbmlsync の新しい -s オプション**　dbmlsync の新しい -s オプションを使用して、同期させるサブスクリプションの名前を指定します。「-s dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しい -bk オプション**　このオプションは、バックグラウンド同期を有効にします。「-bk dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しい -bkr オプション**　このオプションは、バックグラウンド同期が中断された後の dbmlsync の動作を制御します。「-bkr dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しい -ci オプション**　このオプションは、新しい動的キャッシュサイズ決定機能の一部です。このオプションを使用して、データの同期時に dbmlsync で使用するキャッ

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

シュの初期サイズを設定します。「-ci dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しい -cl オプション**　このオプションは、新しい動的キャッシュサイズ決定機能の一部です。このオプションを使用して、dbmlsync キャッシュファイルを縮小する最小サイズを設定します。「-cl dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しい -cm オプション**　このオプションは、新しい動的キャッシュサイズ決定機能の一部です。このオプションを使用して、dbmlsync キャッシュファイルの最大サイズ制限を設定します。「-cm dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しい BufferDownload 拡張オプション**　このオプションは、新しい動的キャッシュサイズ決定機能の一部です。このオプションでは、Mobile Link サーバーからのダウンロードを、リモートデータベースに適用する前にすべてキャッシュに読み込むかどうかを指定します。「BufferDownload (bd) 拡張オプション」『Mobile Link クライアント管理』を参照してください。

新しい dbmlsync C++ API オブジェクト

- **CancelSync メソッド**　この方法を使用すると、Mobile Link クライアントで同期をキャンセルできます。DbmlsyncClient.CancelSync メソッド [Dbmlsync C++]『Mobile Link クライアント管理』を参照してください。

- **DBSC_CancelRet 列挙**　この列挙は、同期のキャンセル試行の結果を示します。DBSC_CancelRet 列挙 [Dbmlsync C++]『Mobile Link クライアント管理』を参照してください。

- **DBSC_ERR_ACTIVE_SYNC_NOT_CANCELED メンバー**　このメンバーは、同期がアクティブであるために、サーバーが同期要求をキャンセルできなかったことを示します。DBSC_ErrorType 列挙 [Dbmlsync C++]『Mobile Link クライアント管理』を参照してください。

- **DBSC_ERR_DEAD_SERVER メンバー**　このメンバーは、起動時に dbmlsync サーバーにエラーが発生し、シャットダウンされたことを示します。DBSC_ErrorType 列挙 [Dbmlsync C++]『Mobile Link クライアント管理』を参照してください。

- **"enable status" プロパティ**　このプロパティが有効な場合、3 つの新しいイベント (DBSC_EVENTTYPE_ML_CONNECT、DBSC_EVENTTYPE_UPLOAD_COMMITTED、DBSC_EVENTTYPE_DOWNLOAD_COMMITTED) が Mobile Link クライアントに送信されます。DBSC_EventType 列挙 [Dbmlsync C++]『Mobile Link クライアント管理』、DbmlsyncClient.SetProperty メソッド [Dbmlsync C++]『Mobile Link クライアント管理』、DbmlsyncClient.GetProperty メソッド [Dbmlsync C++]『Mobile Link クライアント管理』を参照してください。

新しい dbmlsync .NET API オブジェクト

- **CancelSync メソッド**　この方法を使用すると、Mobile Link クライアントで同期をキャンセルできます。DbmlsyncClient.CancelSync メソッド [Dbmlsync .NET]『Mobile Link クライアント管理』を参照してください。

- **DBSC_CancelRet 列挙**　この列挙は、同期のキャンセル試行の結果を示します。DBSC_CancelRet 列挙 [Dbmlsync .NET]『Mobile Link クライアント管理』を参照してください。

- **DBSC_ERR_ACTIVE_SYNC_NOT_CANCELED メンバー**　このメンバーは、同期がアクティブである場合、サーバーが同期要求をキャンセルできなかったかどうかを示します。DBSC_ErrorType 列挙 [Dbmlsync .NET]『Mobile Link クライアント管理』を参照してください。

- **DBSC_ERR_DEAD_SERVER メンバー**　このメンバーは、起動時に dbmlsync サーバーにエラーが発生し、シャットダウンされたことを示します。DBSC_ErrorType 列挙 [Dbmlsync .NET]『Mobile Link クライアント管理』を参照してください。

- **"enable status" プロパティ**　このプロパティが有効な場合、3 つの新しいイベント (DBSC_EVENTTYPE_ML_CONNECT、DBSC_EVENTTYPE_UPLOAD_COMMITTED、DBSC_EVENTTYPE_DOWNLOAD_COMMITTED) が Mobile Link クライアントに送信されます。DBSC_EventType 列挙 [Dbmlsync .NET]『Mobile Link クライアント管理』、DbmlsyncClient.SetProperty メソッド [Dbmlsync .NET]『Mobile Link クライアント管理』、DbmlsyncClient.GetProperty メソッド [Dbmlsync .NET]『Mobile Link クライアント管理』を参照してください。

新しいクライアントユーティリティ機能

- **mlfiletransfer を使用したファイルのアップロード**　mlfiletransfer ユーティリティを使用してファイルをアップロードできるようになりました。「Mobile Link ファイル転送ユーティリティ (mlfiletransfer)」『Mobile Link クライアント管理』を参照してください。

- **mlfiletransfer ユーティリティの新しい -i オプション**　再開機能を無効にするために、mlfiletransfer に -i オプションが追加されました。「Mobile Link ファイル転送ユーティリティ (mlfiletransfer)」『Mobile Link クライアント管理』を参照してください。

Relay Server

このリリースでは、次の Relay Server 機能が追加されています。

- **SQL Anywhere モニターのサポート**　SQL Anywhere モニターを使用して、Relay Server リソースをモニタリングできるようになりました。「SQL Anywhere モニター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **集中管理のサポート**　Relay Server は、Sybase Central を使用して集中管理をサポートします。

- **Mac OS での RSOE サポート**　Mac OS で Relay Server Outbound Enabler がサポートされるようになりました。サポートされるプラットフォームの詳細については、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **Windows IIS 7 と 7.5 のサポート**　Relay Server は、Windows Server 2008 上の IIS 7 と Windows Server 2008 R2 上の IIS 7.5 でサポートされるようになりました。サポートされるプラットフォームの詳細については、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい active_cookie オプション**　Relay Server 設定のバックエンドファームセクションに active_cookie オプションが追加されました。「バックエンドファームセクションのプロパティ」『Relay Server』を参照してください。

- **新しい active_header オプション**　Relay Server 設定のバックエンドファームセクションに active_header オプションが追加されました。「バックエンドファームセクションのプロパティ」『Relay Server』を参照してください。

- **Relay Server トラブルシューティングの向上**　トラブルシューティングを向上させるため、Relay Server に、ローカライズされたエラーメッセージを含む標準化された Relay Server エラーコード、システムエラーコードを含む選択的エラーメッセージ、埋め込みのシステムエラーメッセージが用意されました。「Relay Server のエラーメッセージ」『エラーメッセージ』を参照してください。

- **バックエンド HTTP サーバーとしての SQL Anywhere データベースのサポート**　Relay Server で、SQL Anywhere フェールオーバーと読み込み専用のスケールアウトがサポートされるようになりました。「SQL Anywhere Web サービスの高可用性とスケールアウトソリューション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Outbound Enabler 動的応答バッファーサイズ決定**　動的応答バッファーサイズ決定によって、Outbound Enabler のメモリオーバーヘッドが大幅に減少します。

- **Outbound Enabler ユーザーインターフェイスの改善**　ウィンドウタイトル内のインスタンス識別子、systray ヒントテキストと systray メニューがすべて Outbound Enabler ユーザーインターフェイスに追加され、Outbound Enabler ステータスが systray ヒントテキストに追加されました。

- **共有メモリのより効率的な使用**　Relay Server では共有メモリ (shared_mem 設定オプションによって設定) がより効率的に使用されます。比較的読み込みが遅いクライアントが、サイズの大きい応答を伴う HTTP 要求を実行する配備では、Relay Server を非常に少ない共有メモリで実行できます。

- **Relay Server Outbound Enabler による HTTP を使用した定期的なバックエンドサーバーステータス要求の新規サポート**　RSOE が強化され、HTTP を使用した定期的なバックエンドサーバーステータス要求がサポートされるようになりました。バックエンドサーバーステータス要求は活性 ping の代替であり、バックエンドサーバーがクライアント要求を受け入れることができるかどうかを確認する場合に使用できます。rsoe -cs オプションの一部として指定する新しい status_url パラメーターを使用すると、定期的なバックエンドサーバーステータス要求を有効にできます。

Mobile Link の動作の変更

次に、バージョン 12.0.0 で導入された Mobile Link の動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

Mobile Link サーバーの変更

- **Mobile Link サーバーで *log4j.jar* が不要になった**　*log4j.jar* ファイルは Mobile Link サーバーで不要になり、Mobile Link サーバーに配備されなくなりました。*log4j.jar* が必要な場合は、jar の独自のバージョンをインストールし、クラスパスに格納する必要があります。

- **-cn オプションの新しい動作**　mlsrv12 の -cn オプションでは、データベースワーカースレッドに使用されるデータベース接続の最大数が設定されます。SQL Anywhere 12 より前のバージョンでは、mlsrv12 の -cn オプションでデータベース接続の最大数が設定されました。「-cn mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-sl dnet オプションの新しい動作**　以前のバージョンでは、Mobile Link サーバーは、.NET スクリプトの使用時にデフォルトでワークステーション CLR をロードしました。このリリースでは、サーバー CLR をロードするようになりました。以前の動作をリストアするには、mlsrv12 の -sl dnet オプションに **-clrFlavor=wks** を追加します。「-sl dnet mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **Mobile Link サーバーで V_$TRANSACTION の代わりに GV_$TRANSACTION Oracle システムビューを使用**　Mobile Link サーバーで使用される Oracle アカウントに、V_$TRANSACTION システムビューではなく GV_$TRANSACTION Oracle システムビューのパーミッションが必要になりました。「Oracle 統合データベース」『Mobile Link サーバー管理』を参照してください。

- **バージョン 6.0.x のアップグレードスクリプトの削除**　6.0.x の Mobile Link アップグレードスクリプトは削除されました。このアップグレードが必要な場合は、テクニカルサポート (http://www.sybase.com/support) に連絡してください。

- **ml_add_column システムプロシージャーは不要**　バージョン 12 以降のクライアントでは、名前付きパラメーター内のカラム名を使用する場合、ml_add_column システムプロシージャーは不要になりました。デフォルトで、追加設定しなくてもカラム名を直接参照できるようになりました。「ml_add_column system procedure」『Mobile Link サーバー管理』を参照してください。

- **Java と .NET サーバー API の新しい BIGINT データ型サポート**　SQL データ型の BIGINT は Java データ型と .NET データ型の LONG にマッピングされるようになりました。「SQL データ型と Java データ型」『Mobile Link サーバー管理』と「SQL データ型と .NET データ型」『Mobile Link サーバー管理』を参照してください。

- **データスクリプトが必要**　誤ってデータスクリプトが作成されないことによってデータが消失するリスクを減らすため、Mobile Link サーバーでは、次のイベントについて、無視されたスクリプトか有効なスクリプトのいずれかが必要になりました。「無視されたスクリプト」『Mobile Link サーバー管理』を参照してください。

  - **upload_insert**　挿入されたローがリモートからアップロードされ、かつ、handle_UploadData 接続スクリプトが定義されていない場合。

  - **upload_update**　更新されたローがリモートからアップロードされ、かつ、handle_UploadData 接続スクリプトが定義されていない場合。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○ **upload_delete**　削除されたローがリモートからアップロードされ、かつ、handle_UploadData 接続スクリプトが定義されていない場合。

○ **download_cursor と download_delete_cursor**　handle_DownloadData 接続スクリプトが定義されておらず、かつ、同期がアップロード専用でない場合。

バージョン 12 にアップグレードする場合は、ml_add_missing_dnld_scripts ストアドプロシージャーを使用して無視されたスクリプトを追加すると、ダウンロードスクリプトの欠如によるエラーを回避できます。「ml_add_missing_dnld_scripts システムプロシージャー」『Mobile Link サーバー管理』を参照してください。

Mobile Link クライアントの変更

次に、バージョン 12.0.0 で導入された Mobile Link クライアントの動作の変更を示します。

- **バージョン 12 の Mobile Link 同期クライアントはデフォルトでカラム名を送信**　バージョン 12 では、Mobile Link 同期クライアントはすべてデフォルトで Mobile Link サーバーにカラム名を送信します。つまり、ほとんどの場合、Mobile Link スクリプトの名前付きパラメーターで使用するカラム名と順序付けを定義するときに ml_add_column システムストアドプロシージャーを使用する必要がなくなりました。「ml_add_column system procedure」『Mobile Link サーバー管理』と「SendColumnNames (scn) 拡張オプション」『Mobile Link クライアント管理』を参照してください。

- **強化された TLS セッション再ネゴシエーションのサポート**　Mobile Link クライアントで、脆弱なサードパーティのサーバーを保護する新しい TLS 拡張がサポートされるようになりました。

- **MLFileTransfer メソッドの名前変更**　Ultra Light クライアントの MLFileTransfer メソッドは、MLFileUpload メソッドと MLFileDownload メソッドに分割されました。

- **-df の -lf への名前変更**　mlfiletransfer ユーティリティの -df オプションは -lf に名前変更され、宛先ファイルではなくローカルファイルを参照するようになりました。

- **-dp の -lp への名前変更**　mlfiletransfer ユーティリティの -dp オプションは -lp に名前変更され、宛先パスではなくローカルパスを参照するようになりました。

- **UPLD_ERR_USERID_ALREADY_IN_USE の変更**　Dbmlsync は sp_hook_dbmlsync_upload_end イベントフックの障害の原因として UPLD_ERR_USERID_ALREADY_IN_USE を返さなくなりました。代わりに、値 UPLD_ERR_REMOTE_ID_ALREADY_IN_USE が返されます。

- **Ultra Light クライアントでの Palm のサポート終了**　Palm OS のサポートは終了しました。Palm を使用する場合は、引き続き SQL Anywhere 11 を使用してください。

Sybase Central の Mobile Link プラグインの変更

次に、バージョン 12.0.0 で導入された Sybase Central の Mobile Link プラグインの動作の変更を示します。

- **同期モデルのマッピングエディターの変更**

  ○同期モデルのテーブルマッピングエディターとカラムマッピングエディターでは、左側に統合データベース、右側にリモートデータベースが表示されるようになりました。「テーブルマッピングとカラムマッピングの変更」『Mobile Link クイックスタート』を参照してください。

  ○テーブルマッピングエディターには、同期される統合データベースのみが表示されます。テーブルマッピングの方向を **[同期しない]** に変更することは、テーブルマッピングを削除し、モデルの保存時に、データベーススキーマを変更しないでそのローをエディターから削除することと同じです。

  ○同期されていない統合テーブルのテーブルマッピングを追加し、必要に応じてリモートスキーマにテーブルを追加するには、**[テーブルマッピング新規作成]** コマンドを使用します。**[スキーマの更新]** コマンドを使用して、モデルの統合スキーマまたはリモートスキーマを変更することもできます。

  ○テーブルマッピングとカラムマッピングの右側の選択に、ドロップダウンリストの代わりにポップアップメニューが使用されるようになりました。メニュー内のすべての選択肢が適合しない場合、オプションのウィンドウ (省略記号ボタンでアクセス) が表示されます。テーブルマッピングでは、同期されていないリモートテーブルのみが表示されます。カラムマッピングでは、リモートテーブル内の同期されていないカラムが、値マッピングのオプションおよびカラムを同期しないオプションとともに表示されます。

  ○テーブルマッピングの同期オプションを変更する場合、下ウィンドウ枠が、対応するサブオプションのタブに自動的に切り替わります。

- **VARBIT カラムと LONG VARBIT カラム**　リモートスキーマを使用する同期モデルを新しい Ultra Light リモートデータベースに展開する場合、VARBIT カラムと LONG VARBIT カラムの代わりに、VARCHAR カラムと LONG VARCHAR カラムがそれぞれ使用されます。

- **文デリミターとしての GO の使用**　SQL Anywhere データベースと UltraLight データベースでは、同期モデルで文のデリミターとしてセミコロンの代わりに GO が使用されるようになりました。これにより、同期モデルで生成された SQL をリモートタスクの集中管理で使用できるようになりました。

Mobile Link の廃止予定機能とサポート終了機能

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

- **IBM DB2 メインフレームはバージョン 12.0.0 以降でサポート終了**　バージョン 12.0.0 では、IBM DB2 メインフレームは統合データベースとしてサポートされていません。ただし、Mobile Link では、DB2 LUW (Linux、UNIX、Windows) が統合データベースとして引き続きサポートされています。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **Adaptive Server Enterprise 12.5.x のサポート終了**　Adaptive Server Enterprise 12.5.x は、Mobile Link バージョン 12.0.0 ではサポートされなくなりました。
- **IBM DB2 LUW 8.2 のサポート終了**　IBM DB2 LUW 8.2 は、Mobile Link バージョン 12.0.0 ではサポートされなくなりました。
- **mlsrv12 の -xo オプションの削除**　バージョン 10 より前のクライアントはサポートされなくなりました。
- **mlsrv12 の -f オプションの削除**　-zf mlsrv12 オプションを使用すると、各同期の始めに Mobile Link サーバーがスクリプトの変更をチェックするように指定できます。「-zf mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。
- **mlsrv12 の -nba オプションの削除**　ブロッキングダウンロード確認はサポートされなくなりました。リモートデータベースがダウンロード確認を要求する場合、Mobile Link サーバーは常に非ブロッキング確認を使用します。
- **mlsrv12 の -fr オプションの削除**　mlsrv12 の -fr オプションはサポートされなくなりました。(データ消失の原因となる可能性がある) スクリプトを無視する場合は、スクリプトを --{ml_ignore} として定義します。
- **SQL を返す Java データスクリプトと .NET データスクリプトは廃止予定**　Java および .NET スクリプトロジックの、Mobile Link サーバーによって SQL スクリプトとして解釈される文字列を返す機能は、データスクリプトで廃止される予定です。統合データベースでスクリプトによって変更を加える場合は、Java または .NET から直接変更を加えます。

  次の項を参照してください。

  ○「SQL を返す Java と .NET のデータスクリプト (非推奨)」『Mobile Link サーバー管理』
  ○「ダイレクトローハンドリング」『Mobile Link サーバー管理』
  ○「データスクリプト」『Mobile Link サーバー管理』
- **ダウンロードエラーフックの削除**　次の dbmlsync フックはバージョン 12 で削除されました。sp_hook_dbmlsync_download_com_error、sp_hook_dbmlsync_download_fatal_sql_error、sp_hook_dbmlsync_download_sql_error。
- **Mobile Link ファイル転送ユーティリティの -f オプションの削除**　mlfiletransfer ユーティリティの -f オプションはサポートされなくなりました。
- **Mobile Link ファイル転送ユーティリティの -r オプションの削除**　mlfiletransfer ユーティリティの -r オプションはサポートされなくなりました。
- **dbmlsync の Memory (mem) 拡張オプションと DownloadBufferSize (dbs) 拡張オプションの削除**　dbmlsync の Memory (mem) 拡張オプションと DownloadBufferSize (dbs) 拡張オプションはサポートされなくなりました。代わりに、CacheMin オプション、CacheInit オプション、CacheMax オプションを使用します。
- **SQL パススルーの dbmlsync サポートの削除**　SQL パススルー機能は、Mobile Link クライアントでサポートされなくなりました。リモートデータベース機能の集中管理に置き換えられました。「リモートデータベースの集中管理」『Mobile Link サーバー管理』を参照してください。

- **mlsrv12 の -ss オプションは不要**　12 より以前のバージョンでは、mlsrv12 の -ss オプションは、Mobile Link サーバーをサーバーファーム環境で実行できるようにするために使用されました。冗長な同期を防止する新しいリモート ID ロックロジックによって、-ss オプションは、Mobile Link サーバーをサーバーファームで実行する場合に必要でなくなったため、削除されました。サーバー起動同期または QAnywhere を Mobile Link サーバーファームと使用する場合は、監視サーバーが必要です。

  注意
  サーバーファームでの Mobile Link サーバーの実行は、Mobile Link の高可用性オプションの機能であり、別途ライセンスが必要です。「別途ライセンスが必要なコンポーネント」『SQL Anywhere 12 紹介』を参照してください。

- **Mobile Link リダイレクターの削除**　リダイレクターは使用できなくなりました。代わりに Relay Server を使用します。「Relay Server の概要」『Relay Server』を参照してください。

- **スクリプトバージョンの使用の推奨事項**　ScriptVersion 拡張オプションを使用しないことを強くおすすめします。代わりに、CREATE SYNCHRONIZATION SUBSCRIPTION 文と ALTER SYNCHRONIZATION SUBSCRIPTION 文で SCRIPT VERSION 句を使用してください。「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』と「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **dbmlsync の -n オプションは廃止予定**　このオプションは廃止される予定です。代わりに -s dbmlsync オプションを使用することをおすすめします。「-s dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の -u オプションは廃止予定**　このオプションは廃止される予定です。代わりに -s dbmlsync オプションを使用することをおすすめします。「-s dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **Publication 同期プロファイルオプションは廃止予定**　このオプションは廃止される予定です。代わりに -s dbmlsync オプションを使用することをおすすめします。「-s dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **MLUser 同期プロファイルオプションは廃止予定**　このオプションは廃止される予定です。代わりに -s dbmlsync オプションを使用することをおすすめします。「-s dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync 統合コンポーネントの削除**　dbmlsync 統合コンポーネントは削除されました。代わりに、dbmlsync プログラミングインターフェイスを使用してください。「Dbmlsync API」『Mobile Link クライアント管理』を参照してください。

- **強制的な競合モードは廃止予定**　upload_insert、upload_update、upload_delete スクリプトがすべて未定義の場合、Mobile Link サーバーは強制的な競合解決を使用します。この機能は廃止される予定です。

- **upload_update による競合の検出は廃止予定**　upload_update スクリプトでアップロードされた更新の競合を検出して解決するか、upload_fetch または upload_fetch_column_conflict スクリプトを使用して競合を検出する必要があります。Mobile Link サーバーに依存して、

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

upload_update スクリプトによって影響を受けるローをカウントして競合を検出し、その後競合解決スクリプトを呼び出す方法は、廃止される予定です。「upload_update スクリプトによる競合の検出」『Mobile Link サーバー管理』を参照してください。

- **SQL スクリプトでの疑問符の使用**　Mobile Link SQL スクリプトでの単純な疑問符の使用は廃止される予定です。代わりに名前付きパラメーターを使用します。「名前付きスクリプトパラメーター」『Mobile Link サーバー管理』を参照してください。

QAnywhere の新機能

次に、バージョン 12.0.0 で導入された QAnywhere の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **新しいライトウェイトポーリングのサポート**　QAnywhere は、ライトウェイトポーリングを使用して Mobile Link サーバーから Push 通知を受け取れるようになりました。この新しい機能を使用するには、次の新しいオプションを使用します。
  - 「-push qaagent オプション」『QAnywhere』と「-push qauagent オプション」『QAnywhere』の新しい lwpoll モード。
  - PUSH QAnywhere Manager 設定プロパティの新しい lwpoll プロパティ。「QAnywhere Manager の設定プロパティ」『QAnywhere』を参照してください。

QAnywhere クライアント API 機能

- **.NET QAManagerBase クラスの ExceptionListener デリゲートのサポート**　このサポートにより、バックグラウンドスレッドでメッセージを受信している間に発生した QAExceptions をアプリケーションに通知できます。ExceptionListener デリゲート [QAnywhere .NET]『QAnywhere』と QAManagerBase.SetExceptionListener メソッド [QAnywhere .NET]『QAnywhere』を参照してください。

- **QAManagerBase クラス (.NET および Java) への新しい ReOpen メソッドの追加**　このメソッドは、ExceptionListener または MessageListener でメッセージストアデータベースへの接続を再確立するために呼び出されます。QAManagerBase.ReOpen メソッド [QAnywhere .NET]『QAnywhere』 (.NET) と QAManagerBase.reOpen メソッド [QAnywhere Java]『QAnywhere』(Java) を参照してください。

QAnywhere Agent の機能

- **QAnywhere Agent の -cd オプション**　このオプションを -cr オプションとともに使用して、データベースへの接続を試行する間隔を指定します。このオプションは、qaagent と qauagent に対して有効です。「-cd qaagent オプション」『QAnywhere』を参照してください。

- **QAnywhere Agent の -cr オプション**　このオプションを使用して、QAnywhere Agent がデータベースへの失敗した接続を再試行する回数を指定します。このオプションは、qaagent と qauagent に対して有効です。「-cr qaagent オプション」『QAnywhere』を参照してください。

QAnywhere の動作の変更と廃止予定機能

次に、バージョン 12.0.0 で導入された QAnywhere の動作の変更と廃止予定機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

SQL Remote の新機能

次に、バージョン 12.0.0 で導入された SQL Remote の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **SQL Remote で空間値のレプリケーションのサポート**　SQL Remote で空間データ型のレプリケーションがサポートされるようになりました。また、新しい TIMESTAMP WITH TIMEZONE データ型のレプリケーションもサポートされるようになりました。「TIMESTAMP WITH TIME ZONE データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **抽出ユーティリティ (dbxtract) への -g オプションの追加**　MANUAL REFRESH と定義されているマテリアライズドビューは、デフォルトで再ロード時に初期化されません。dbxtract で -g オプションを使用して、このようなマテリアライズドビューを再ロードの一部として初期化できるようになりました。「抽出ユーティリティ (dbxtract)」『SQL Remote』を参照してください。

- **パブリッシャーが未定義の場合の、SQL Remote によるエラーメッセージ出力**　リモートユーザーまたは統合ユーザーが定義されており、パブリッシャーが定義されていないデータベースに SQL Remote が接続すると、データベースにパブリッシャーが定義されていないことを示すエラーメッセージが SQL Remote から返されます。

Ultra Light の新機能

次に、バージョン 12.0.0 で導入された Ultra Light の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

一般的な機能

- **空間データのサポート**　Ultra Light の新しい空間データ機能をサポートするため、次の機能が追加されました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **ST_Geometry データ型**　ST_Geometry データ型では、空間値に適用できる関数がサポートされています。「ST_Geometry タイプ」『Ultra Light データベース管理とリファレンス』を参照してください。

- **空間データ関数**　Ultra Light には、空間データ処理をサポートするための関数がいくつか備えられています。「空間データ用の関数」『Ultra Light データベース管理とリファレンス』を参照してください。

- **新しい暗号化の例**　BlackBerry スマートフォンでの Ultra Light J 暗号化を説明するため、新しいコードサンプルが追加されました。詳細については、*%SQLANYSAMP12%¥UltraLiteJ* ディレクトリを参照してください。

プラットフォームとデバイス

iPhone と Mac OS X のサポート

Ultra Light C/C++ API. を使用して、Mac OS X と iPhone を対象にした Ultra Light アプリケーションを Mac OS X で開発できるようになりました。詳細については、「Ultra Light C++ アプリケーション開発」『Ultra Light C/C++ プログラミング』を参照してください。 また、iPhone アプリケーションの開発を支援する詳しいチュートリアルもあります。「チュートリアル：C++ API を使用した iPhone アプリケーションの構築」『Ultra Light C/C++ プログラミング』を参照してください。

64 ビット Linux へのインストール

SQL Anywhere を 64 ビットの Linux コンピューターにインストールする場合、32 ビットのサブシステムも別個にインストールする必要があります。これは、一部の 64 ビットの Linux オペレーティングシステムに、32 ビットの互換性ライブラリが事前にインストールされていないためです。32 ビットのクライアントソフトウェアを使用するには、Linux ディストリビューション用に 32 ビットの互換性ライブラリをインストールする必要があります。たとえば、Ubuntu では、次のコマンドの実行が必要となることがあります。

```
sudo apt-get install ia32-libs
```

セキュリティ

- **Ultra Light で FIPS 140-2 認定の暗号化をサポート**　64 ビットの Windows では、FIPS 140-2 認定の暗号化が Ultra Light 用にサポートされるようになりました。

- **Ultra Light データベース暗号化で 256 ビットの AES を使用**　Ultra Light データベース暗号化が、128 ビットの AES ではなく、256 ビットの AES を使用して実行されるようになりました。

- **パスワードのハッシングにソルトを使用**　新しいユーザーが作成されるか、または既存のユーザー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るようになりました。「Ultra Light PWD 接続パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

ユーティリティ

- **Ultra Light ユーティリティの改善**　ulinit ユーティリティは、SQL Anywhere データベースから抽出されたスキーマに Ultra Light でサポートされない要素 (カラムのデータ型やインスタンスのデフォルト値など) が含まれていても、SQL Anywhere データベースの情報に基づいて Ultra Light データベースを作成できるように改善されました。これは、デフォルトの動作になりました (-f オプションを使用してオフにすることが可能)。「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』を参照してください。 ulcreate ユーティリティは、サポートされなくなりました。

  ulinit、ulerase、ulinfo、ulload、ulsync の各ユーティリティにも変更が加えられました。

  次の項を参照してください。

  - 「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light ユーティリティ消去 (ulerase)」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light 情報ユーティリティ (ulinfo)」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light データベースへの XML のロードユーティリティ (ulload)」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light 同期ユーティリティ (ulsync)」『Ultra Light データベース管理とリファレンス』

SQL

- **CREATE／DROP／ALTER USER の追加**　次の項を参照してください。

  - 「CREATE USER 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP USER 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「ALTER USER 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **CREATE TABLE と ALTER TABLE の新しいテーブル制約**　CREATE TABLE 文または ALTER TABLE 文で、追加のテーブル制約 (SYNCHRONIZE ON | OFF | ALL) を指定できます。次の項を参照してください。

  - 「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「ALTER TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **DROP 文への IF EXISTS 句の追加**　DROP INDEX、DROP PUBLICATION、DROP SYNCHRONIZATION PROFILE、DROP TABLE の各文で、新しい IF EXISTS 句をオプションで指定できるようになりました。 次の項を参照してください。

  - 「DROP INDEX 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **CREATE 文への IF NOT EXISTS 句の追加**　CREATE TABLE、CREATE INDEX、CREATE PUBLICATION、CREATE SYNCHRONIZATION PROFILE の各文で、新しい IF NOT EXISTS 句をオプションで指定できるようになりました。次の項を参照してください。

  - 「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「CREATE INDEX 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「CREATE PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **CREATE SYNCHRONIZATION PROFILE では、OR REPLACE 句もサポートされるようになりました。**　CREATE SYNCHRONIZATION PROFILE 文では、テーブルを置き換えるためのオプションもサポートされるようになりました。「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **COUNT_UPLOAD_ROWS 関数の追加**　COUNT_UPLOAD_ROWS 関数では、次回の同期でアップロードされるロー数を問い合わせできます。「COUNT_UPLOAD_ROWS 関数 [集合]」『Ultra Light データベース管理とリファレンス』を参照してください。

データ型

- **ST_Geometry**　ST_Geometry データ型では、空間値に適用できる関数がサポートされています。「ST_Geometry タイプ」『Ultra Light データベース管理とリファレンス』を参照してください。

- **TIMESTAMP WITH TIMEZONE データ型**　TIMESTAMP WITH TIMEZONE データ型を使用すると、日付値と時刻値をタイムゾーンオフセットと一緒に格納できます。タイムゾーンオフセットは、UTC の前または後の分数です。「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』と「Ultra Light timestamp_with_time_zone_format 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

プログラミングインターフェイス

Ultra Light for C/C++

- **新しい Ultra Light C++ API の追加**　Ultra Light C++ API の新バージョンが、このリリースに追加されました。この API は、*ulcpp.h* ヘッダーファイルで宣言されます。「Ultra Light C/C++ API リファレンス」『Ultra Light C/C++ プログラミング』を参照してください。

このバージョンの Ultra Light C++ API で、次のオブジェクトの名前が変更されました。

- ul_synch_info は ul_sync_info に名前が変更されました。
- ul_synch_result は ul_sync_result に名前が変更されました。
- ul_synch_status は ul_sync_status に名前が変更されました。
- ul_synch_stats は ul_sync_stats に名前が変更されました。
- ul_synch_observer_fn は ul_sync_observer_fn に名前が変更されました。
- ul_synch_state は ul_sync_state に名前が変更されました。
- ULInitSynchInfo は ULInitSyncInfo に名前が変更されました。
- ULSetSynchInfo は ULSetSyncInfo に名前が変更されました。
- ULGetSynchResult は ULGetSyncResult に名前が変更されました。
- All UL_SYNCH_STATE オブジェクトは UL_SYNC_STATE オブジェクトに名前が変更されました。
- UL_SYNCH_STATUS_FLAG_IS_BLOCKING は UL_SYNC_STATUS_FLAG_IS_BLOCKING に名前が変更されました。
- ULRegisterErrorCallback は ULSetErrorCallback に名前が変更されました。
- ULRegisterSynchronizationCallback は ULSetSynchronizationCallback に名前が変更されました。

GetDatabaseID、SetDatabaseID、IsCaseSensitive、GetCollationName メソッドが、C++ API から削除されました。この機能は、GetDatabaseProperty と SetDatabaseOption で処理されるようになりました。ULConnection.GetDatabaseProperty メソッド [Ultra Light C++]『Ultra Light C/C++ プログラミング』と ULConnection.SetDatabaseOption メソッド [Ultra Light C++]『Ultra Light C/C++ プログラミング』を参照してください。

GetDatabasePropertyInt メソッドが追加されました。ULConnection.GetDatabasePropertyInt メソッド [Ultra Light C++]『Ultra Light C/C++ プログラミング』を参照してください。

Ultra Light.NET

- **新しい ULConnection.ValidateDatabase(ULDBValid) メソッドの追加**　tableName に NULL を渡して ULConnection.ValidateDatabase(ULDBValid, String) を呼び出すのと同じ機能として、このメソッドが追加されました。ULConnection.ValidateDatabase(ULDBValid) メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

Ultra Light for M-Business Anywhere

- **API 機能**　このリリースには、次のオブジェクトが追加されています。

  ○SyncParms クラスの setPublications メソッド。このメソッドは、同期させるパブリケーションを設定します。SyncParms.setPublications メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』を参照してください。

  ○SyncParms クラスの getPublications メソッド。このメソッドは、同期させるパブリケーションを返します。SyncParms.getPublications メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』を参照してください。

  ○CreationParms クラスの timestampWithTimeZoneFormat メンバー。このメンバーは、データベースから取り出したタイムゾーンでタイムスタンプのフォーマットを設定します。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

CreationParms.timestampWithTimeZoneFormat 変数 [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』を参照してください。

○PreparedStatement クラスの setTimestampWithTimeZoneParameter メソッドと ULTable クラスの setTimestampWithTimeZone メソッド。これらのメソッドは、指定された SQLType.TIMESTAMP_WITH_TIME_ZONE 型のパラメーターの値を、Date オブジェクトを使用して設定します。PreparedStatement.setTimestampWithTimeZoneParameter メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』と ULTable.setTimestampWithTimeZone メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』を参照してください。

○ResultSet と ULTable クラスの getTimestampWithTimeZone メソッド。これらのメソッドは、指定されたカラムの値を Date オブジェクトとして返します。ResultSet.getTimestampWithTimeZone メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』 (ResultSet クラス) と ULTable.getTimestampWithTimeZone メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』(ULTable クラス) を参照してください。

Ultra Light の動作の変更と廃止予定機能

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

次に、バージョン 12.0.0 で導入された Ultra Light の廃止予定機能と動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

廃止予定機能

- **ulcreate ユーティリティのサポート終了**　ulcreate ユーティリティは使用できなくなりました。すべての機能は ulinit で処理されるようになりました。「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』を参照してください。

- **SQL パススルーのサポート終了**　Ultra Light 11 でリリースされた SQL パススルーは、サポートされなくなりました。この機能は「リモートデータベースの集中管理」『Mobile Link サーバー管理』 で処理されるようになりました。

- **ALTER SYNCHRONIZATION PROFILE での OR REPLACE 句の使用のサポート終了**　**OR REPLACE** 句は、**ALTER SYNCHRONIZATION PROFILE** 文から削除されました。

- **Ultra Light ODBC API のサポート終了**　Ultra Light ODBC API がサポートされなくなりました。代わりに、Ultra Light C/C++ API を使用します。Ultra Light C/C++ プログラミングを参照してください。

廃止予定のプラットフォーム

- **Ultra Light for M-Business Anywhere のサポート終了**　M-Business Anywhere の Ultra Light サポートは、Ultra Light 12 で廃止されます。

- **Palm オペレーティングシステムのサポート終了**　Palm オペレーティングシステムは、Ultra Light 12 ではサポートされません。

削除、廃止、変更された API

- **Ultra Light C/C++ API の置き換え**　*uliface.h* ヘッダーファイルで定義される Ultra Light C/C++ API は、*ulcpp.h* ヘッダーファイルで定義される新バージョンに置き換えられました。以前のバージョンの API は、引き続き使用できます。廃止された Ultra Light C/C++ API の詳細については、http://dcx.sybase.com/1101ja/ulc_ja11/c-common-apiref.html を参照してください。

  Ultra Light C/C++ API の古い実装を使用するには、Ultra Light アプリケーションプロジェクトに *%SQLANY11%¥SDK¥C¥ulcpp11.cpp* ファイルを追加します。SQLANY11 は、SQL Anywhere のインストールディレクトリをポイントする環境変数です。

- **Ultra Light C/C++ API オブジェクトの変更**　次のオブジェクトは前回のリリースから変更され、新しい Ultra Light C/C++ API に適用されます。

  ○SQL パススルーは、API によってサポートされなくなりました。次のオブジェクトが削除されました。

  - ul_sql_passthrough_state
  - ul_sql_passthrough_status
  - ul_sql_passthrough_observer_fn

- **Ultra Light C/C++ 共通 API オブジェクトの変更**　前回のリリース以降、次のオブジェクトが変更されています。

  ○MLFileTransfer 関数は、MLFileDownload に名前が変更されました。MLFileDownload メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』を参照してください。

  ○force_transfer field of ml_file_transfer_info 構造体は削除されました。

  ○enable_resume field of ml_file_transfer_info のデフォルトは、false ではなく true になりました。MLFileDownload メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』を参照してください。

  ○ファイル転送の制御を強化するため、Mobile Link サーバーに渡される新しい remote_key field of ml_file_transfer_info が MLFileDownload でサポートされています。MLFileDownload メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』を参照してください。

- **Ultra Light 11.0 での埋め込み Visual C++ のサポート終了**　Visual Studio 2003 のサポートは、Ultra Light 11.0 で終了しました。したがって、埋め込み Visual C++ のサポートは、Visual Studio 2005 に移動されました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **埋め込み SQL API オブジェクトの変更**　前回のリリース以降、次のオブジェクトが変更されています。

  ○SQL パススルーは、API によってサポートされなくなりました。次のオブジェクトが削除されました。

  - ULGetSQLPassthroughScriptCount
  - ULExecuteNextSQLPassthroughScript
  - ULExecuteSQLPassthroughScripts
  - ULSetSQLPassthroughCallback

- **Ultra Light for M-Business Anywhere オブジェクトの変更**　前回のリリース以降、次のオブジェクトが変更されています。

  ○Connection クラスの GetSQLPassthroughScriptCount、ExecuteNextSQLPassthroughScript、ExecuteSQLPassthroughScripts メソッドが削除されました。

  ○DatabaseManager クラスの CreateDatabase メソッドの構文が変更されました。

- **Ultra Light .NET API オブジェクトの変更**　前回のリリース以降、次のオブジェクトが変更されています。

  ○ULConnection クラスの DatabaseManager プロパティが削除され、不要になりました。ULDatabaseManager は、シングルトンクラスではなく、静的になりました。ULDatabaseManager クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○ULConnectionParms クラスの DatabaseOnCE プロパティの名前が DatabaseOnDevice に変更されました。ULConnectionParms.DatabaseOnDevice プロパティ [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○ULTableSchema クラスの GetOptimalIndex メソッドで、最適なインデックスの名前が返されるようになりました。ULTableSchema.GetOptimalIndex メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○ULConnection クラスの CountUploadRows(String, UInt32) メソッドが削除されました。CountUploadRows(String, Int64) を代わりに使用します。ULConnection.CountUploadRows メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○SQL パススルーは、API によってサポートされなくなりました。次のオブジェクトが削除されました。

  - ULConnection.GetSQLPassthroughScriptCount
  - ULConnection.ExecuteNextSQLPassthroughScript
  - ULConnection.ExecuteSQLPassthroughScripts
  - ULSqlPassthroughProgressListener
  - ULSqlProgressData
  - ULSqlProgressState

  ○ULDatabaseSchema クラスの GetPublicationSchema メソッドと一緒に、ULPublicationSchema クラスとそのメソッドが削除されました。SYNC_ALL_DB フィールドと

SYNC_ALL_PUBS フィールドが、ULConnection クラスに移動されました。ULConnection.SYNC_ALL_DB フィールド [Ultra Light.NET]『Ultra Light .NET プログラミング』と ULConnection.SYNC_ALL_PUBS フィールド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

その他

- **ユーザーパブリケーション制限の増大** ユーザーパブリケーションの最大数は 63 に増加されました。
- **Ultra Light データベースのデフォルトのエンコードは UTF-8 エンコード** Ultra Light データベースは、UTF-8 エンコードがデフォルトになりました。「Ultra Light utf8_encoding 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

Ultra Light J の新機能

次に、バージョン 12.0.0 で導入された Ultra Light J の追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **外部 BLOB ファイルのサポート** Ultra Light J で BlackBerry OS または Java SE を使用している場合、データベースファイルの分割がサポートされ、データベース内の特定のカラムを使用して参照されるファイルとともに、外部ファイルを使用して大きな BLOB 値を格納できるようになります。これは、CREATE TABLE SQL 関数の一部として実装されます。

  BlackBerry アプリケーションでのサンプルの使用シナリオとしては、小規模で高速の永続ストアを使用して Ultra Light J データベースを格納し、大規模で低速のフラッシュドライブまたは SD カードを使用して写真などの大規模な BLOB 値を格納します。写真をキャプチャしてデータベースに格納するアプリケーションで、バッテリー電力と写真をデータベースにコピーする時間が無駄にならないという利点があります。

  「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **BlackBerry の内部フラッシュと SD カードのサポート** Ultra Light J では、Ultra Light J データベースと内部フラッシュメモリまたは SD カードとの間で読み込み／書き込みを行うことができます。ConfigFileME インターフェイス [BlackBerry] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **複数バージョンの Ultra Light J のサポート** Ultra Light J の複数のバージョンが BlackBerry と Java ME 環境で共存できるようにするため、Ultra Light J の JAR ファイル、COD ファイル、Java パッケージの名前にバージョン番号が次のように含まれるようになりました。

  ○ JAR ファイルは *UltraLiteJ12.jar* と呼ばれます。
  ○ COD ファイルは *UltraLiteJ12.cod* と呼ばれます。
  ○ すべてのパブリッククラスは、パッケージ com.ianywhere.ultralitej12 に含まれます。
  ○ Unsigned64 クラスは、パッケージ com.ianywhere.ultralitej12 に移動されました。

- **Ultra Light J アプリケーションの暗号化とセキュリティ保護** BlackBerry スマートフォン用にセキュリティ保護が非常に強化された Ultra Light J アプリケーションの作成方法を示す新し

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

いサンプルが追加されました。詳細については、*Samples¥UltraLite¥UltraLiteJ ¥BlackBerryEncryption¥ReadMe.html* を参照してください。BlackBerry セキュリティの詳細については、http://www.sybase.com/detail_list?id=9814 にあるホワイトペーパー『Ultra Light J Security on BlackBerry Devices』を参照してください。

- **新しい ST_Geometry データ型**　ST_Geometry データ型では、空間値に適用できる関数がサポートされています。「ST_Geometry タイプ」『Ultra Light データベース管理とリファレンス』を参照してください。

- **TIMESTAMP WITH TIMEZONE データ型**　TIMESTAMP WITH TIMEZONE データ型を使用すると、日付値と時刻値をタイムゾーンオフセットと一緒に格納できます。タイムゾーンオフセットは、UTC の前または後の分数です。「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』と「Ultra Light timestamp_with_time_zone_format 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

- **RAND SQL 関数のサポート**　Ultra Light J では、RAND SQL 関数をサポートしています。「RAND 関数 [数値]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **CREATE SYNCHRONIZATION PROFILE、ALTER SYNCHRONIZATION PROFILE、DROP SYNCHRONIZATION PROFILE、SYNCHRONIZE のサポート**　これらの文は、SQL を使用した同期パラメーターの編成と同期の起動を行うための代替手段を提供するためのものです。既存の Connection.createSyncParm()、Connection.synchronize(SyncParm)、および関連 API は、引き続き動作します。次の項を参照してください。

  - 「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「SYNCHRONIZE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **CREATE TABLE と ALTER TABLE の新しいテーブル制約**　CREATE TABLE 文または ALTER TABLE 文で、追加のテーブル制約 (SYNCHRONIZE ON | OFF | ALL) を指定できます。次の項を参照してください。

  - 「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「ALTER TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **IF EXISTS 句の追加**　DROP INDEX、DROP PUBLICATION、DROP SYNCHRONIZATION PROFILE、DROP TABLE の各文で、新しい IF EXISTS 句をオプションで指定できるようになりました。次の項を参照してください。

  - 「DROP INDEX 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「DROP TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **IF NOT EXISTS 句の追加**　CREATE TABLE、CREATE INDEX、CREATE PUBLICATION、CREATE SYNCHRONIZATION PROFILE の各文で、新しい IF NOT EXISTS 句をオプションで指定できるようになりました。次の項を参照してください。

  ○「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  ○「CREATE INDEX 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  ○「CREATE PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  ○「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』

- **CREATE SYNCHRONIZATION PROFILE では、OR REPLACE 句もサポートされるようになりました。**　CREATE SYNCHRONIZATION PROFILE 文では、テーブルを置き換えるためのオプションもサポートされるようになりました。「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Mobile Link 経由のファイル転送**　Ultra Light J では、Mobile Link サーバー経由で、リモートデータベースとの間でファイルのアップロードとダウンロードを行うことができます。

  Ultra Light J のデスクトップバージョンでは、Mobile Link からローカルファイルシステムにどのタイプのファイルでもダウンロードでき、ローカルファイルシステムから Mobile Link にどのタイプのファイルでもアップロードできます。

  Ultra Light J の BlackBerry OS バージョンでは、有効であり暗号化および難読化されていないデータベースファイルを Mobile Link からダウンロードしてオブジェクトストアに格納したり、これらのタイプのデータベースを Mobile Link にアップロードしたりできます。Mobile Link からメディアカードまたはフラッシュストレージにどのタイプのファイルでもダウンロードでき、フラッシュカードやメディアカードから Mobile Link にどのタイプのファイルでもアップロードできます。

  Ultra Light J のファイル転送の詳細については、FileTransfer インターフェイス [Ultra Light J] 『Ultra Light - Java プログラミング』を参照してください。

- **Ultra Light J データベース転送ユーティリティで、BlackBerry のファイルシステムでデータベースを開いたり転送したりできる**　Ultra Light J では、データベース名に基づいてデータベースの場所 (オブジェクトストアまたはファイルシステム) が自動的に判断されます。名前の先頭が *file://* (大文字と小文字が区別される) である場合はデータベースがファイルシステムで検索され、それ以外の場合はデータベースがオブジェクトストアで検索されます。

- **BlackBerry インストールディレクトリ名の変更**　BlackBerry スマートフォンファイルのインストールディレクトリは、BlackBerry の最小 OS バージョンを使用するように名前が変更されました。*UltraLite¥UltraLiteJ¥Java MERIM11* ディレクトリは *UltraLite¥UltraLiteJ ¥BlackBerry4.2* になり、BlackBerry OS 4.2 以降と互換性のあるファイルであることを示します。

- **ULjLoad と ULjUnload での blobfile タイプのサポート**　Ultra Light J のロードユーティリティとアンロードユーティリティでは、blobfile タイプのカスタム実装をサポートしています。blobfile タイプの Ultra Light J データベースのサンプル実装については、「Ultra Light Java Edition データベースアンロードユーティリティ (ULjUnload)」『Ultra Light データベース管理とリファレンス』と「Ultra Light Java Edition データベースロードユーティリティ (ULjLoad)」『Ultra Light データベース管理とリファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **ロー制限アルゴリズムの改善**　カラムが多いテーブルのローが、カラムが少ないテーブルのローより多くのリソースを使用できるようにロー制限アルゴリズムが改善されました。
- **システムテーブルの変更**　Ultra Light J システムテーブルに、次の変更が加えられました。
  - **syscolumn システムテーブルの column_default_value カラムで、DEFAULT AUTOFILENAME デフォルト句がサポートされる**　このカラムでは、カラムデフォルト値 DEFAULT AUTOFILENAME で指定された VARCHAR 型を処理できます。「syscolumn システムテーブル」『Ultra Light データベース管理とリファレンス』を参照してください。
  - **syscolumn システムテーブルへの filename_colid カラムの追加**　スキーマ定義の file_name カラムが参照される場合は、このカラムに参照先カラムのカラム ID が格納され、そうでない場合は、このカラムが NULL になります。「syscolumn システムテーブル」『Ultra Light データベース管理とリファレンス』を参照してください。
  - **systable システムテーブルへの table_partition_size カラムの追加**　このカラムには、定義された分割サイズの値が格納されます。「systable システムテーブル」『Ultra Light データベース管理とリファレンス』を参照してください。
- **暗号化のパフォーマンスの変更**　低速の CPU 環境でパフォーマンスを向上するため、EncryptionControl インターフェイスが次のように改善されました。
  - Ultra Light J で、データとシステムにとって重要なページのみが暗号化されるようになりました。
  - 復号化メソッドで、復号化する必要のあるバイト数を指定するため、追加パラメーターを指定できるようになりました。EncryptionControl.decrypt メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

  EncryptionControl インターフェイスの詳細については、EncryptionControl インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

Ultra Light J API の新機能

- **API 機能**　このリリースには、次のオブジェクトが追加されています。
  - ConfigFileMe インターフェイスが追加され、Java ME デバイスファイルシステム上のファイルに保存される永続データベースを設定できるようになりました。ConfigFileME インターフェイス [BlackBerry] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。
  - ConfigPersistent インターフェイスの setRowScoreFlushSize メソッドと setRowScoreMaximum メソッドが追加され、setRowMinimumThreshold メソッドと setRowMaximumThreshold メソッドに置き換えられました。これらのメソッドは、新しいロー制限アルゴリズムに対応しています。ロー制限アルゴリズムによって、カラムが多いテーブルのリソース管理が改善されます。ConfigPersistent.setRowScoreFlushSize メソッド [Ultra Light J]『Ultra Light - Java プログラミング』と ConfigPersistent.setRowScoreMaximum メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

    ユニークな識別子を記述するため、UUIDValue インターフェイスが追加されました。UUIDValue インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○現在の接続で DEFAULT AUTOINCREMENT または DEFAULT GLOBAL AUTOINCREMENT カラムに挿入された最新の値を取り出すため、Connection インターフェイスに getLastIdentity メソッドが追加されました。Connection.getLastIdentity メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○接続用に現在登録されている SyncObserver オブジェクトを返すため、Connection インターフェイスに getSyncObserver メソッドが追加されました。Connection.getSyncObserver メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○接続で最後に実行された SYNCHRONIZE SQL 文の結果を返すため、Connection インターフェイスに getSyncResult メソッドが追加されました。Connection.getSyncResult メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○接続で同期の進行状況をモニターする SyncObserver オブジェクトを設定するため、Connection インターフェイスに setSyncObserver メソッドが追加されました。Connection.setSyncObserver メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○データベースファイルの転送に使用できる FileTransfer オブジェクトを作成するため、DatabaseManager インターフェイスに createFileTransfer メソッドと createObjectStoreTransfer メソッドが追加されました。DatabaseManager.createFileTransfer メソッド [Ultra Light J]『Ultra Light - Java プログラミング』と DatabaseManager.createObjectStoreTransfer メソッド [BlackBerry] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○ファイル転送オプションを指定するメソッドを提供するため、FileTransfer インターフェイスが追加されました。FileTransfer インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○observer コールバックにデータを渡すことができるようにするため、FileTransferProgressData インターフェイスが追加されました。FileTransferProgressData インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○ファイル転送 observer として FileTransferProgressListener インターフェイスが追加されました。 FileTransferProgressListener インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○クエリプランをコンピューターのモニターに表示したり印刷したりする場合の読みやすさを向上させるために、PreparedStatement インターフェイスに getPlanTree メソッドが追加されました。PreparedStatement.getPlanTree メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○シャドウページングがオンになったことを検出するために、ConfigPersistent インターフェイスに hasShadowPaging メソッドが追加されました。ConfigPersistent.hasShadowPaging メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Ultra Light J の動作の変更と廃止予定機能

次に、バージョン 12.0.0 で導入された Ultra Light J の廃止予定機能と動作の変更を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

削除、廃止予定、変更されたユーティリティオプション

- **ULjInfo、ULjLoad、ULjUnload の -p オプションは省略可能**　**-p** が指定されない場合は、**dba** がデフォルトのパスワードとして使用されます。

廃止予定のプラットフォーム

- **Ultra Light J での BlackBerry OS 4.1 のサポート終了**　Ultra Light J で、BlackBerry OS 4.2 以降をサポートするようになりました。

削除、廃止、変更された API

- **Ultra Light J API オブジェクトの変更**　前回のリリース以降、次のオブジェクトが変更されています。

  ○ianywhere.ultralitej パッケージの名前が変更されました。Ultra Light J の複数のバージョンが BlackBerry と Java ME 環境で共存できるようにするため、Ultra Light J の JAR ファイル、COD ファイルのパッケージ名にバージョン番号が含まれるようになりました。「Ultra Light J API リファレンス」『Ultra Light - Java プログラミング』を参照してください。

  ○Ultra Light J の syscolumn システムテーブルから table_autoinc カラムが削除されました。「syscolumn システムテーブル」『Ultra Light データベース管理とリファレンス』を参照してください。

  ○CollectionOfValueReaders インターフェイスが削除されました。すべての CollectionOfValueReader メソッドが ResultSet インターフェイスに移動されました。ResultSet インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

  ○CollectionOfValueWriters インターフェイスが削除されました。すべての CollectionOfValueWriter メソッドが PreparedStatement インターフェイスに移動されました。PreparedStatement インターフェイス [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

  ○ConfigPersistent methods setRowMaximumThreshold と setRowMinimumThreshold が、setRowScoreFlushSize と setRowScoreMaximum に置き換えられました。ConfigPersistent.setRowScoreFlushSize メソッド [Ultra Light J]『Ultra Light - Java プログラミング』と ConfigPersistent.setRowScoreMaximum メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

  ○スキーマ関連のすべてのメソッドとインターフェイスが削除されました。ForeignKeySchema インターフェイスと、ColumnSchema、IndexSchema、TableSchema イン

ターフェイスのすべてメソッドが削除されました。次のメソッドが Connection インターフェイスから削除されました。

- createPublication
- createTable
- dropForeignKey
- dropPublication
- dropTable
- enableSynchronization
- renameTable
- schemaCreateBegin
- schemaCreateComplete
- setNeverSynchronized
- startSynchronizationDelete
- stopSynchronizationDelete
- truncateTable

スキーマ操作を実行するには、Ultra Light J でサポートされている SQL 文 (CREATE TABLE や CREATE PUBLICATION など) を使用します。「Ultra Light SQL 文」『Ultra Light データベース管理とリファレンス』を参照してください。

○Connection インターフェイスの setDatabaseId と getDatabasePartitionSize メソッドが削除されました。デフォルトの分割サイズは、指定できなくなりました。デフォルトの分割サイズを上書きするには、DEFAULT GLOBAL AUTOINCREMENT 文を使用します。「Ultra Light での GLOBAL AUTOINCREMENT の使用」『Ultra Light データベース管理とリファレンス』を参照してください。

○Connection インターフェイスの getDatabaseID メソッドは、Connection.getOption(OPTION_DATABASE_ID) の呼び出しと同じ効果になりました。Connection.getOption メソッド [BlackBerry] [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

○Value、ValueReader、ValueWriter クラスは削除されました。

その他

- **ユーザーパブリケーション制限の増大**　ユーザーパブリケーションの最大数は 63 に増加されました。

管理ツールの新機能

次に、バージョン 12.0.0 で導入された管理ツールの追加機能を示します。サポートされているプラットフォームとバージョンについては、http://www.ianywhere.jp/tech/1061806-os_components.html を参照してください。

- **[接続] ウィンドウ**　バージョン 12.0.0 では、**[接続]** ウィンドウのレイアウトが簡易化され、**[接続アシスタント]** は削除されました。以前は、目的の接続タイプに必要なオプションを把

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

握しておく必要がありました。現在は、選択した接続タイプに応じたオプションが **[接続]** ウィンドウに表示されます。

警告

○**[アクション]** ドロップダウンリストから、次のいずれかの接続タイプを選択してください。

[このコンピューターで稼働しているデータベースに接続]　このコンピューター上ですでに稼働しているデータベースに接続します。

[ODBC データソースを使用した接続]　ODBC データソースを使用してデータベースに接続します。

[別のコンピューターで稼働しているデータベースに接続]　ネットワーク内の別のコンピューター上ですでに稼働しているデータベースに接続します。

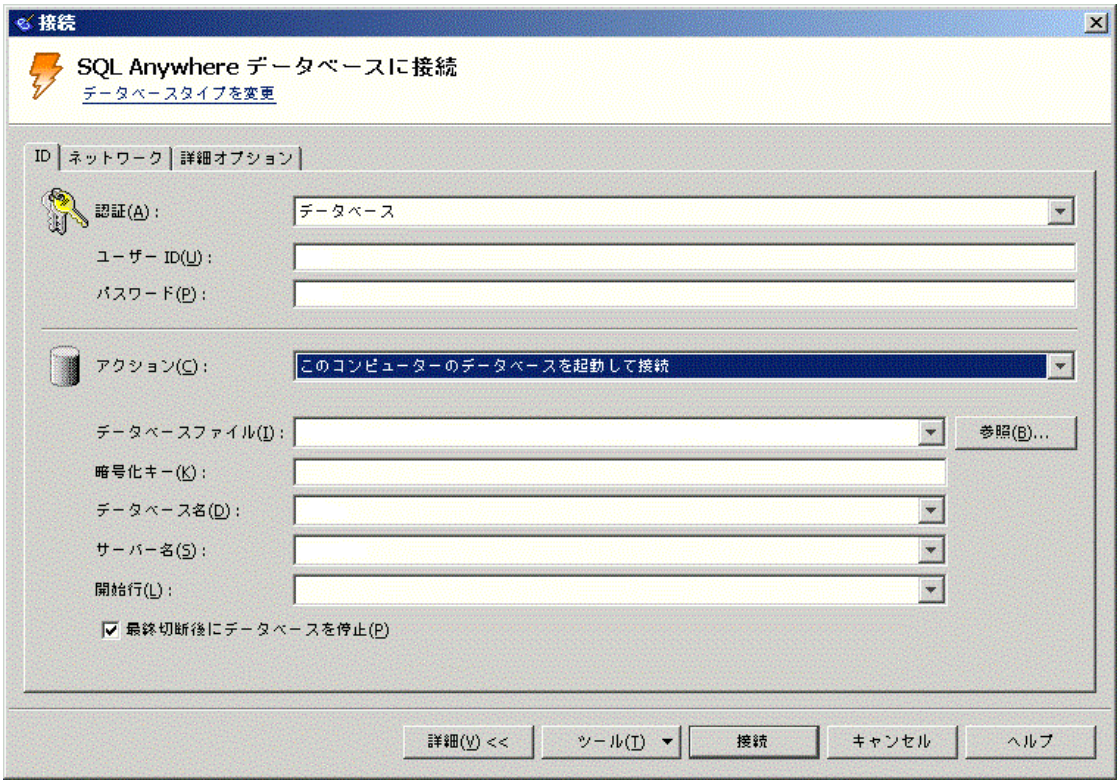
[このコンピューターのデータベースを起動して接続]　このコンピューターでデータベースを開始して、接続します。

[別のコンピューターのデータベースを起動して接続]　ネットワークを経由して別のコンピューター上でデータベースを開始し、接続します。

[接続文字列を使用して接続]　接続文字列を使用してデータベースに接続します。

[アクション] ドロップダウンリストの下に表示されるオプションは、選択内容によって異なります。

○Interactive SQL では、**[SQL Anywhere データベースに接続]** 見出しの下にある **[データベースタイプを変更]** をクリックして、接続先となるデータベースのタイプを変更します。



たとえば、ODBC データソースがあるデータベースに接続している場合、**[接続]** ウィンドウには、**[ODBC データソース名]** と **[ODBC データソースファイル]** の 2 つのオプションのみが表示されます。

必要に応じて **[詳細]** をクリックして、TCP/IP や暗号化オプションと、その他の詳細オプションを指定できます。「[接続] ウィンドウを開く (Sybase Central の場合)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ソフトウェア更新をチェックする新しいデフォルト** Interactive SQL、Sybase Central、SQL Anywhere コンソールユーティリティ、Mobile Link モニターで、デフォルトにより更新が毎日チェックされるようになりました。ソフトウェアの以前のバージョンのデフォルトでは、チェックされませんでした。「ソフトウェアの更新」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Interactive SQL と Sybase Central でのイメージデータ、HTML データ、XML データの表示** 結果セットで、イメージデータ、HTML データ、XML データをプレビューできます。「Interactive SQL とグラフィックイメージ」『SQL Anywhere サーバー データベース管理』と「Interactive SQL での HTML データと XML データの表示」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Sybase Central と Interactive SQL 用の新しいシステムトレイアイコン** Sybase Central または Interactive SQL の高速ランチャーオプションを有効にすると、システムトレイに新しいアイコンが表示されるようになりました。アプリケーションを開くには、アイコンを右クリッ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

クして **[開く]** をクリックし、アプリケーションを閉じて (実行中の場合) 高速ランチャープロセスを終了するには、**[終了]** をクリックします。「高速ランチャーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **アクセシビリティ有効化オプション**　アクセシビリティ有効化オプションが、デフォルトでインストールされるようになりました。以前は、このオプションを別個にインストールする必要がありました。アクセシビリティ有効化オプションを使用すると、Interactive SQL、Sybase Central、SQL Anywhere コンソールユーティリティ、Mobile Link モニターを画面リーダーなどのアクセシビリティ補助で操作することができます。「アクセシビリティ有効化オプション」『SQL Anywhere 12 紹介』を参照してください。

Sybase Central プラグインの新機能

次に、バージョン 12.0.0 で導入された Sybase Central のプラグインの追加機能を示します。

SQL Anywhere プラグインの新機能

- **ウィザードによって作成された SQL 文とユーティリティコマンドの表示**　データベースオブジェクトを作成したりデータベースユーティリティを実行したりするほとんどの SQL Anywhere プラグインウィザードの最後には、新しいページが表示されます。このページには、**[完了]** をクリックすると実行される SQL 文とユーティリティコマンドが表示されます。**[完了]** をクリックすると、SQL 文かユーティリティコマンドまたはその両方が実行されます。

  または、SQL 文をクリップボードにコピーし、**[キャンセル]** をクリックしてウィザードを終了した後、Interactive SQL を使用して SQL 文を実行することもできます。この機能を使用すると、データベースを変更することなくウィザードを使用して SQL スクリプトを生成できます。「ウィザードによって生成された SQL 文とユーティリティコマンドの表示」『SQL Anywhere サーバー データベース管理』を参照してください。

- **空間データのサポート**　空間データに関連する、ウィザードを含めたプラグインの新機能の詳細については、空間データのサポート 45 ページを参照してください。

- **新しいデータベース[断片化] タブ**　ベーステーブルとインデックスの断片化をグラフィックで表示できます。**[断片化]** タブには、ベーステーブルで sa_table_fragmentation システムプロシージャーを実行した結果がグラフィックで表示されます。「[断片化] タブ (SQL Anywhere プラグインの場合)」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **データベース変更のロギング**　SQL Anywhere プラグインによって実行された、データベースを変更するすべての SQL 文を記録できます。ログは、*.sql* ファイルに保存されます。「データベース変更のロギング」『SQL Anywhere サーバー データベース管理』を参照してください。

- **WITH NULLS NOT DISTINCT 句、CREATE INDEX 文のサポート**　SQL Anywhere プラグインを使用して、WITH NULLS NOT DISTINCT 句を使用するインデックスを作成、表示、変更することができます。[インデックス] フォルダーには、**[NULL 排他]** カラムがあります。インデックスがプライマリキーインデックス、外部キーインデックス、一意性制約インデックス、または非ユニークインデックスの場合、カラム値は空白になります。

  「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』の UNIQUE 句の説明を参照してください。

- **単語区切り**　SQL Anywhere プラグインでは、テキスト設定オブジェクトの外部単語区切りと事前フィルターがサポートされています。「ALTER TEXT CONFIGURATION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **データベースドキュメントの改善**　**データベースドキュメントウィザード**を使用すると、生成されたドキュメントにテーブルに関する情報が含まれます。テーブル情報には、所有者名、テーブルを変更するプロシージャー、カラム情報、コメントなどがあります。「データベースのドキュメント化」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[タイプフィルターの設定] ウィンドウの改善**　**[タイプフィルターの設定]** ウィンドウでは、Sybase Central の左ウィンドウ枠にあるフォルダーリストに表示するデータベースオブジェクトを指定できます。独自の指定をデフォルトのタイプフィルターとして設定できます。Sybase Central では、タイプフィルターが指定されていないデータベースに接続するたびに、デフォルトのタイプフィルターが使用されます。「Sybase Central の左ウィンドウ枠の設定」『SQL Anywhere サーバー データベース管理』を参照してください。

- **メンテナンスプラン作成ウィザードの改善**　メンテナンスプランには、ユーザー定義の操作が含まれています。**メンテナンスプラン作成ウィザード**で、ユーザー定義の操作を、検証前またはバックアップ後に実行する SQL 文として追加できます。

  また、**[メンテナンスプラン]** フォルダーに **[ステータス]** カラムが含まれるようになりました。「メンテナンスプランの作成」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ファンクション作成ウィザードの改善**　以前は、**ファンクション作成ウィザード**で戻り値のタイプが組み込みタイプに制限されていましたが、組み込みタイプまたはドメインのいずれかを選択できるようになりました。「ユーザー定義関数の作成」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **データベースバックアップウィザードの改善**　**データベースバックアップウィザード**で、空きページの削除がデフォルトで有効になるようになりました。「データベースバックアップウィザードを使用した、アーカイブバックアップの作成」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[概要] タブの改善**　データベースの **[概要]** タブには、コピーサーバーの正常性と統計情報がデータベースミラーリング情報と一緒に表示されます。「データベースの正常性と統計情報」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[同期サブスクリプションのプロパティ] ウィンドウの改善**　SQL Anywhere 12 以降で作成されたデータベースでは、同期サブスクリプションの名前が、**[同期サブスクリプションのプロパティ]** ウィンドウと **[同期サブスクリプション]** タブの両方に表示されます。以前のバージョンの SQL Anywhere で作成されたデータベースでは、名前が **(unnamed)** として表示されます。「同期サブスクリプションの作成」『Mobile Link クライアント管理』を参照してください。

- **新しい [同期プロファイルを使用して同期] ウィンドウ**　同期プロファイルを右クリックし、**[同期]** をクリックしてから **[OK]** をクリックすると、SQL Anywhere データベースを同期できます。「Mobile Link 同期プロファイル」『Mobile Link クライアント管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Sybase Central の動作の変更と廃止予定機能

次に、バージョン 12.0.0 で導入された Sybase Central の変更点を示します。

- **Sybase Central 6.1.0 へのバージョン変更**　SQL Anywhere 12 には、バージョン 6.1.0 の Sybase Central が含まれています。

- **Sybase Central は、SQL Anywhere バージョン 10x 以降のデータベースのみサポート**　バージョン 9 のデータベースサーバーと、バージョン 9 のソフトウェアで作成されたデータベースのサポートが SQL Anywhere プラグインから削除されました。データベースをアンロードして、再ロードファイル、新しいデータベース、または既存のデータベースに再ロードする場合、バージョン 9 以降のデータベースサーバーで稼働している、バージョン 5、6、7、8 または 9 のソフトウェアで作成したデータベースには引き続き接続できます。

  バージョン 6、7、8、9 データベースサーバーで動作するバージョン 5、6、7、8、9 ソフトウェアで作成されたデータベースの場合、一時的に SQL Anywhere プラグインからデータベースに接続して、次のいずれかのタスクを実行できます。

  ○データベースを再ロードファイルにアンロードします。

  ○データベースをアンロードして、新しいバージョン 12 データベースに再ロードします。

  ○データベースをアンロードして、既存のバージョン 12 データベースに再ロードします。

  アンロードされるデータベースのファイルはローカルのコンピューターに置いておく必要があります。

  バージョン 5 以前のサーバーで動作するバージョン 4 以前のソフトウェアによって作成されたデータベースの SQL Anywhere プラグインはサポートされません。

  「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **32 ビットと 64 ビットのコンピューターがサポートされています。**　Sybase Central の設定ファイルの名前が変更されました。

  ○32 ビットのコンピューターでは、*.scRepository600* ファイルの名前が *.scRepository610_32* に変更されました。

  ○64 ビットのコンピューターでは、インストール環境に応じて、*.scRepository600* ファイルの名前を *.scRepository610_32* または *.scRepository610_64* に変更できます。

- **Sybase Central を配備するときに、プラグイン用の JPR ファイルを作成する必要がなくなった**　以前は、Sybase Central と SQL Anywhere を配備するときに、プラグインごとに JPR ファイルを作成する必要がありましたが、Sybase Central で環境変数 *%SQLANY12%* を使用してプラグインの検出と登録が行われるようになりました。

SQL Anywhere プラグインの変更機能

- **メンテナンスプランの強化**　メンテナンスプランが次のように強化されています。

  ○ 検証前に実行される SQL 文を含めることができる。
  ○ バックアップ後に実行される SQL 文を含めることができる。
  ○ メンテナンスプランの実行中にステータスを表示できる。
  ○ 一度に実行できるメンテナンスプランは 1 つのみに限定される。

  詳細については、「メンテナンスプランの作成」『SQL Anywhere サーバー データベース管理』を参照してください。

- **動的オブジェクトとプロパティが自動的にリフレッシュされます。**　イベントの動的プロパティ、メンテナンスプラン、Windows サービスに加え、接続とロックのリストがデフォルトにより 10 秒ごとに自動的にリフレッシュされます。「再表示頻度の設定」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベース作成ウィザードの新しいデフォルト**　**データベース作成ウィザード**を使用して新しいバージョン 12 のデータベースを作成する場合、グローバルチェックサムがデフォルトで有効になりました。ウィザードを使用してバージョン 11 以降のデータベースを作成する場合、グローバルチェックサムはデフォルトで無効になります。Windows Mobile 用のデータベースの作成では、グローバルチェックサムはデフォルトで常に有効になります。新しいデータベースでチェックサムがデフォルトで有効 73 ページを参照してください。

Interactive SQL の新機能

次に、バージョン 12.0.0 で導入された Interactive SQL の追加機能を示します。

- **[空間プレビュー] と [空間ビューアー] を使用した空間データの表示**　「空間データのイメージとしての表示 (Interactive SQL の場合)」『SQL Anywhere サーバー 空間データサポート』を参照してください。

- **COMMIT 文と ROLLBACK 文の新しい実行方法**　Interactive SQL で、COMMIT 文を実行するには **[SQL]** » **[コミット]** をクリックし、ROLLBACK 文を実行するには **[SQL]** » **[ロールバック]** をクリックします。また、キーボードショートカットも使用できます。COMMIT 文を実行するには [Ctrl + Shift + C] キーを使用し、ROLLBACK 文を実行するには [Ctrl+Shift+R] キーを使用します。

  [SQL] メニューまたはキーボードショートカットを使用して COMMIT または ROLLBACK を実行した場合、**[SQL 文]** ウィンドウ枠の内容は変更されません。ただし、**[結果]** ウィンドウ枠の **[結果]** タブの内容はクリアされます。「Interactive SQL での COMMIT 文と ROLLBACK 文の実行」『SQL Anywhere サーバー データベース管理』を参照してください。

- **結果セットからのカラム、ロー、セルの選択とコピー方法に加えられた変更**　Interactive SQL の **[結果]** ウィンドウ枠で、結果セットから複数のカラム、ロー、セルを選択してコピーできます。たとえば、複数のカラムを選択するには、[Ctrl] キーを押しながらコピーするカラム内のセルをクリックし、右クリックして **[データのコピー]** » **[カラム]** を選択します。「Interactive SQL の結果セットからカラム、ロー、セルをコピーする」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **OEM ユーザーがパスワードをお気に入りに保存できなくする**　OEM 配備で、Interactive SQL のお気に入りの接続にユーザーがパスワードを保存することを禁止できるようになりました。「管理ツールの設定」『SQL Anywhere サーバー プログラミング』の allowPasswordsInFavorites オプションを参照してください。

- **お気に入りの編集、インポート、エクスポート**　Interactive SQL のお気に入りを編集、エクスポート、インポートできるようになりました。「SQL スクリプトファイルと接続のお気に入りリストへの保存 」『SQL Anywhere サーバー データベース管理』と「お気に入りの SQL スクリプトファイルと接続の共有」『SQL Anywhere サーバー データベース管理』を参照してください。

- **実行時間**　Interactive SQL のステータスバーに、現在の SQL 文の実行時間が表示されます。

- **[結果] ウィンドウ枠の変更**

  - **結果がテキストとして、またはスクロールテーブルに表示されます。**　以前は、**[オプション]** ウィンドウの設定を変更して、**[結果]** ウィンドウ枠の結果セットの表示のみが設定可能でした。**[データ]** » **[結果をスクロール可能なテーブルとして表示]** をクリックして、結果セットをスクロール可能なテーブルに表示できるようになりました。また、**[データ]** » **[結果をテキストとして表示]** をクリックすることによって、結果セットをテキストとして表示することもできます。変更内容を有効にするには、文を実行する必要があります。「Interactive SQL のカスタマイズ」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **Interactive SQL でのカラムのサイズ設定**　結果セットを右クリックし、ウィンドウまたはデータに合わせてカラムのサイズを設定するかどうかを選択できます。「Interactive SQL のカスタマイズ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL 関数のヘルプ**　**[SQL 文]** ウィンドウ枠で、SQL 関数を右クリックし、**[*function-name* のヘルプ]** をクリックすると、関数の説明が表示されます。

- **Interactive SQL で警告メッセージを非表示にする**　Interactive SQL に表示される警告メッセージの一部を無効にすることができます。たとえば、**[SQL 文]** ウィンドウ枠でテキストを保存しない場合に表示される警告を非表示にすることができます。

  [ツール] » **[オプション]** » **[メッセージ]** をクリックし、**[オプションのメッセージ]** リストのチェックボックスをオフにすることで、警告メッセージを無効にすることができます。「Interactive SQL のカスタマイズ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Interactive SQL でのファイルのリカバリ**　Interactive SQL が予期せず終了した場合は、*.sql* ファイルに保存されていない変更内容をリカバリしようとします。ファイルを編集すると、最後の変更から 30 秒が経過し、文の実行前に、Interactive SQL でバックアップコピーが作成されます。

Interactive SQL の動作の変更と廃止予定機能

次に、バージョン 12.0.0 で導入された Interactive SQL の変更点を示します。

- **32 ビットと 64 ビットのコンピューターがサポートされています。**

  ○Interactive SQL の 32 ビットバージョンと 64 ビットバージョンの両方を同じコンピューターにインストールできるようになりました。

- **Windows のデフォルトエンコードの変更**　次の変更は、たとえば、英語版の Windows XP コンピューターで ANSI エンコードと OEM エンコードが異なる場合に、Windows コンピューターでコンソールアプリケーションとして稼働している Interactive SQL (ウィンドウベースのユーザーインターフェイスは使用しない) に適用されます。

  1. 以前は、Interactive SQL をコンソールアプリケーションとして実行する場合、ENCODING 句が明示的に指定されていないと、INPUT 文と READ 文では OEM エンコード (英語版の Windows XP コンピューターの cp437) を使用してファイルがエンコードされていると見なされました。同様に、OUTPUT 文では OEM エンコードを使用してファイルが出力されました。

     Interactive SQL をコンソールアプリケーションとして実行する場合、INPUT 文と READ 文では、ANSI エンコード (英語版の Windows XP コンピューターの cp1252) を使用してファイルがエンコードされていると見なされるようになりました。同様に、OUTPUT 文では ANSI エンコードを使用してファイルが出力されます。

     OEM エンコードを使用するファイルをコマンドプロンプトから処理するには、エンコードを明示的に指定する必要があります。次に例を示します。

     ```
     dbisql READ ENCODING 'cp437' myfile.sql
     ```

  2. 以前は、Interactive SQL をコンソールアプリケーションとして実行した場合、ANSI エンコード (英語版の Windows XP コンピューターの cp1252) を使用するコマンドプロンプトとの間で結果の書き込み／読み込みが行われていたため、拡張文字が正しく表示されないことがありました。

     Interactive SQL をコンソールアプリケーションとして実行した場合、OEM エンコード (英語版の Windows XP コンピューターの cp437) を使用するコマンドプロンプトとの間で結果の書き込み／読み込みが行われるようになりました。

  「default_isql_encoding オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』を参照してください。

- **CLEAR 文、[クリア] メニュー項目、[Esc] キーの変更**　CLEAR 文を実行すると、開いているすべての結果セットが閉じ、**[SQL 文]** ウィンドウ枠の内容はそのままとなります。「CLEAR 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  また、**[編集]** » **[結果を閉じる]** メニュー項目を使用することは、CLEAR 文を実行することと同じであり、開いているすべての結果セットが閉じ、SQL 文のウィンドウ枠の内容はそのままとなります。

  SQL 文のウィンドウ枠の内容をクリアするために使用する **[編集]** » **[クリア]** メニュー項目は削除されました。その結果、**[クリア]** メニュー項目のキーボードショートカットである [Esc] キーも削除されました。デフォルトでは、[Esc] キーを押しても、何も起こらなくなりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ただし、**[SQL 文]** ウィンドウ枠の内容をクリアし、開いている結果セットを閉じるように [Esc] キーを設定できます。**[ツール]** » **[オプション]** » **[互換性]** をクリックし、**[Esc キーを押すと、SQL 文がクリアされ、結果セットが閉じます。]** をクリックします。

- **-codepage オプションが削除された**　Interactive SQL に特定のコードページを含むファイルを読み込ませる場合、INPUT 文、OUTPUT 文、または READ 文の ENCODING 句を使用してください。-codepage オプションはソフトウェアから削除されました。次の項を参照してください。

  ○「Interactive SQL ユーティリティ (dbisql)」『SQL Anywhere サーバー データベース管理』
  ○「Ultra Light 用 Interactive SQL ユーティリティ (dbisql)」『Ultra Light データベース管理とリファレンス』
  ○「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』
  ○「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』
  ○「READ 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』

- **SET OPTION 文 [Interactive SQL] の変更**　以前は、オプションを設定するための SET OPTION 文で値を指定しなかった場合、オプションは Off に設定されました。オプション値を省略した場合に、指定したオプションがデフォルト値に設定されるようになりました。この変更は、auto_commit、auto_refetch、bell、commit_on_exit、echo の各オプションに適用されます。「SET OPTION 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **OUTPUT 文の変更**　結果を TEXT ファイルに出力する場合、WITH COLUMN NAMES 句を使用して、カラム名をファイルの先頭に挿入できます。「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **INPUT 文の変更**　INPUT 文で TEXT ファイルから行を挿入する場合、SKIP 句を使用して、ファイルの先頭から指定の行数を省略できるようになりました。「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **READ キーワードが不要になった**　コマンドプロンプトから Interactive SQL を実行する場合、実行する *.sql* ファイルを指定するときに READ キーワードがオプションになりました。*.sql* ファイルにパラメーターが必要な場合は、ファイル名の後にパラメーターを指定します。

  たとえば、次の 2 つのコマンドは同じです。

READ キーワードを指定する	READ キーワードを指定しない
READ file.sql parm1	file.sql parm1

  「Interactive SQL ユーティリティ (dbisql)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Microsoft Excel ODBC ドライバーのサポートの改善**　次に、Microsoft Excel ODBC ドライバーを使用してデータを SQL Anywhere から Excel ファイルにエクスポートする場合の変更点を示します。

  ○以前は、CHAR、LONG VARCHAR、NCHAR、NVARCHAR、LONG NVARCHAR データ型として格納されているデータはエクスポートできませんでした。

Microsoft Excel ODBC ドライバーを使用して、CHAR、LONG VARCHAR、NCHAR、NVARCHAR、または LONG NVARCHAR データ型として格納されているデータを SQL Anywhere データベースからエクスポートする場合、データは VARCHAR (Excel ドライバーによってサポートされる最も近い型) として格納されます。

Microsoft Excel ODBC ドライバーがサポートするテキストカラムの幅は、255 文字までです。

○REAL、FLOAT、BIGINT データ型として格納されているデータは、エクスポートできます。

○MONEY データ型と SMALLMONEY データ型として格納されているデータは、CURRENCY データ型にエクスポートされます。それ以外の場合、数値データは数値としてエクスポートされます。

○テーブルは、**エクスポートウィザード**を使用してエクスポートできます。

「データエクスポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

SQL Anywhere モニターの新機能

次に、バージョン 12.0.0 で導入された SQL Anywhere モニターの追加機能を示します。

- **新しいユーザーインターフェイス**　モニターのインターフェイスは、ダッシュボードベースです。ダッシュボードには、メトリック、警告、リソース情報を表示するためのウィジェットがあります。ダッシュボードは、各ユーザーに固有です。どのユーザーも各自のダッシュボードを追加、編集、または削除できます。「[概要] ダッシュボード」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Mobile Link サーバーファームと Relay Server ファームのモニタリング**　モニターを使用して、SQL Anywhere データベースと Mobile Link サーバーだけでなく、Mobile Link サーバーファームと Relay Server ファームもモニタリングできます。「SQL Anywhere モニター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **モニター内からの接続の切断**　モニター内から、リソースデータベースへの接続を切断できます。「モニターを使用した接続の切断」『SQL Anywhere サーバー データベース管理』を参照してください。

- **モニタリングする SQL Anywhere リソースのインポート**　リソースのリストを含む CSV ファイルを作成して、このリストをモニターにインポートできます。以前は、モニターに一度に追加できるリソースは 1 つのみでした。「複数のリソースの追加」『SQL Anywhere サーバー データベース管理』を参照してください。

- **オンデマンドでのメンテナンスの実行**　管理者は、スケジュールされていないメンテナンスをモニターで実行できます。「モニターのバックアップ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL Anywhere リソースの新しいバックアップ警告**　管理者は、SQL Anywhere リソースが指定日数の間に正常にバックアップされていない場合、警告を発行するようにモニターを設定

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

できます。デフォルトでは、リソースが正常にバックアップされてから 14 日が経過した後で警告が発行されます。「警告スレッショルドの指定」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ブラウザーのローカル時刻によるレポート**　モニターでレポートされる時刻は、常に、使用しているブラウザーのローカル時刻です。ブラウザーとリソース間の時刻の差異を確認するには、「時刻の表示方法の概要」『SQL Anywhere サーバー データベース管理』を参照してください。

- **メトリックのエクスポート**　グラフや表を関連付けられたメトリックを XML ファイルにエクスポートできます。たとえば、**[主要なパフォーマンスのメトリック]** ウィジェットのほとんどのメトリックをエクスポートできます。「メトリックのエクスポート」『SQL Anywhere サーバー データベース管理』を参照してください。

- **トラブルシューティング機能**　モニターをトラブルシューティングする必要がある場合、管理者は **[メッセージログ]** 機能と **[例外レポート]** 機能を使用できます。「メッセージログの表示。」『SQL Anywhere サーバー データベース管理』と「例外レポートの表示」『SQL Anywhere サーバー データベース管理』を参照してください。

SQL Anywhere モニターの動作の変更

次に、バージョン 12.0.0 で導入された SQL Anywhere モニターの変更点を示します。

- **メトリック収集の変更**　以前は、モニターで収集するメトリックのタイプや警告スレッショルドを設定できましたが、警告スレッショルドのみが設定できるようになりました。「SQL Anywhere モニター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **リソースの削除**　リソースを停止せずに、モニタリング済みのリソースをモニターから削除できます。以前は、削除する前にリソースを停止する必要がありました。「リソースの削除」『SQL Anywhere サーバー データベース管理』を参照してください。

- **リソースの状態がなくなった**　リソースにはステータスのみが存在するようになりました。「[概要] ダッシュボード」『SQL Anywhere サーバー データベース管理』を参照してください。

- **警告ステータスの変更**　警告をトリガーする条件が存在しない場合、警告ステータスは **[非アクティブ]** に変更されます。ステータスが **[非アクティブ]** の警告は、その警告をトリガーした条件は存在しませんが、モニターのユーザーが手動で解決していないことを示します。また、モニターで発行された各警告には重大度が割り当てられます。「[概要] ダッシュボード」『SQL Anywhere サーバー データベース管理』を参照してください。

- **警告スレッショルドのデフォルトの変更**　以前は、CPU メモリの使用率が 2 つの連続する収集間隔で最大キャッシュサイズの 85 パーセントに達したときに、モニターによって警告が発行されました。デフォルトは 90 パーセントになり、時間スレッショルドは 30 秒になりました。「警告スレッショルド」『SQL Anywhere サーバー データベース管理』を参照してください。

- **すべてのユーザーがモニターにログインする必要がある**　以前は、モニターへの読み込み専用アクセスにログインは不要でしたが、すべてのユーザーがモニターにログインすることが必

要になりました。「モニターのユーザー」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ユーザーの言語設定の変更**　ユーザーがログインすると、そのユーザーの言語設定を使用して、モニターに表示される言語と警告に使用される言語が設定されます。以前は、モニターではブラウザーで設定された言語が使用されていました。「モニターユーザーの編集」『SQL Anywhere サーバー データベース管理』を参照してください。

マニュアルの強化

次に、バージョン 12.0.0 で SQL Anywhere マニュアルに加えられた変更を示します。

- **よくある質問 (FAQ)**　よくある質問の包括的なリストを使用できるようになりました。FAQ は、主に初級レベルと中級レベルのユーザーを対象としており、フォーラムのディスカッションで頻繁に発生する、数多くの一般的な質問に対する回答が含まれています。「よくある質問 - SQL Anywhere」『SQL Anywhere 12 紹介』を参照してください。

- **データベースサーバーのパフォーマンス警告**　データベースサーバーメッセージウィンドウに表示されるパフォーマンス警告に関して、その説明と、パフォーマンス問題の対応策を示す情報のリンクを使用できるようになりました。「データベースサーバーパフォーマンス警告のトラブルシューティング」『SQL Anywhere サーバー データベース管理』を参照してください。

- **DocCommentXchange はデフォルトのマニュアル**　DocCommentXchange は、SQL Anywhere 12 のデフォルトのマニュアル形式です。DocCommentXchange は、Web で SQL Anywhere マニュアルにアクセスし、マニュアルについて議論するためのオンラインコミュニティです。

  マニュアルのローカルコピーをインストールする場合は、www.sybase.com/detail?id=1069195 を参照してください。

- **Sybase.com の新しい開発者センター**　SQL Anywhere マニュアルを補足するには、http://www.sybase.com/developer/library/sql-anywhere-techcorner にある SQL Anywhere 開発者センターにアクセスしてください。技術ホワイトペーパー、FAQ、テクニカルノート、ダウンロード、テックキャストなどを参照し、質問に対する回答を見つけることができます。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 11.0.1 の新機能

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

SQL Anywhere のバージョン 10 より前における新機能や動作の変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

SQL Anywhere の新機能

次に、バージョン 11.0.1 で導入された SQL Anywhere データベースとデータベースサーバーの追加機能を示します。

接続プロパティ

SQL Anywhere バージョン 11.0.1 で追加された接続プロパティは次のとおりです。

- Authenticated
- IsDebugger
- QueryBypassedCosted
- QueryBypassedHeuristic
- QueryBypassedOptimized
- QueryOpened
- QueryDescribedBypass
- QueryDescribedOptimizer
- StatementDescribes
- StatementPostAnnotates
- StatementPostAnnotatesSimple
- StatementPostAnnotatesSkipped
- WaitStartTime
- WaitType

これらのプロパティの説明については、「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースプロパティ

SQL Anywhere バージョン 11.0.1 で追加されたデータベースプロパティは次のとおりです。

- Authenticated
- Prepares
- QueryBypassedCosted
- QueryBypassedHeuristic
- QueryBypassedOptimized
- QueryOpened
- QueryDescribedBypass
- QueryDescribedOptimizer
- StatementDescribes
- StatementPostAnnotates
- StatementPostAnnotatesSimple
- StatementPostAnnotatesSkipped

これらのプロパティの説明については、「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースサーバープロパティ

次に、SQL Anywhere バージョン 11.0.1 でのデータベースサーバープロパティの強化を示します。

- ServerEdition

これらのプロパティの説明については、「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースユーティリティ

次に、SQL Anywhere バージョン 11.0.1 でのデータベースユーティリティの強化を示します。

- **サービスユーティリティ (dbsvc) の強化**　Windows で、サービスユーティリティを使用して、Mobile Link リレーサーバー (rshost)、Mobile Link リレーサーバー Outbound Enabler (RSOE)、および SQL Anywhere ボリュームシャドウコピーサービス (dbvss11) のサービスを作成できるようになりました。「Windows 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **アンロードユーティリティ (dbunload) の強化**　dbunload ユーティリティで次のオプションがサポートされるようになりました。

  - **-cm オプション**　アンロードされるデータベースの dbinit コマンドまたは CREATE DATABASE 文を表示します。

  - **-l オプション**　再構築されたデータベースの AUTOINCREMENT カラムに、次に使用可能な値を保持します。

  「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

システムプロシージャーとファンクション

次に、SQL Anywhere バージョン 11.0.1 で追加されたシステムプロシージャーとファンクションの強化を示します。

- **sa_get_table_definition システムプロシージャー**　新しい sa_get_table_definition システムプロシージャーは、指定されたテーブルおよびそのインデックス、外部キー、トリガー、付与されたパーミッションの作成に必要な SQL 文を含む LONG VARCHAR 文字列を返します。「sa_get_table_definition システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **FIRST_VALUE 関数 [集合]**　FIRST_VALUE 関数 [集合] に RESPECT NULLS 句が含まれるようになりました。「FIRST_VALUE 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **LAST_VALUE 関数 [集合]**　LAST_VALUE 関数 [集合] に RESPECT NULLS 句が含まれるようになりました。「LAST_VALUE 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_set_http_option システムプロシージャー**　AcceptCharset オプションにより、文字セットをより柔軟に選択できるようになりました。「sa_set_http_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

SQL 文

次に、SQL Anywhere バージョン 11.0.1 での SQL の強化を示します。

- **新しい DEFAULT VALUES 句、INSERT 文**　INSERT 文に新しく追加された DEFAULT VALUES 句により、すべてのカラムにデフォルト値を挿入できます。「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE ENCRYPTED DATABASE 文**　この文は、既存のデータベースの暗号化されたコピー (すべてのトランザクションログ、ミラーログ、DB 領域を含む) を作成します。この文を使用して、テーブル暗号化が有効になっているデータベースのコピーも作成できます。「CREATE ENCRYPTED DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  リカバリが必要なデータベースを暗号化したい場合 (テクニカルサポートに送る場合など) は、今までどおり CREATE ENCRYPTED FILE 文を使用する必要があります。「CREATE ENCRYPTED FILE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE DECRYPTED DATABASE 文**　この文は、既存のデータベースの復号化されたコピー (すべてのトランザクションログ、ミラーログ、DB 領域を含む) を作成します。「CREATE DECRYPTED DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  リカバリが必要なデータベースを復号化したい場合 (テクニカルサポートに送る場合など) は、今までどおり CREATE DECRYPTED FILE 文を使用する必要があります。「CREATE DECRYPTED FILE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER DATABASE 文の強化**　ミラーリング中のデータベースサーバーに ALTER DATABASE UPGRADE 文を実行すると、エラーになります。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **MESSAGE 文の強化**　IMMEDIATE 句を指定すると、接続がアイドルであるか要求を行っているかに関係なく、クライアントのメッセージコールバックルーチンによってメッセージが短時間に受信されます。「MESSAGE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **関数、プロシージャー、トリガー、ビューの新規作成または同じ名前のオブジェクトの置換**　新しい OR REPLACE 句により、関数、プロシージャー、トリガー、ビューの新規作成や、同じ名前のオブジェクトが存在する場合は置換することができます。次の項を参照してください。

  - 「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE TRIGGER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE VIEW 文」『SQL Anywhere サーバー SQL リファレンス』

- **存在しないデータベースオブジェクトの削除を文で試みてもエラーを表示しない**　新しい IF EXISTS 句により、存在しないデータベースオブジェクトの削除を DROP 文で試みてもエラーを返さないように指定できます。次の項を参照してください。

  - 「DROP EVENT 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP FUNCTION 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP TRIGGER 文」『SQL Anywhere サーバー SQL リファレンス』
  - 「DROP VIEW 文」『SQL Anywhere サーバー SQL リファレンス』

- **新しい INTO LOCAL TEMPORARY TABLE 句、SELECT 文**　SELECT 文に新しく追加された INTO LOCAL TEMPORARY TABLE 句により、ローカルテンポラリテーブルを作成し、SELECT 文の結果セットを設定できます。これまでは、テンポラリテーブル名が # から始まる場合に INTO 句を使用する方法しかありませんでした。「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい IF NOT EXISTS 句、CREATE TABLE 文**　CREATE TABLE 文に新しく追加された IF NOT EXISTS 句により、永久テーブル、グローバルテンポラリテーブル、ローカルテンポラリテーブルが既存しない場合に作成できます。「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **テンポラリプロシージャーやテンポラリ関数の作成時に所有者を指定**　CREATE FUNCTION 文および CREATE PROCEDURE 文により、テンポラリプロシージャーまたはテンポラリ関数に対して任意で所有者を指定できるようになりました。次の項を参照してください。

  - 「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』
  - 「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

プログラミングインターフェイス

次に、SQL Anywhere バージョン 11.0.1 でのプログラミングインターフェイスの強化を示します。

- **新しい ASP.NET プロバイダー**　次の新しい ASP.NET プロバイダーは、標準の ASP.NET プロバイダーの機能を模倣しますが、データを SQL Server データベースではなく SQL Anywhere データベースに格納します。

  - **メンバーシップ**　ログイン、ログアウト、ユーザーおよびパスワードの管理を実現します。
  - **ロール**　ユーザーのグループへの割り当て、および単純で簡単なパーミッション管理を実現します。
  - **プロファイル**　ユーザー変数を格納します。
  - **Web パーツパーソナル化**　Web パーツデータの格納を管理し、ユーザーによるビューのパーソナル化を実現します。
  - **Web イベント**　ヘルスモニタリング機能と連携し、フラッシュされた Web イベント情報をデータベースに格納します。

  「SQL Anywhere ASP.NET プロバイダー」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Ruby のサポート**　SQL Anywhere は、Ruby オープンソースプログラミング言語をサポートするようになりました。「SQL Anywhere の Ruby API サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **OLE DB での CATALOGS と SCHEMATA の各ローセットのサポート**　OLE DB の CATALOGS および SCHEMATA の各ローセットがサポートされるようになりました。SQL Anywhere はカタログの概念をサポートしていないため、SQL Anywhere OLE DB プロバイダーは、CATALOGS の代わりに現在開始されているすべてのデータベースとそのロケーションを含む結果セットを返します。同様に SCHEMATA では、結果セット内でカタログとしてデータベース名が使用されます。

- **ADO.NET Entity Framework に追加されたサポート**　SQL Anywhere で、ADO.NET Entity Framework モデルがサポートされるようになりました。「SQL Anywhere .NET データプロバイダーの機能」『SQL Anywhere サーバー プログラミング』を参照してください。

その他

次に、SQL Anywhere バージョン 11.0.1 でのその他の強化を示します。

- **単純で低コストな文の実行に関するパフォーマンスの向上**　実行時間が低コストの文のための実行プランを生成するのに要するコストは、文を実行するためのコストを上回る場合があります。次の強化により、SQL Anywhere で認識される、実行時間が低コストの単純な文のクラスが拡張され、このような文がオプティマイザーをバイパスできるようになりました。

詳細については、「クエリ処理のフェーズをスキップするための条件」『SQL Anywhere サーバー SQL の使用法』を参照してください。

○ **新しい FORCE NO OPTIMIZATION 句**　以前までは、コストベースの最適化が必要な文はすべてオプティマイザーによって処理されていました。今回、FORCE NO OPTIMIZATION 句が追加されたことで、文がオプティマイザーをバイパスするよう指定できるようになりました。文が複雑すぎて (データベースオプションの設定またはスキーマやクエリの特徴などにより) このような処理が実行できない場合、文は失敗し、データベースサーバーはエラーを返します。

次の文は、新しい FORCE NO OPTIMIZATION 句をサポートしています。

- 「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「UPDATE 文」『SQL Anywhere サーバー SQL リファレンス』

○ **新しい接続プロパティ**

- QueryBypassedCosted
- QueryBypassedHeuristic
- QueryBypassedOptimized
- QueryOpened
- QueryDescribedBypass
- QueryDescribedOptimizer
- StatementDescribes
- StatementPostAnnotates
- StatementPostAnnotatesSimple
- StatementPostAnnotatesSkipped

○ **新しいデータベースプロパティ**

- Prepares
- QueryBypassedCosted
- QueryBypassedHeuristic
- QueryBypassedOptimized
- QueryOpened
- QueryDescribedBypass
- QueryDescribedOptimizer
- StatementDescribes
- StatementPostAnnotates
- StatementPostAnnotatesSimple
- StatementPostAnnotatesSkipped

○ **新しい [最適化方法] フィールド**　グラフィカルなプランの [オプティマイザー統計] セクションに、[最適化方法] フィールドが追加されました。このフィールドは、クエリオプティマイザーによって選択された実行方式を返します。「実行プランのコンポーネント」『SQL Anywhere サーバー SQL の使用法』を参照してください。

● **データベースサーバーがデフォルトデータベースサーバーにならないようにする**　-xd サーバーオプションにより、データベースサーバーがデフォルト TCP ポートで受信するのを防止し、

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

データベースサーバーがデフォルトデータベースサーバーにならないようにします。「-xd dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **並列アーカイブバックアップのサポート**　SQL Anywhere データベースサーバーで、サーバー側のアーカイブバックアップに対する並列バックアップがサポートされるようになりました。並列データベースバックアップでは、物理 I/O を利用して、直列ではなく並列で読み書き操作を行うことで、パフォーマンスが向上します。

  BACKUP DATABASE 文に、並列アーカイブバックアップをサポートするための新しい句が 2 つ追加されました。

  - **WITH CHECKPOINT LOG [NO] COPY**
  - **MAX WRITE** { *n* | **AUTO** }

  バージョン 11.0.0 以前のデータベースサーバーでは、バージョン 11.0.1 のデータベースサーバーで生成されたアーカイブバックアップをリストアできません。バージョン 11.0.1 のデータベースサーバーでは、旧バージョンのデータベースサーバーで生成されたバックアップをリストアできます。

  「BACKUP 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **Mac OS X での Developer Edition と Evaluation Edition のデータベースサーバーの実行**　Mac OS X で、Developer Edition と Evaluation Edition のデータベースサーバーを管理ツールから自動的に起動できるようになりました。

SQL Anywhere の動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された SQL Anywhere データベースとデータベースサーバーに加えられた変更を、カテゴリごとに示します。

動作の変更

- **全文検索**　全文検索をサポートするよう、次の動作変更が行われました。

  - **演算子の優先度が適用される**　以前までは、クエリ文字列に演算子の優先度は適用されていませんでした。このリリースでは、次の演算子の優先度が適用されるようになりました。

    - NEAR、FUZZY 演算子
    - AND NOT 演算子
    - AND 演算子
    - OR 演算子

    「CONTAINS 探索条件における演算子の優先度」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **NEAR 句の引数は単語またはプレフィクス単語にする**　近接検索を実行する場合、NEAR 句の引数は単語またはプレフィクス単語である必要があります。「CONTAINS 探索条件」

『SQL Anywhere サーバー SQL リファレンス』と「近接検索」『SQL Anywhere サーバー SQL の使用法』を参照してください。

○ **ハイフンおよび AND NOT 句の使用**　フレーズ内では、ハイフンは特殊文字ではなく単語区切りとして処理されます。フレーズの外にあるハイフンの処理は、ハイフンを囲む構文によって異なります。「ハイフン (-) を使用できる構文」『SQL Anywhere サーバー SQL リファレンス』と「全文検索での AND NOT 演算子の使用」『SQL Anywhere サーバー SQL の使用法』を参照してください。

○ **アスタリスクおよびプレフィクス検索の使用**　プレフィクス検索を実行する場合、単語にアスタリスクを付加し、クエリ文字列の末尾とするか、またはそのすぐ後にスペースか許可された特殊文字を 1 つ挿入する必要があります。「アスタリスク (*) を使用できる構文」『SQL Anywhere サーバー SQL リファレンス』と「プレフィクス検索」『SQL Anywhere サーバー SQL の使用法』を参照してください。

○ **重複するテキストインデックスを作成するとエラーが返される**　重複するテキストインデックスを作成できなくなりました。次の設定が既存のテキストインデックスと同じ場合、そのテキストインデックスは重複するものと見なされます。

●参照されるベーステーブル

●インデックス対象カラム (順序は問わない)

●使用される設定オブジェクトの設定内容 (TERM BREAKER、MINIMUM TERM LENGTH、MAXIMUM TERM LENGTH、STOPLIST、照合情報)

SQL Anywhere 11.0.1 よりも前のバージョンで作成された重複するテキストインデックスは、データベース内に残すことができ、11.0.1 のデータベースサーバーで開始してもエラーは発生しません。ただし、重複するテキストインデックスを含むデータベースをバージョン 11.0.1 以降で再ロードすると、エラーが返されます。

既存のデータベース内にある重複するテキストインデックスを識別するには、次のクエリを実行します。

```
SELECT LIST( i.index_name )
FROM SYS.SYSIDX i
   JOIN SYS.SYSTEXTIDX t ON i.object_id = t.index_id AND t.sequence = 1
   JOIN SYS.SYSTEXTCONFIG F ON F.object_id = t.text_config
   JOIN (
   SELECT table_id, index_id, LIST( column_id, ', ' ORDER BY column_id ) col_id
   FROM SYS.SYSIDXCOL
   GROUP BY table_id, index_id) x
   ON x.table_id = i.table_id AND x.index_id = i.index_id
WHERE i.index_category=4
GROUP BY i.table_id, f.term_breaker, f.min_term_length, f.max_term_length,
   f.collation, ISNULL( f.char_stoplist, '-' ),
   ISNULL( f.nchar_stoplist, '-' ), x.col_id
HAVING count(*) > 1
```

このクエリは、ストップリストを表す文字列がまったく同じであるか、NO STOPLIST が指定されている場合のみ有効です。たとえば、ストップリスト 'a b c' とストップリスト 'a - b c' は、このクエリでは同じものとは見なされませんが、テキストインデックスを作成する際の重複のチェックでは同じものと見なされます。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **正規表現**　SIMILAR TO、REGEXP の各探索条件、および REGEXP_SUBSTR 関数の動作に変更が加えられました。変更の主な目的は、SIMILAR TO の ANSI/SQL 標準との整合性を保ちながら、REGEXP と REGEXP_SUBSTR の動作を Perl と整合させることにあります。

  - **データベースの照合および一致**　以前までは、REGEXP と REGEXP_SUBSTR は照合等価やソート順を使用して、パターン内のリテラルクラスや文字クラスの範囲が文字列と一致するかを判断していました。このリリースでは、REGEXP と REGEXP_SUBSTR はコードポイント値のバイナリ比較を使用して、範囲の一致や評価を行っています。この変更は、Perl 5.0 と動作の整合性が保たれるようにするため行われています。

    SIMILAR TO は依然としてデータベース照合を使用して一致や範囲の評価を行っています。「LIKE 探索条件、REGEXP 探索条件、SIMILAR TO 探索条件」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **データベースの大文字と小文字の区別および [[:upper:]] と [[:lower:]] の部分文字クラス**　大文字と小文字を区別しないデータベースでは、SIMILAR TO と REGEXP の [[:upper:]] と [[:lower:]] の部分文字クラスは区別されていませんでした。データベースでの大文字と小文字の区別にかかわらず、[[:upper:]] は大文字とだけ一致し、[[:lower:]] は小文字とだけ一致するように、これは変更されました。

  - **メタ文字としての脱字記号 (^)、アンダースコア (_)、パーセント記号 (%) の処理**　次の表に、メタ文字としての各文字がどのように処理されるかを、以前の動作と新しい動作に分けて示します。

文字	以前の動作	新しい動作
_ (アンダースコア)	SIMILAR TO、REGEXP、および REGEXP_SUBSTR では、アンダースコアはメタ文字として処理され、任意の 1 文字と一致していました。	SIMILAR TO では、アンダースコアはメタ文字として処理され、任意の 1 文字と一致します。 REGEXP と REGEXP_SUBSTR では、アンダースコアはメタ文字として処理されません。REGEXP と REGEXP_SUBSTR で任意の 1 文字と一致させるためには、代わりにピリオド (.) を使用します。

文字	以前の動作	新しい動作
%	SIMILAR TO、REGEXP、および REGEXP_SUBSTR では、パーセント記号はメタ文字として処理され、任意の数の任意の文字と一致していました。	SIMILAR TO では、パーセント記号はメタ文字として処理され、任意の数の任意の文字と一致します。 REGEXP と REGEXP_SUBSTR では、パーセント記号はメタ文字として処理されません。REGEXP と REGEXP_SUBSTR で任意の数の任意の文字と一致させるためには、代わりにピリオドとアスタリスク (.*) を使用します。
^	SIMILAR TO、REGEXP、および REGEXP_SUBSTR では、文字クラス内の脱字記号は右にある文字の否定や減法として処理され、一致しないと解釈されていました。	SIMILAR TO では、脱字記号は右にある文字の否定や減法として処理されます。たとえば、SIMILAR TO [a-d^c] は a、b、d とは一致しますが、c とは一致しません。 REGEXP と REGEXP_SUBSTR では、脱字記号は文字クラス内で先頭に位置する場合のみメタ文字として処理され、文字クラスの否定として解釈されます。たとえば、REGEXP [^abc] は a、b、c 以外の任意の 1 文字と一致し、REGEXP [a-d^c] は a、b、c、d、および ^ と一致します。

- **アップグレードユーティリティ (dbupgrad) の動作の変更**　データベースミラーリングに参加しているデータベースをアップグレードするために、アップグレードユーティリティを使用するとエラーになります。「アップグレードユーティリティ (dbupgrad)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Mac OS X で dbmodenv が不要になった**　以前のリリースでは、Mac OS X でグラフィカルな管理ツールを使用するためには、ユーザーの *$HOME/.MacOSX/environment.plist* ファイルの PATH と DYLD_LIBRARY_PATH に、SQL Anywhere のバイナリとライブラリのロケーショ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ンを追加する必要がありました。そのために dbmodenv ツールを使用していました。SQL Anywhere は、*$HOME/.MacOSX/environment.plist* の設定に依存しなくなったため、dbmodenv を実行したり、SQL Anywhere をインストールした後にログアウトし、ログインし直す必要がなくなりました。

- **dbisqlc の OUTPUT 文から返される NULL 値のデフォルトが変更**　以前のリリースでは、dbisqlc の OUTPUT 文を使用すると、文は NULL 値として **(NULL)** 値を返していました。このリリースでは、文で NULL 値のデフォルトとして空の文字列が返されるようになりました。output_nulls オプションを設定することで、NULL 値のエクスポート方法を変更できます。「output_nulls オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』を参照してください。

- **エンディアンのサポート**　アップグレード後、ビッグエンディアンコンピューターで作成された 11.0.1 よりも前のテキストインデックスは、トランケートおよびリフレッシュ (MANUAL REFRESH および AUTO REFRESH のテキストインデックスの場合)、または再作成 (IMMEDIATE REFRESH のインデックスの場合) が必要になります。

- **Mac OS X の管理ツール**　Mac OS X の SQL Anywhere 管理ツールで、64 ビットの JDK 1.6 が使用されるようになりました。管理ツールは、Apple JDK 1.6 (Mac OS X 10.5.2 以降) でサポートされている、64 ビットのプロセッサーを搭載した Intel ベースの Macintosh コンピューターだけで動作します。Mac OS X 用の管理ツールを配備する場合、ネイティブライブラリは *$SQLANY11/System/lib64* にあります。「Linux、Solaris、Mac OS X における管理ツールの配備」『SQL Anywhere サーバー プログラミング』を参照してください。

- **HTTP クライアントのチャンクされた転送コーディングの新しいデフォルトサイズ**　これまでは、HTTP クライアントから 2048 バイトを超えるデータが送信された場合、デフォルトで (またはユーザーが CREATE PROCEDURE...SET 'HTTP(CH=auto)' を指定した場合)、チャンクされた転送コーディングが試行されました。デフォルトのサイズが 2048 バイトから 8196 バイトに変更されました。また、チャンクされた転送コーディングを使用しないで要求を再発行する基準に、新しいステータス 411 **Length Required** が追加されました。「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ansi_substring オプションのサポート**　ansi_substring オプションはバージョン 11.0.0 で非推奨になりましたが、バージョン 11.0.1 でまたサポートされるようになりました。「ansi_substring オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

廃止予定機能とサポート終了機能

- **COMMENT ON EXTERNAL ENVIRONMENT OBJECT *object-name***　この構文は **COMMENT ON EXTERNAL OBJECT** *object-name* に変更されました。以前の構文は現在でも受け入れられますが、今後のリリースではサポートされない可能性があります。「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Mobile Link の新機能

次に、バージョン 11.0.1 で導入された Mobile Link の追加機能を示します。

- **リモートデータベーススキーマのスキーマキャッシュ**　新しいスキーマキャッシュ機能により、小規模な同期のオーバーヘッドが縮小されます。リモートスキーマは、最初の同期で Mobile Link サーバーによってキャッシュされます。以降の同期では、スキーマがキャッシュされていない場合にのみリモートスキーマ情報が Mobile Link サーバーに送信されます。

- **mlsrv11 の -vi オプション**　アップロードされた各ローのカラム値を表示します。「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv11 の -vq オプション**　ダウンロードされた各ローのカラム値を表示します。「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv11 の -vm オプション**　同期が完了するたびに、各同期と各同期フェーズの継続時間をログに出力します。「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv11 の -ppv オプション**　指定された期間に従って、Mobile Link が新しい定期モニタリングの値を出力するようにします。「-ppv mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **ml_ignore プレフィクス**　Mobile Link サーバーでは、--{ml_ignore} で始まる SQL スクリプトが、意図的に無視するスクリプトとして認識されます。「無視されたスクリプト」『Mobile Link サーバー管理』を参照してください。

- **dblsn の -sv オプション**　データベースを認証するために Mobile Link Listener が使用するスクリプトバージョンを指定します。「-sv dblsn オプション」『Mobile Link サーバー起動同期』を参照してください。

- **Oracle VArray のサポート**　iAnywhere Solutions 11 - Oracle ODBC ドライバーで、ストアドプロシージャーでの Oracle VArray の使用がサポートされるようになりました。「Oracle VARRAY」『Mobile Link サーバー管理』を参照してください。

- **ライトウェイトポーリングのリスナーキーワード変数**　ライトウェイトポーリングをサポートする、poll_connect、poll_notifier、poll_key、poll_every の各リスナーキーワードが追加されました。

- **ライトウェイトポーリングのアクション変数**　ライトウェイトポーリングをサポートする、$poll_connect、$poll_notifier、$poll_key、$poll_every の各アクション変数が追加されました。

- **Common Access Card を使用したクライアント認証**　Mobile Link クライアントで、Common Access Card (CAC) のクライアント ID を使用した認証がサポートされるようになりました。「identity_name」『Mobile Link クライアント管理』を参照してください。

  注意
  この機能は CAC 認証アドオンの一部であり、別途ライセンスが必要です。「別途ライセンスが必要なコンポーネント」『SQL Anywhere 12 紹介』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **Microsoft SQL Server 2008 のサポート**　Mobile Link 同期サーバーで Microsoft SQL Server 2008 上で実行される統合データベースがサポートされるようになりました。Microsoft の新しい DATE、TIME、DATETIME2 データ型のマッピングの詳細については、「Microsoft SQL Server データのマッピング」『Mobile Link サーバー管理』を参照してください。

- **.NET DownloadTableData インターフェイスの新しいメソッド**　getLastDownloadTime メソッドによってテーブルの最終ダウンロード時刻が返されます。DownloadTableData.GetLastDownloadTime メソッド [Mobile Link サーバー .NET]『Mobile Link サーバー管理』を参照してください。

- **対象 Mobile Link ユーザーとリモート ID に対するログの冗長性**　対象とする Mobile Link ユーザーまたはリモート ID のログに、別の冗長性を設定できるようになりました。「対象 Mobile Link ユーザーとリモート ID に対する ログの冗長性」『Mobile Link サーバー管理』を参照してください。

- **モデルモードでの MySQL のサポート**　Mobile Link プラグインで MySQL 統合データベースがサポートされるようになりました。

- **mlmon の -c オプション**　Mobile Link モニター用に、mlmon コマンドに -c オプションが追加されました。-c オプションは、接続の最後で Mobile Link モニターを閉じ、指定したデータベースにセッションを保存します。

Mobile Link の動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された Mobile Link の変更点を示します。

- **mlsrv11 の -sm オプション**　mlsrv11 の -sm オプションが改善され、非永続的な HTTP/HTTPS に使用したときに -nc オプションに似た機能を提供するようになりました。「-sm mlsrv12 オプション」『Mobile Link サーバー管理』と「-nc mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **Microsoft SQL Server データ型**　SQL Server 2005 で、TEXT、NTEXT、IMAGE の各データ型が推奨されなくなりました。代わりに VARCHAR(*max*)、NVARCHAR(*max*)、VARBINARY(*max*) を使用してください。「Microsoft SQL Server データのマッピング」『Mobile Link サーバー管理』を参照してください。

QAnywhere の新機能

次に、バージョン 11.0.1 で導入された QAnywhere の追加機能を示します。

- **QAnywhere スタンドアロンクライアント**　QAnywhere スタンドアロンクライアントは、QAnywhere Agent を実行したり、データベースを管理したりせずに、メッセージングシステムを設定できる小型のクライアントです。「QAnywhere スタンドアロンクライアント」『QAnywhere』を参照してください。

Ultra Light の新機能

次に、バージョン 11.0.1 で導入された Ultra Light の追加機能を示します。

- **ミラーファイル**　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。「Ultra Light MIRROR_FILE 接続パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

- **ml_remote_id の設定解除**　SET OPTION を使用して **ml_remote_id** の設定を解除できるようになりました。「SET OPTION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』と「ML_GET_SERVER_NOTIFICATION 関数 [システム]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **ML_GET_SERVER_NOTIFICATION**　この関数を使用すると、Ultra Light ユーザーはライトウェイトポーリングを使用して、サーバー起動同期要求について Mobile Link サーバー上の Notifier に問い合わせできます。「ML_GET_SERVER_NOTIFICATION 関数 [システム]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **SYNC_PROFILE_OPTION_VALUE 関数**　この関数は、同期プロファイル内の指定されたオプションの値を返します。「SYNC_PROFILE_OPTION_VALUE 関数 [システム]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light.NET API の StreamErrorParameters プロパティ**　**StreamErrorParameters** が **SyncResult** クラスに追加されました。このメンバーには、StreamErrorCode でレポートされるストリームエラーコードのエラーパラメーターをカンマで区切ったリストが含まれます。ULSyncResult.StreamErrorParameters プロパティ [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

- **Ultra Light for M-Business API の getStreamErrorParameters メソッド**　このメソッドは、同期ストリーム処理によってレポートされるエラーパラメーターをカンマで区切ったリストを返します。SyncResult.getStreamErrorParameters メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』を参照してください。

Ultra Light の動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された Ultra Light の変更点を示します。

- **Palm のサスペンド機能**　推奨されなくなりました。

Ultra Light J の新機能

次に、バージョン 11.0.1 で導入された Ultra Light J の追加機能を示します。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **追加 SQL サポート**　Ultra Light J には、次の追加 SQL サポートがあります。

  ○ALTER TABLE

  ○CREATE TABLE

  ○CREATE INDEX

  ○DROP INDEX

  ○DROP TABLE

  ○TRUNCATE TABLE

  また、次の制約がなくなりました。

  ○HAVING がサポートされます。

  ○集合関数内の DISTINCT がサポートされます。

  ○CURRENT TIME、CURRENT TIMESTAMP、CURRENT DATE がサポートされます。

  「Ultra Light SQL 文」『Ultra Light データベース管理とリファレンス』を参照してください。

- **DatabaseInfo の新しいメソッド**　**getPageReads()** と **getPageWrites** の 2 つの新しいメソッドが **DatabaseInfo** インターフェイスに追加されました。これらのメソッドは、DatabaseInfo オブジェクトの作成時点のページ読み込みとページ書き込みの数を返します。DatabaseInfo.getPageReads メソッド [Ultra Light J]『Ultra Light - Java プログラミング』と DatabaseInfo.getPageWrites メソッド [Ultra Light J]『Ultra Light - Java プログラミング』を参照してください。

- **Ultra Light J データベース転送ユーティリティの更新**　Ultra Light J データベース転送ユーティリティで、クライアントからのデータベースの転送に加えて、データベースの削除、データベース情報の表示、データベース転送ログの表示または電子メール送信が可能になりました。「Ultra Light Java Edition データベース転送ユーティリティ (ULjDbT)」『Ultra Light データベース管理とリファレンス』を参照してください。

Ultra Light J の動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された Ultra Light J の変更点を示します。

- **@@identity グローバル変数**　**@@identity** グローバル変数は Ultra Light J でサポートされていません。

管理ツールの新機能

次に、バージョン 11.0.1 で導入された Sybase Central と Interactive SQL の追加機能を示します。

Sybase Central の新機能

次に、バージョン 11.0.1 で導入された Sybase Central のプラグインの追加機能を示します。

すべてのプラグイン

- **サービス作成ウィザードの強化**　**サービス作成ウィザード**を使用して、Mobile Link リレーサーバー (rshost)、Mobile Link リレーサーバー Outbound Enabler (RSOE)、SQL Anywhere ボリュームシャドウコピーサービス (dbvss11)、Mobile Link Listener ユーティリティ (dblsn)、Broadcast Repeater ユーティリティ (dbns11) のサービスを作成できるようになりました。「現在のセッション外でのデータベースサーバーの起動」『SQL Anywhere サーバー データベース管理』を参照してください。

SQL Anywhere プラグイン

- **Sybase Central 内でイベントのステータスを表示**　Sybase Central で、イベントの現在の実行ステータスを確認できるようになりました。**[イベント]** フォルダーの新しい **[実行中]** カラムに、イベントの現在の実行ステータスが表示されます。左ウィンドウ枠でフォルダーを選択すると、フォルダーの内容が更新されます。また、**[イベントのプロパティ]** ウィンドウに **[実行中]** プロパティが表示されます。

  この値は、イベントが実行中の場合は **[はい]**、イベントが実行中ではない場合は **[いいえ]**、イベントが SQL Anywhere 9.0.2 以前のデータベースにある場合は **[不明]** になります。

- **Sybase Central 内でのメンテナンスプランのステータスの表示**　Sybase Central で、メンテナンスプランの現在の実行ステータスを確認できるようになりました。**[メンテナンスプラン]** フォルダーの新しい **[実行中]** カラムに、プランの現在の実行ステータスが表示されます。左ウィンドウ枠でフォルダーを選択すると、フォルダーの内容が更新されます。

  この値は、メンテナンスプランが実行中の場合は **[はい]**、プランが実行中ではない場合は **[いいえ]**、プランが SQL Anywhere 9.0.2 以前のデータベースにある場合は **[不明]** になります。

- **メンテナンスプランの強化**　**メンテナンスプラン作成ウィザード**で、メンテナンスプランレポートを電子メールで送信する設定を行う際に、テスト電子メールを送信できるようになりました。「メンテナンスプランの作成」『SQL Anywhere サーバー データベース管理』を参照してください。

- **プロシージャー作成ウィザードとファンクション作成ウィザードの強化**　**ファンクション作成ウィザード**と**プロシージャー作成ウィザード**で、プロシージャーまたはファンクションを記述する SQL ダイアレクトまたは言語を、Watcom-SQL、Transact-SQL、外部 C/C++、または外部環境から選択できるようになりました。外部環境を選択した場合、C_ESQL32、C_ESQL64、C_ODBC32、C_ODBC64、CLR、JAVA、PERL、PHP のいずれかの言語を選択します。これまでは、Watcom-SQL または Transact-SQL だけを選択できました。C/C++ または Java を選択すると、EXTERNAL NAME 句を含む関数またはプロシージャーのコードスケルトンが生成されます。「CREATE FUNCTION 文 [外部呼び出し]」『SQL Anywhere サーバー SQL リファレンス』と「CREATE PROCEDURE 文 [外部呼び出し]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Mobile Link プラグイン

- **Oracle データベース用の Mobile Link モデルの強化**　**同期モデル作成ウィザード**を使用して Oracle 統合データベースを使用する同期モデルを作成するときに、スキーマ全体をロードするか、同期に必要なテーブルの所有者のサブセットを選択するかを選べるようになりました。所有者のサブセットを選択すると、スキーマのロード時間を短縮できます。

- **Mobile Link モデルモードでの MySQL のサポート**　Mobile Link モデルモードで、MySQL 統合データベースの使用がサポートされるようになりました。

管理ツールの動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された Sybase Central と Interactive SQL の変更点を示します。

Sybase Central の動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された Sybase Central の変更点を示します。

SQL Anywhere プラグイン

- **読み込み専用データベースの強化**　読み込み専用データベースに接続するときに警告が表示されるようになりました。この警告が表示されないようにするには、**[ツール] » [SQL Anywhere 11] » [ユーザー設定]** をクリックして設定します。

Interactive SQL の動作の変更と廃止予定機能

次に、バージョン 11.0.1 で導入された Interactive SQL の変更点を示します。

- **データベースロックの自動解放の設定**　結果セットが表示されるときに作成されるデータベーススキーマロックを解放するように Interactive SQL を設定できるようになりました。このように設定するには、Interactive SQL で **[ツール] » [オプション] » [SQL Anywhere] » [データベースロックの自動解放]** をクリックします。

  このオプションを選択すると、結果セットを返す文を実行した後、データベースにコミットされていない変更が接続にあるかどうかが Interactive SQL によって確認されます。コミットされていない変更がなかった場合は Interactive SQL によってスキーマロックが解放されます。それ以外の場合、スキーマロックは解放されません。つまり、データベースにコミットされていない変更がある場合、スキーマロックは解放されません。

製品全体の新機能

次に、バージョン 11.0.1 で導入された製品全体の追加機能を示します。

- **新しいオンラインマニュアルフォーラム、DocCommentXchange (DCX)**　DocCommentXchange と呼ばれる新しいオンラインフォーラムが作成されました。DocCommentXchange は、SQL Anywhere マニュアルにアクセスし、マニュアルについて議論するためのコミュニティです。

DocCommentXchange は次の目的に使用できます (現在のところ、日本語はサポートされておりません)。

○マニュアルを表示する

○マニュアルの項目について明確化するために、ユーザーによって追加された内容を確認する

○すべてのユーザーのために、今後のリリースでマニュアルを改善するための提案や修正を行う

http://dcx.sybase.com を参照してください。

- **SQL Anywhere モニター**　SQL Anywhere モニターは、SQL Anywhere データベースや Mobile Link サーバーの正常性や可用性に関する情報を示す、ブラウザーベースの管理ツールです。

  Mobile Link では、この機能は既存の Mobile Link モニターの機能に代わるものではなく、また Mobile Link モニターの機能と重複しません。

  「SQL Anywhere モニター」『SQL Anywhere サーバー データベース管理』を参照してください。

マニュアルの強化

- **SQL Anywhere のバックアップとリカバリのマニュアルの強化**　SQL Anywhere に付属のバックアップツールおよびリカバリツールを使用するためのマニュアルが、今回のリリースに合わせて書き直されました。「バックアップとデータリカバリ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **全文検索のマニュアルの強化**　全文検索機能のマニュアルの構成が変更され、例やチュートリアルが追加されました。「全文検索」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **Mobile Link サーバー起動同期のマニュアル**　Mobile Link サーバー起動同期のマニュアルが、バージョン 11.0.0 から改善されました。これには、目次の構成の改善、サーバー起動同期の説明の詳細化、リファレンス情報のフォーマットの改善などが含まれます。Mobile Link サーバー起動同期を参照してください。

- **SQL Remote のマニュアル**　SQL Remote のマニュアルは、構成の変更と改訂が行われ、使いやすくなりました。SQL Remote を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 11.0.0 の新機能

SQL Anywhere のバージョン 10 より前における新機能や動作の変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

SQL Anywhere

- 「SQL Anywhere の新機能」144 ページ
- 「SQL Anywhere の動作の変更」173 ページ
- 「SQL Anywhere の廃止予定機能とサポート終了機能」183 ページ

Mobile Link

- 「Mobile Link の新機能」186 ページ
- 「Mobile Link の動作の変更と廃止予定機能」191 ページ

QAnywhere

- 「QAnywhere の新機能」192 ページ
- 「QAnywhere の動作の変更と廃止予定機能」193 ページ

SQL Remote

- 「SQL Remote の新機能」194 ページ
- 「SQL Remote の動作の変更と廃止予定機能」194 ページ

Ultra Light

- 「Ultra Light の新機能」195 ページ
- 「Ultra Light の動作の変更と廃止予定機能」200 ページ

Sybase Central と Interactive SQL

- 「Sybase Central と Interactive SQL の新機能」202 ページ
- 「Sybase Central と Interactive SQL の動作の変更と廃止予定機能」205 ページ

マニュアルの強化

- 「マニュアルの強化」211 ページ

製品全体の機能

- 「製品全体の新機能」212 ページ
- 「製品全体の動作の変更」212 ページ

SQL Anywhere

次の項では、SQL Anywhere バージョン 11.0.0 の新機能、動作の変更、廃止予定機能について説明します。

SQL Anywhere の新機能

次に、バージョン 11.0.0 で導入された SQL Anywhere データベースとデータベースサーバーの追加機能を示します。

主な機能

次に、SQL Anywhere バージョン 11.0.0 の主な機能を示します。

- **テーブルのマージのサポート**　SQL Anywhere では、テーブル、ビュー、システムプロシージャーの結果をテーブルまたはビューにマージできるようになりました。「MERGE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ログインポリシーのサポート**　SQL Anywhere では、ログインポリシーがサポートされるようになりました。ログインポリシーは、ユーザーに対してデータベース接続が確立されるときに適用するルールを定義する一連のオプションです。SQL Anywhere には、すべてのログインポリシーのデフォルト値を含むルートポリシーがあります。ルートログインポリシーに優先する設定を含む個別のログインポリシーを作成できます。ログインポリシーは各ユーザーに個別に割り当てます。各ユーザーが所属するグループからログインポリシーを継承することはありません。特定のユーザーへのポリシーの割り当ては、必要に応じて変更できます。「ログインポリシー」『SQL Anywhere サーバー データベース管理』を参照してください。

  この機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **全文検索のサポート**　SQL Anywhere で全文検索がサポートされるようになりました。全文検索を行うと、データベース内で特定の単語が出現するすべての箇所を簡単に検索できます。全文検索は、単語単位であるという点、またテーブルのローをスキャンするのではなくテキストインデックスを使用する点で、LIKE、REGEXP、SIMILAR TO などの述部を使用した検索とは異なります。「全文検索」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  全文検索機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **正規表現のサポート**　SQL Anywhere では、REGEXP と SIMILAR TO の 2 つの探索条件を使用した正規表現がサポートされるようになりました。「REGEXP 探索条件」『SQL Anywhere サーバー SQL リファレンス』と「SIMILAR TO 探索条件」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「正規表現の概要」『SQL Anywhere サーバー SQL リファレンス』と「LIKE 探索条件、REGEXP 探索条件、SIMILAR TO 探索条件」『SQL Anywhere サーバー SQL リファレンス』も参照してください。

- **データベースオプション設定のトランザクションログへの記録**　LOAD 操作時に有効なデータベースオプション設定がトランザクションログに記録されるようになりました。このため、最初の LOAD 操作と、リカバリ時にトランザクションログを適用して実行される最後の LOAD 操作で、date_order や nearest_century などのオプションが違っても、データの矛盾が生じません。

- **オプティマイザーでのインデックス使用の強化**　SQL Anywhere のインデックス機能がいくつかの点で強化されました。これらの新機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

  - **複合インデックススキャンのサポート**　オプティマイザーが強化され、複数のインデックス (最大 4 つまで) を考慮して、ベーステーブルの複数の述部に基づいてテーブルからデータが取り出されるようになりました。これまでは、クエリのインデックスヒントとしてインデックスを 1 つだけ指定できました。SELECT 文の WITH 句の新しいインデックスヒントを使用すると、複合インデックススキャンを使用可能なことを指定できます。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

    複合インデックススキャンという新しいテーブルアクセスアルゴリズムが追加されました。

  - **インデックス専用取得のサポート**　オプティマイザーが強化され、インデックス専用取得がサポートされるようになりました。インデックス専用取得では、テーブル内の対応するローにアクセスする必要がなく、インデックスのデータを使用してクエリが処理されます。オプティマイザーは、可能な場合は常にインデックス専用取得を実行します。INDEX ONLY {ON | OFF} ヒントを使用して、インデックス専用取得を実行するかどうかを制御できます。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **データのロードとアンロードの強化**　データのロードとアンロードが次のように強化されています。

  - **クライアントコンピューターにあるファイルからのデータのロードとファイルへのデータのアンロード**　SQL 文と関数は、データベースサーバーに存在するデータの読み込みと書き込みに使用されます。今回実装された新しい機能では、この機能をクライアントコンピューターにあるファイルにも適用できるようになったので、クライアントのファイルをデータベースサーバーにコピーする必要がなくなりました。このデータの転送は効率的に行われる一方、クライアントコンピューター上のデータに対するセキュリティとアクセス制御が実現します。

    クライアントコンピューターにあるファイルの実際の読み込みはクライアントライブラリによって透過的に行われます。したがって、既存のクライアントアプリケーションで新しい SQL 言語のサポートを使用することで、この新機能をすぐに利用できます。

この新機能を利用するには、クライアントとデータベースサーバーの両方が SQL Anywhere バージョン 11.0.0 であり、またクライアントで Command Sequence 通信プロトコル (CmdSeq) を使用する必要があります。

「クライアントコンピューター上のデータへのアクセス」『SQL Anywhere サーバー SQL の使用法』を参照してください。

○ **変数へのデータのアンロード**　UNLOAD 文が強化されて INTO VARIABLE 句が追加され、変数にデータをアンロードできるようになりました。「UNLOAD 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **別のテーブルのカラムからのデータのロード**　LOAD TABLE 文が強化されて USING COLUMN 句が追加され、別のテーブルのカラムからデータをロードできるようになりました。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「LOAD TABLE 文を使用したデータのインポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

この新機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」 366 ページを参照してください。

○ **値 (BLOB) からのデータのロード**　LOAD TABLE 文が強化されて USING VALUE 句が追加され、関数やシステムプロシージャーの結果などの値式からデータをロードできるようになりました。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「LOAD TABLE 文を使用したデータのインポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

この新機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」 366 ページを参照してください。

○ **LOAD TABLE 文のリカバリとミラーリングの強化**　これまでは、ミラーリングされたデータベース構成で LOAD TABLE 文を使用してファイルからデータをロードできませんでした。これは、LOAD TABLE 文だけがトランザクションログに記録され、ロード対象のデータは記録されなかったからです。また、データベースをリカバリするときに、LOAD TABLE 文を使用してロードされたデータは、リカバリ時に元のロードファイルがないかぎり回復不能でした。

LOAD TABLE 文が強化され、WITH CONTENT LOGGING、WITH ROW LOGGING、WITH FILE NAME LOGGING の 3 つの新しいログオプション句が追加されました。これらの句を使用すると、ロードするデータをトランザクションログに記録するかどうかを制御できます。データベースのミラーリングシステムでは、このデータを使用してミラーデータベースをロードできます。また、リカバリ時にロードファイルがなくてもリカバリが可能になります。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

この機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」 366 ページを参照してください。

● **マテリアライズドビューの強化**　マテリアライズドビューのサポートが次のように強化されました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **即時マテリアライズドビューのサポート**　基本となるテーブル内のデータの変更がマテリアライズドビュー内のデータに影響する場合に、マテリアライズドビューをすぐにリフレッシュするように設定できます。このようにすぐにリフレッシュするビューを「即時ビュー」と呼び、すぐにリフレッシュされないビューは今後「手動ビュー」と呼びます。このリリースの前に作成されたマテリアライズドビューは手動ビューと見なされます。また、新規に作成するマテリアライズドビューはデフォルトで手動ビューになります。

  手動ビューと即時ビューの詳細については、「マテリアライズドビュー」『SQL Anywhere サーバー SQL の使用法』と「再表示タイプの手動または即時への設定」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  この機能をサポートするシステムプロシージャーを使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **複数のマテリアライズドビューの同時リフレッシュ**　これまでは、マテリアライズドビューは一度に 1 つずつリフレッシュする必要がありました。各リフレッシュ操作の間で基本となるデータが変更された場合、マテリアライズドビューの間で矛盾が生じる可能性がありました。このリリースでは、同じデータを使用するマテリアライズドビューをリフレッシュするときに、REFRESH MATERIALIZED VIEW 文にマテリアライズドビューのリストを指定できるようになりました。「REFRESH MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』と「手動マテリアライズドビューのリフレッシュ」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **REFRESH MATERIALIZED VIEW 文の新しい WITH SHARE MODE 句**　REFRESH MATERIALIZED VIEW 文に、新しい句 WITH SHARE MODE が追加されました。このモードでは、リフレッシュ操作中に、基本となるテーブルを他のトランザクションで読み込むことができます。この句を指定すると、基本となるすべてのテーブルの共有テーブルロックが取得されてから、リフレッシュ操作が実行されます。マテリアライズドビューが IMMEDIATE REFRESH と定義されているか、データベースのスナップショットアイソレーションが有効になっていないかぎり、デフォルトのモードは WITH SHARE MODE になります。リフレッシュのデフォルト動作の詳細については、「REFRESH MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ファイルまたは BLOB 文字列の内容の問い合わせのサポート**　FROM 句の新しい OPENSTRING 句を使用して、ファイルや BLOB 文字列のデータを問い合わせることができるようになりました。OPENSTRING 句では、問い合わせるオブジェクト、またデータのスキーマやその他の解析情報を指定できます。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  OPENSTRING 操作が実行されると、新しいプラン項目 OpenString が実行プランに表示されます。

- **圧縮インデックスのサポートの向上**　圧縮インデックスのサポートが向上された結果、データベースをアンロードしてから再ロードして再構築すると、再構築されたデータベースが元のデータベースよりも小さくなる場合があります。このデータベースのサイズの縮小は、問題や、データが失われたことを示すものでは**ありません**。

- **ミラーサーバーで実行されているデータベースへの読み込み専用アクセス**　データベースミラーリングを使用している場合に、ミラーサーバーで実行されているデータベースに接続できるようになりました。したがって、プライマリサーバーの可用性を維持しながら、リソースを大量に使用する可能性があるレポート操作の負荷をミラーサーバーに割り当てることができます。ミラーデータベースに接続するには、読み込み専用のミラーデータベースへのアクセスに使用できる -sm サーバーオプションでデータベースサーバー名を指定します。「-sm dbsrv12 データベースオプション (旧式)」『SQL Anywhere サーバー データベース管理』と「ミラーサーバーで実行されているデータベースへの読み込み専用アクセスの設定」『SQL Anywhere サーバー データベース管理』を参照してください。

データベース接続

次に、SQL Anywhere バージョン 11.0.0 でのデータベース接続の強化を示します。

- **AppInfo 接続パラメーターの強化**　AppInfo 接続パラメーターで OSUSER キーがサポートされるようになりました。このキーは、クライアントプロセスに関連付けられたオペレーティングシステムユーザー名を返します。Linux と Solaris で EXE キーがサポートされるようになりました。「AppInfo (APP) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Elevate 接続パラメーター**　Elevate 接続パラメーターは、Windows Vista で自動的に起動した SQL Anywhere データベースサーバーを昇格します。「Elevate 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **NewPassword 接続パラメーター [NEWPWD]**　NewPassword 接続パラメーターを使用すると、DBA に依頼しなくてもユーザーが各自でパスワードを変更できます。パスワードの有効期限が切れていてもかまいません。「NewPassword (NEWPWD) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **プリフェッチの強化**　PrefetchBuffer (PBUF) 接続パラメーターのデフォルト値が変更されました。デフォルト値は、Windows Mobile では 64 KB、その他のプラットフォームでは 512 KB になりました。この接続パラメーターには、64 KB ～ 8 MB の値を指定できます。「PrefetchBuffer (PBUF) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

  これまでのバージョンでは、プリフェッチされる最大ロー数は、プリフェッチできる最大データ量に基づいていました。このリリースでは、プリフェッチされる最大ロー数には、プリフェッチされる実際のデータ量に加えて、PrefetchBuffer 接続パラメーターで指定されたデータ制限が考慮されるようになりました。その結果、カラム内のデータ量が、ホスト変数長と記述長の両方よりも大幅に少ない場合に、パフォーマンスが大きく向上する可能性があります。

  また、パフォーマンスが向上する可能性が高い場合に、プリフェッチするロー数が動的に増えます。「プリフェッチ」『SQL Anywhere サーバー プログラミング』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バックアップとリカバリ

次に、SQL Anywhere バージョン 11.0.0 でのバックアップとリカバリの強化を示します。

- **Microsoft ボリュームシャドウコピーサービス (VSS) のサポート**　SQL Anywhere は、Microsoft ボリュームシャドウコピーサービス (VSS) との互換性があります。VSS を使用するには、既存のデータベースをすべて再構築する必要があります。「SQL Anywhere ボリュームシャドウコピーサービス (VSS)」『SQL Anywhere サーバー データベース管理』を参照してください。

セキュリティ

次に、SQL Anywhere バージョン 11.0.0 でのセキュリティの強化を示します。

- **テーブルの暗号化が有効になっているときの ISYSUSER と ISYSEXTERNLOGIN の各システムテーブルの暗号化**　これまでは、データベースを暗号化するときや、テーブルの暗号化を有効にしてデータベースを作成するときに、ISYSCOLSTAT システムテーブルが自動的に暗号化されました。このリリースでは、さらにセキュリティを強化するため、ISYSUSER と ISYSEXTERNLOGIN の各システムテーブルも暗号化されるようになりました。

- **監査の強化**　このリリースでは、Sybase Central を使用して監査を制御できるようになりました。DBA 権限を持つユーザーは、**[データベースのプロパティ]** ウィンドウから、監査を有効または無効にしたり、監査する情報を指定したりできます。監査情報は、Sybase Central の右ウィンドウ枠の **[監査]** タブで確認できます。「監査の制御」『SQL Anywhere サーバー データベース管理』と「監査情報の取り出し」『SQL Anywhere サーバー データベース管理』を参照してください。

  監査が有効になっているときは、接続失敗のエラーが、失敗の理由とともにログに記録されます。

- **256 ビットの AES 暗号化のサポート**　SQL Anywhere で、データベース、テーブル、ファイル、データに対して 256 ビットの AES 暗号化がサポートされるようになりました。この強化は、次に示す複数の領域に影響します。

  - **データベースとテーブルの暗号化**　CREATE DATABASE 文の ENCRYPTION 句に、AES256 と AES256_FIPS を指定できるようになりました。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

    初期化ユーティリティ (dbinit) とアンロードユーティリティ (dbunload) の -ea オプションに、AES256 と AES256_FIPS を指定することもできます。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』と「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **FIPS 認定のアルゴリズム**　256 ビットの FIPS 認定の AES アルゴリズムを使用できるようになりました。「-fips dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **データの暗号化と復号化**　ENCRYPT 関数と DECRYPT 関数を使用してデータを暗号化するときに、AES256 と AES256_FIPS を指定できるようになりました。「ENCRYPT 関数 [文

字列]」『SQL Anywhere サーバー SQL リファレンス』と「DECRYPT 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **データベース、トランザクションログ、DB 領域の暗号化されたコピーの作成** CREATE ENCRYPTED FILE 文を使用して、暗号化されているか、暗号化されてないデータベース、トランザクションログ、または DB 領域の暗号化されたコピーを作成するとき、256 ビットの AES アルゴリズム (AES256 または AES256_FIPS) を指定できるようになりました。「CREATE ENCRYPTED FILE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DBTools での 256 ビットの AES 暗号化のサポート** a_create_db と an_unload_db の各構造体が拡張され、encryption_algorithm メンバーの値として AES256 と AES256_FIPS がサポートされるようになりました。a_create_db 構造体 [データベースツール]『SQL Anywhere サーバー プログラミング』と an_unload_db 構造体 [データベースツール]『SQL Anywhere サーバー プログラミング』を参照してください。

参照：

- 「データベースの暗号化と復号化」『SQL Anywhere サーバー データベース管理』
- 「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』

- **jConnect と Open Client のパスワード暗号化のサポート** jConnect 接続と Open Client 接続でパスワードの暗号化がサポートされるようになりました。次の項を参照してください。

  - 「jConnect JDBC ドライバー」『SQL Anywhere サーバー プログラミング』
  - 「JDBC クライアントの配備」『SQL Anywhere サーバー プログラミング』
  - 「SQL Anywhere における Open Client の既知の制限」『SQL Anywhere サーバー プログラミング』

データベースのパーミッションと権限

次に、SQL Anywhere の新しい、または強化されたパーミッションと権限を示します。これらの変更点を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **一部の権限の継承サポート** SQL Anywhere で、次のように一部の権限の継承がサポートされるようになりました。

  ○PROFILE、READCLIENTFILE、READFILE、WRITECLIENTFILE は継承可能

  ○DBA、BACKUP、RESOURCE の各権限は継承不可

  「権限」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいプロファイル権限** これまでは、ユーザーがアプリケーションプロファイリングや診断トレーシングを実行するには、DBA 権限が必要でした。このリリースでは、プロファイル権限を持つユーザーもこれらの操作を行うことができます。プロファイル権限の全パーミッションのリストについては、「プロファイル権限」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい READFILE 権限**　READFILE 権限を使用すると、ユーザーは SELECT 文の OPENSTRING 句を使用してファイルから選択できます。「READFILE 権限」『SQL Anywhere サーバー データベース管理』と「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい READCLIENTFILE 権限**　READCLIENTFILE 権限を使用すると、ユーザーはクライアントコンピューターにあるファイルを読み込むことができます。たとえば、ユーザーが LOAD TABLE 文を使用して、クライアントコンピューターにあるファイルからデータをロードするには、READCLIENTFILE 権限が必要です。「READCLIENTFILE 権限」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい WRITECLIENTFILE 権限**　WRITECLIENTFILE 権限を使用すると、ユーザーはクライアントコンピューターにあるファイルに書き込むことができます。たとえば、ユーザーが UNLOAD TABLE 文を使用して、クライアントコンピューターにあるファイルにデータをアンロードするには、WRITECLIENTFILE 権限が必要です。「WRITECLIENTFILE 権限」『SQL Anywhere サーバー データベース管理』を参照してください。

- **DB 領域の CREATE パーミッションのサポート**　SQL Anywhere で、指定の DB 領域にデータベースオブジェクトを作成できる CREATE ON パーミッションがサポートされるようになりました。CREATE パーミッションは、ユーザーに割り当てるか、グループから継承できます。「GRANT 文」『SQL Anywhere サーバー SQL リファレンス』と「パーミッション」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースユーティリティ

次に、SQL Anywhere バージョン 11.0.0 でのデータベースユーティリティの強化を示します。

- **設定ファイルの強化**　設定ファイル内でアンパサンド (&) を使用して、前のトークンが次の行に続いていることを指定できるようになりました。「設定ファイル」『SQL Anywhere サーバー データベース管理』を参照してください。

- **アンロードユーティリティ (dbunload) の強化**　dbunload が次のように強化されています。

  ○新しいオプション -cp が追加され、dbunload でデータ出力ファイルを圧縮できるようになりました。

  ○これまでは、再ロードオプション -an または -ar を指定しないで暗号化オプション -ek、-ep、または -ea を指定すると、エラーが返されました。このリリースでは、dbunload で暗号化オプションが受け入れられ、作成される出力ファイルに適用されるようになりました。

  ○-g オプションによって、MANUAL REFRESH と定義されているテキストインデックスが再表示されるようになりました。「テキストインデックスの概念と参照」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  ○MANUAL REFRESH と定義されているテキストインデックスは、デフォルトで再ロード時に初期化されません。これらのテキストインデックスを初期化したい場合は、dbunload -g オプションを指定できます。

○-no オプションを使用して、オブジェクト定義をオブジェクトタイプ別にアルファベット順でアンロードできます。これはデータベースの *reload.sql* ファイルを比較する場合に便利です。

「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **検証ユーティリティ (dbvalid) の強化**　これまで、dbvalid ユーティリティでは、デフォルトでテーブルとマテリアライズドビューがすべて検証されていました。このリリースでは、dbvalid によって VALIDATE DATABASE 文も実行されます。

  検証ユーティリティを実行してデータベースを自動的に起動すると、データベースは読み込み専用モードで起動します。このようにすることで、バックアップとリカバリのプランの一環としてデータベースが検証されている場合にデータベースを変更できないようになっています。

  「検証ユーティリティ (dbvalid)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ログ変換ユーティリティ (dbtran) で破損が検出された場合の生成された *.sql* ファイルの保守**　ログ変換ユーティリティの -k オプションを使用すると、トランザクションログファイルが破損しているためログファイルの変換に失敗した場合に不完全な *.sql* ファイルを削除しないことを指定できます。「ログ変換ユーティリティ (dbtran)」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースオプション

次に、SQL Anywhere バージョン 11.0.0 でのデータベースオプションの強化を示します。

- **allow_read_client_file オプション**　このオプションは、クライアントコンピューター上のファイルの読み込みを許可するかどうかを制御します。「allow_read_client_file オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **allow_write_client_file オプション**　このオプションは、クライアントコンピューターへのファイルの書き込みを許可するかどうかを制御します。「allow_write_client_file オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **login_procedure オプション**　パスワードの有効期限が切れていることを示すエラーメッセージを通知できるようになりました。「login_procedure オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **max_priority オプション**　このオプションは、データベース接続の最高優先度レベルを制御します。「max_priority オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **priority オプション**　このオプションは、接続からの要求を実行する優先度レベルを制御します。「priority オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **query_mem_timeout オプション**　このオプションは、要求にメモリが付与されるまで待つ時間を制御します。「query_mem_timeout オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースサーバーオプション

次に、SQL Anywhere バージョン 11.0.0 でのデータベースサーバーオプションの強化を示します。

- **-es サーバーオプション**　SQL Anywhere のこれまでのバージョンでは、-ec オプションを指定してデータベースサーバーが起動され (トランスポートレイヤーセキュリティのサポートのため)、許可する暗号化プロトコルのリストに NONE または SIMPLE が含まれなかった場合、共有メモリポートが起動しませんでした。これは、このポートでトランスポートレイヤーセキュリティがサポートされていないからです。このため、データベースサーバーへの接続は、強力な暗号化を使用して TCP/IP で行う必要がありました。

  -es サーバーオプションは、共有メモリを経由した、暗号化されていない接続を許可するようにデータベースサーバーに指定します。「-es dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-gb サーバーオプション**　サーバープロセスの優先度クラスを制御する -gb サーバーオプションが、Windows に加えて UNIX でサポートされるようになりました。「-gb dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-im サーバーオプション**　アプリケーションで、データベースの操作内容がすべて失われてもかまわない場合は、データベースを完全にメモリ内で実行できます。この機能は、データが頻繁に挿入される、高速のテンポラリデータストアとして SQL Anywhere を使用する場合を対象としています。「-im dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **クライアントコンピューターにあるファイルの読み込みと書き込み**　-sf サーバーオプションを使用して、クライアントコンピューターにあるファイルの読み込みと書き込みの機能を制御できるようになりました。「-sf dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-um サーバーオプション**　-um オプションを使用して、*DBLauncher.app* インスタンスが実行中の場合にこのインスタンスに接続し、*DBLauncher.app* 内の新しいウィンドウにデータベースサーバーメッセージを表示することができます。このオプションは Mac OS X だけに適用されます。「-um dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Windows パフォーマンスモニターのオプション**　Windows パフォーマンスモニターをさらに詳細に設定するため、次のサーバーオプションが追加されました。

  - **-ks オプション**　パフォーマンスモニターで、データベースサーバーからカウンター値を収集するために使用される共有メモリの作成を無効にします。「-ks dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-ksc オプション**　パフォーマンスモニターでモニターできる接続の最大数を指定します。「-ksc dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
- **-ksd オプション**　パフォーマンスモニターでモニターできるデータベースの最大数を指定します。「-ksd dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

プロパティとパフォーマンスモニターの統計値

次に、SQL Anywhere バージョン 11.0.0 でのプロパティとパフォーマンスモニターの統計の強化を示します。

- **新しい接続プロパティ**　このリリースには、次の接続プロパティが追加されています。
  - allow_read_client_file
  - allow_write_client_file
  - AuthType
  - CacheReadWorkTable
  - ClientNodeAddress
  - DiskReadWorkTable
  - DiskSyncRead
  - DiskSyncWrite
  - DiskWaitRead
  - DiskWaitWrite
  - DiskWriteHint
  - DiskWriteHintPages
  - LockIndexID
  - LockRowID
  - max_priority
  - OSUser
  - priority
  - query_mem_timeout
  - QueryMemActiveCurr
  - QueryMemExtraAvail
  - QueryMemGrantFailed
  - QueryMemGrantGranted
  - QueryMemGrantWaiting
  - QueryMemGrantRequested
  - QueryMemWaited
  - ServerNodeAddress
  - ReadHint
  - ReadHintScatter

  これらのプロパティの説明については、「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しいデータベースサーバープロパティ**　このリリースには、次のデータベースサーバープロパティが追加されています。

  - DiskRetryRead
  - DiskRetryReadScatter
  - DiskRetryWrite
  - EventTypeDesc
  - EventTypeName
  - HttpAddresses
  - HttpsAddresses
  - HttpNumActiveReq
  - HttpNumConnections
  - HttpNumSessions
  - HttpsNumActiveReq
  - HttpsNumConnections
  - MaxEventType
  - MaxRemoteCapability
  - MessageCategoryLimit
  - OptionWatchAction
  - OptionWatchList
  - QueryMemActiveCurr
  - QueryMemActiveEst
  - QueryMemActiveMax
  - QueryMemExtraAvail
  - QueryMemGrantBase
  - QueryMemGrantBaseMI
  - QueryMemGrantExtra
  - QueryMemGrantFailed
  - QueryMemGrantGranted
  - QueryMemGrantWaiting
  - QueryMemGrantRequested
  - QueryMemPages
  - QueryMemPercentOfCache
  - QueryMemWaited
  - ReadHintScatterLimit
  - RemoteCapability
  - StreamsUsed
  - TcpIpAddresses
  - WebClientLogFile
  - WebClientLogging

  これらのプロパティの説明については、「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいデータベースプロパティ**　このリリースには、次のデータベースプロパティが追加されています。

  - AlternateMirrorServerName
  - CacheReadWorkTable
  - DiskReadWorkTable
  - DiskRetryReadScatter
  - DiskSyncRead
  - DiskSyncWrite
  - DiskWaitRead
  - DiskWaitWrite
  - DiskWriteHint
  - DiskWriteHintPages
  - HasEndianSwapFix
  - MirrorMode
  - ReadHint
  - ReadHintScatter

  これらのプロパティの説明については、「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **パフォーマンスモニター統計値の追加**　このリリースには、次のパフォーマンスモニターの統計値が追加されています。

  - キャッシュ読み込み：ワークテーブル
  - ディスク読み込み：ワークテーブル

システムプロシージャーとファンクション

次に、SQL Anywhere バージョン 11.0.0 で追加されたシステムプロシージャーとファンクションの強化を示します。

- **sa_get_dtt_groupreads システムプロシージャー**　新しい sa_get_dtt_groupreads システムプロシージャーを使用して、データベースサーバーに対してグループ読み込みを発行するコストを推定できます。「sa_get_dtt_groupreads システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **PROPERTY_NAME ファンクションの強化**　指定した接続レベルで、指定したプロパティ ID を持つプロパティの名前を返すようになりました。「PROPERTY_NAME 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **READ_CLIENT_FILE ファンクション**　新しい READ_CLIENT_FILE ファンクションは、クライアントコンピューターにある、指定したファイルからデータを読み込みます。「READ_CLIENT_FILE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **WRITE_CLIENT_FILE ファンクション**　新しい WRITE_CLIENT_FILE ファンクションは、クライアントコンピューターにある、指定したファイルにデータを書き込みます。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「WRITE_CLIENT_FILE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **REGEXP_SUBSTR ファンクション**　新しい REGEXP_SUBSTR ファンクションを使用すると、文字列内で部分文字列を検索できます。この新しいファンクションでは、引数として正規表現を指定できます。「REGEXP_SUBSTR 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_char_terms システムプロシージャー**　新しい sa_char_terms システムプロシージャーは、CHAR 文字列を単語に分解し、各単語とその位置を返します。「sa_char_terms システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_nchar_terms システムプロシージャー**　新しい sa_nchar_terms システムプロシージャーは、NCHAR 文字列を単語に分解し、各単語とその位置を返します。「sa_nchar_terms システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_refresh_text_indexes システムプロシージャー**　新しい sa_refresh_text_indexes システムプロシージャーは、MANUAL REFRESH または AUTO REFRESH として定義されているすべてのテキストインデックスを再表示します。「sa_refresh_text_indexes システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_text_index_stats システムプロシージャー**　新しい sa_text_index_stats システムプロシージャーは、データベース内のすべてのテキストインデックスに関する統計情報を返します。これには、最終再表示時刻と保留中の変更のサイズが含まれます。「sa_text_index_stats システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_text_index_vocab システムプロシージャー**　新しい sa_text_index_vocab システムプロシージャーは、テキストインデックスに含まれるすべての単語と、各単語が含まれるインデックス化された値の合計数のリストを返します。「sa_text_index_vocab システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  2 つの新しいシステムプロシージャー sa_internal_text_index_vocab と sa_internal_text_index_postings も追加されていますが、これらは sa_text_index_vocab システムプロシージャーでのみ使用されます。

- **sa_text_index_postings システムプロシージャー**　この新しいシステムプロシージャーは内部でのみ使用されます。

- **sa_text_index_handles システムプロシージャー**　この新しいシステムプロシージャーは内部でのみ使用されます。

- **sa_get_user_status システムプロシージャー**　新しい sa_get_user_status システムプロシージャーを使用すると、ユーザーの現在のログインステータスを確認できます。「sa_get_user_status システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **インボーカーとしてのプロシージャーやファンクションの実行**　プロシージャーまたはファンクションを作成するときに、そのプロシージャーまたはファンクションを、呼び出したユーザー (インボーカー) が呼び出したかのように実行するか、それを作成したユーザー (デファイナー) が呼び出したかのように実行するかを指定できます。これを指定するには、CREATE

PROCEDURE 文または CREATE FUNCTION 文の SQL SECURITY 句を使用します。「CREATE FUNCTION 文」『SQL Anywhere サーバー SQL リファレンス』と「CREATE PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

この変更は、外部プロシージャーと関数にも適用されます。

- **sa_disk_free_space システムプロシージャー** sa_disk_free_space システムプロシージャーで、total_space という新しいカラムが返されるようになりました。このカラムは、DB 領域があるドライブで使用可能な合計ディスク領域を示します。バージョン 11.0.0 より前の SQL Anywhere で作成されたデータベースでは、データベースをアップグレードするまで total_space カラムは返されません。「sa_disk_free_space システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_external_library_unload システムプロシージャー** 新しいシステムプロシージャー sa_external_library_unload が追加され、使用していない外部ライブラリをアンロードできるようになりました。「sa_external_library_unload システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_index_density システムプロシージャーがスキュー量を返す** sa_index_density システムプロシージャーが強化され、インデックス内にあるスキュー量を返すようになりました。スキューが多いと、バランスがとれているインデックスと比べてパフォーマンスが低くなる可能性があります。「インデックスの断片化とスキューの削減」『SQL Anywhere サーバー SQL の使用法』と「sa_index_density システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_materialized_view_info システムプロシージャーの強化** sa_materialized_view_info で返される Status カラムの情報が、Status と DataStatus の 2 つのカラムに分割されました。Status には、ビューが有効であるか、無効であるかに関する情報が返されます。新しい DataStatus カラムには、ビュー内にデータがあるかどうか、またデータが最新かどうかに関する情報が返されます。また、ビューが手動ビューであるか、即時ビューであるかを示す新しいカラム RefreshType が追加されています。「sa_materialized_view_info システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_materialized_view_can_be_immediate システムプロシージャー** 新規に作成するマテリアライズドビューはデフォルトで手動ビューになりますが、即時ビューの制約に違反していなければ、即時ビューに変更できます。新しい sa_materialized_view_can_be_immediate システムプロシージャーを使用すると、手動ビューを即時ビューに変更できるかどうかをテストできます。「sa_materialized_view_can_be_immediate システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』と「マテリアライズドビューを手動から即時に変更する際の制限」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **sa_post_login_procedure システムプロシージャー** ユーザーのパスワードの有効期限が近いときに警告するかどうかを指定できる新しいシステムプロシージャーが追加されました。「sa_post_login_procedure システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **EVENT_PARAMETER ファンクションの強化** EVENT_PARAMETER ファンクションで、DisconnectReason として abnormal がサポートされるようになりました。この新しい理由は、データベースへの接続を切断する前にクライアントアプリケーションが異常終了したか、ク

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ライアントコンピューターとサーバーコンピューターの間で通信エラーが発生した結果、切断が発生したことを示します。「EVENT_PARAMETER 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_server_option システムプロシージャーの強化** sa_server_option システムプロシージャーに OptionWatchList と OptionWatchAction の 2 つのプロパティが追加されました。これらのプロパティを使用して、データベースオプションの設定が変更されようとしていないかどうかをモニターし、そのときの処置を指定できます。「オプション設定のモニタリング」『SQL Anywhere サーバー データベース管理』と「sa_server_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_db_properties システムプロシージャーの強化** sa_db_properties システムプロシージャーが、NULL 値を持つ有効なプロパティを返すようになりました。「sa_db_properties システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_conn_properties システムプロシージャーの強化** sa_conn_properties システムプロシージャーが、NULL 値を持つ有効なプロパティを返すようになりました。「sa_conn_properties システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

SQL 文

次に、SQL Anywhere バージョン 11.0.0 での SQL の強化を示します。

- **新しい CALIBRATE GROUP READ 句、ALTER DATABASE 文** ALTER DATABASE 文の新しい CALIBRATE GROUP READ 句を使用して、テンポラリ DB 領域に対してグループ読み込みの調整を実行できます。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい CHECK 句、CREATE MATERIALIZED VIEW 文** CREATE MATERIALIZED VIEW 文の新しい CHECK 句を使用して、ビューの作成前に文を検証できます。「CREATE MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい RECOMPILE 句、ALTER FUNCTION 文** ALTER FUNCTION 文に新しい句 RECOMPILE が追加され、ユーザー定義関数を再コンパイルできるようになりました。「ALTER FUNCTION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい RECOMPILE 句、ALTER PROCEDURE 文** ALTER PROCEDURE 文に新しい句 RECOMPILE が追加され、ストアドプロシージャーを再コンパイルできるようになりました。「ALTER PROCEDURE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい REFRESH 句、ALTER MATERIALIZED VIEW 文** ALTER MATERIALIZED VIEW 文に新しい句 REFRESH が追加され、マテリアライズドビューのリフレッシュ方法を指定できるようになりました。「ALTER MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **リカバリとミラーリングをサポートするための LOAD TABLE 文の強化** リカバリとミラーリングをサポートするため、次の句が LOAD TABLE 文に追加されました。

- **WITH CONTENT LOGGING 句**　WITH CONTENT LOGGING 句は、データソースの内容をトランザクションログに記録することをデータベースサーバーに指定します。データは、入力が LOAD TABLE によって処理されるときに小さいチャンクで記録されます。これらのチャンクは、ミラーリングデータベースで、またトランザクションログを使用したリカバリ時に、ローに再構成できます。WITH CONTENT LOGGING 句は、後でリカバリするために元のデータファイルを維持しないほうがいい場合に便利です。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **WITH ROW LOGGING 句**　WITH ROW LOGGING 句は、ロードされているすべてのローを、一連の INSERT 文として記録することをデータベースサーバーに指定します。このレベルは、同期するデータベースや、ロード先のテーブルに、計算カラムや CURRENT TIMESTAMP のデフォルトなど、非決定的な値が含まれる場合に最適です。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **WITH FILE NAME LOGGING 句**　WITH FILE NAME LOGGING 句は、LOAD TABLE 文だけを記録するようにデータベースサーバーに指定します。これがデフォルトの動作であり、SQL Anywhere のこれまでのバージョンでのロギングの動作と同じです。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **クライアントファイルのロードとアンロードのための新しい句**　クライアントファイルの新しいロード／アンロード機能をサポートするため、LOAD TABLE 文と UNLOAD TABLE 文が強化されました。

  - **LOAD TABLE 文の新しい USING CLIENT FILE 句**　クライアントコンピューターにあるファイル内のデータを使用してテーブルをロードできます。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **UNLOAD TABLE 文の新しい INTO CLIENT FILE 句**　データのアンロード先としてクライアントコンピューターにあるファイルを指定できます。「UNLOAD 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しいログインポリシー文**　新しいログインポリシー機能をサポートするために、次の文が追加されました。

  - CREATE LOGIN POLICY 文：「CREATE LOGIN POLICY 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER LOGIN POLICY 文：「ALTER LOGIN POLICY 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - DROP LOGIN POLICY 文：「DROP LOGIN POLICY 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - CREATE USER 文：「CREATE USER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER USER 文：「ALTER USER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - DROP USER 文：「DROP USER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **全文検索用の新しい文と句**　新しい全文検索機能をサポートするために、次の文が追加されました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい CONTAINS 探索条件**　CONTAINS 探索条件を使用して、指定するカラムのリスト内に、指定する単語やフレーズがあるかどうかを確認できます。CONTAINS 探索条件は、TRUE または FALSE を返します。複数の単語やフレーズを検索する場合は、さまざまなブール演算子で結合できます。「CONTAINS 探索条件」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SELECT 文の FROM 句内の新しい CONTAINS 句**　CONTAINS 句は、SELECT 文の FROM 句内で指定します。この句は CONTAINS 探索条件に似ていますが、一致するカラムごとのスコアと、一致する各ローの合計スコアも返します。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE TEXT CONFIGURATION 文**　この文は、テキスト設定オブジェクトを作成します。テキスト設定オブジェクトは、テキストインデックスの特性を制御する一連の設定です。「CREATE TEXT CONFIGURATION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER TEXT CONFIGURATION 文**　この文は、テキスト設定オブジェクトを変更します。「ALTER TEXT CONFIGURATION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DROP TEXT CONFIGURATION 文**　この文は、テキスト設定オブジェクトを削除します。「DROP TEXT CONFIGURATION 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE TEXT INDEX 文**　この文は、テキストインデックスを作成します。テキストインデックスには、インデックス化されたすべてのカラム内の全単語の出現箇所の詳細な位置情報が含まれます。「CREATE TEXT INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER TEXT INDEX 文**　この文は、テキストインデックスを変更します。「ALTER TEXT INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DROP TEXT INDEX 文**　この文は、データベースからテキストインデックスを削除します。「DROP TEXT INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **REFRESH TEXT INDEX 文**　この文は、テキストインデックスを再表示します。「REFRESH TEXT INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **TRUNCATE TEXT INDEX 文**　この文は、テキストインデックスからデータをトランケートします。「TRUNCATE TEXT INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER EVENT 文の強化**　ALTER EVENT...SET HIDDEN 文を使用して、イベントハンドラーの定義を隠すことができるようになりました。この文を使用すると、ISYSEVENT システムテーブルの action カラムに格納されているイベントハンドラーの定義が難読化されます。「ALTER EVENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **BEGIN SNAPSHOT 文**　BEGIN SNAPSHOT 文を使用すると、スナップショットアイソレーションのためスナップショットをいつ開始するかを制御できます。「BEGIN SNAPSHOT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CASE 文と CASE 式の強化**　互換性の向上のため、CASE 文と CASE 式を END または END CASE で終了できるようになりました。「CASE 文」『SQL Anywhere サーバー SQL リファレンス』と「CASE 式」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **COMMENT 文の強化**　ログインポリシーテーブルと DB 領域にコメントを追加できるようになりました。次の項を参照してください。

  ○「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』
  ○「ログインポリシー」『SQL Anywhere サーバー データベース管理』
  ○「追加の DB 領域の考慮事項」『SQL Anywhere サーバー データベース管理』

- **CREATE MATERIALIZED VIEW 文の強化**　CREATE MATERIALIZED VIEW 文の新しい IMMEDIATE REFRESH 句を使用して、基本となるデータが変更されたときにリフレッシュされるマテリアライズドビューを作成できるようになりました。「CREATE MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DESCRIBE 文の強化**　Interactive SQL の DESCRIBE 文を使用して、Interactive SQL が接続しているデータベースまたはデータベースサーバーに関する情報を取得できるようになりました。「DESCRIBE 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **IF 文と IF 式の強化**　互換性の向上のため、IF 文と IF 式を ENDIF または END IF で終了できるようになりました。「IF 文」『SQL Anywhere サーバー SQL リファレンス』と「IF 式」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **LOAD TABLE 文の強化**　LOAD TABLE 文の新しい COMPRESSED 句または ENCRYPTED 句を使用して、入力ファイルのデータが圧縮されていることや、暗号化されていることを指定できるようになりました。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SELECT 文の強化**

  ○ **INDEX 句の強化**　INDEX 句を使用してインデックスヒントを指定するときに、データベースサーバーで使用する必要があるインデックスを 4 つまで指定できるようになりました。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○ **新しい INDEX ONLY 句**　INDEX 句を使用してインデックスヒントを指定するときに、必要に応じて INDEX ONLY 句を指定して、データベースサーバーでインデックス専用取得 (インデックスのデータのみを使用してクエリを処理する) を行うかどうかを制御できます。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○ **新しい CROSS APPLY 句と OUTER APPLY 句**　SELECT 文が拡張され、FROM 句で適用式 (CROSS APPLY 句と OUTER APPLY 句) がサポートされるようになりました。適用式は、右側が左側に依存するジョインを簡単に指定できる方法です。たとえば、適用式を使用して、テーブル式内のローごとに 1 回ずつプロシージャーまたは派生テーブルを評価で

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

きます。「適用式から生成されるジョイン」『SQL Anywhere サーバー SQL の使用法』と「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- ○ **新しい OPENSTRING 句** 新しい OPENSTRING 句を使用すると、SELECT 文を使用してファイル内のデータを問い合わせることができます。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **イベントの作成時、変更時、削除時、コメント時の所有者の指定** CREATE EVENT、ALTER EVENT、DROP EVENT、COMMENT ON EVENT の各文で、必要に応じて所有者を指定できるようになりました。次の項を参照してください。

  - ○「CREATE EVENT 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「ALTER EVENT 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「DROP EVENT 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』

- **UNLOAD 文の強化** UNLOAD 文の COMPRESSED 句または ENCRYPTED 句を使用して、アンロード対象のデータの圧縮や暗号化を行うかどうかを指定できるようになりました。「UNLOAD 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  これらの句を使用して圧縮または暗号化したファイルは、LOAD TABLE などを使用して、SQL Anywhere 11.0.0 のデータベースサーバーでのみロードできます。その他のツールを使用して圧縮または暗号化したファイルは SQL Anywhere で使用できません。

- **UPDATE 文の強化** 検索と位置付け更新のときに、SET 句を使用してカラム値をデフォルト値に設定できるようになりました。「UPDATE 文」『SQL Anywhere サーバー SQL リファレンス』と「UPDATE (位置付け) 文 [ESQL] [SP]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **OPTION 句の拡張** INSERT、UPDATE、DELETE、SELECT、UNION、EXCEPT、INTERSECT の各文の OPTION 句で、user_estimates データベースオプションの設定を上書きできるようになりました。次の項を参照してください。

  - ○「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「UPDATE 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「UNION 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「EXCEPT 文」『SQL Anywhere サーバー SQL リファレンス』
  - ○「INTERSECT 文」『SQL Anywhere サーバー SQL リファレンス』

プログラミングインターフェイス

次に、SQL Anywhere バージョン 11.0.0 でのプログラミングインターフェイスの強化を示します。

- **新しい SQL Anywhere C API** SQL Anywhere C アプリケーションプログラミングインターフェイス (API) により、PHP、Perl、Python、Ruby など、複数のインタプリター型プログラミング言語での C や C++ ラッパードライバーの作成が簡単になります。SQL Anywhere C API は DBLIB ライブラリの上層に位置し、Embedded SQL で実装されています。

DBLIB に代わるものではありませんが、この API は、C や C++ によるアプリケーションの作成を簡単にします。SQL Anywhere C API を使用するのに、Embedded SQL に関する高度な知識は必要ありません。「SQL Anywhere C API のサポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **新しい Python データベース API (sqlanydb)** 新しい Python データベース API (sqlanydb) を使用すると、Python で作成されたスクリプトから SQL Anywhere データベースにアクセスできるようになります。sqlanydb モジュールは、Python データベース API 仕様バージョン 2.0 を拡張して実装しています。「Python サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **外部環境** SQL Anywhere で、Java、Perl、PHP、CLR、Embedded SQL、ODBC の 6 つの外部ランタイム環境がサポートされるようになりました。これまで SQL Anywhere では、C または C++ で記述されたコンパイル済みネイティブ関数を呼び出すことができました。ただし、これらのプロシージャーがサーバーで実行されるとき、ダイナミックリンクライブラリまたは共有オブジェクトが常にデータベースサーバーによってロードされ、ネイティブ関数への呼び出しがデータベースサーバーによって行われていました。この方法には、ネイティブ関数が原因で障害が発生した場合、データベースサーバーがクラッシュするというリスクがあります。データベースサーバーの外部環境でコンパイル済みネイティブ関数を実行できると、サーバーへのこれらのリスクをなくすことができます。「SQL Anywhere 外部環境のサポート」『SQL Anywhere サーバー プログラミング』を参照してください。

  この新機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」 366 ページを参照してください。

- **PHP 外部環境のサポート** SQL Anywhere 11.0.0 には、5.1.1 ～ 5.1.6 と 5.2.0 ～ 5.2.6 を含むさまざまな PHP バージョン用の構築済みバイナリが含まれます。このいずれかのバージョンをサーバーコンピューターにインストールしてある場合は、PHP 外部環境を構築しないで SQL Anywhere の構築済みバイナリを使用してください。Linux と Solaris 用には、32 ビットと 64 ビットの両方のバージョンのバイナリが用意されています。Windows とその他のシステム用には、32 ビットのバージョンだけがあります。

  上記とは異なる PHP バージョンをインストールしてある場合は、ソフトウェアを構築するか、PHP バージョンを、SQL Anywhere の構築済みバージョンと同じバージョンに切り替える必要があります。SQL Anywhere PHP モジュールの構築方法については、「SQL Anywhere PHP 拡張」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Perl 外部環境のサポート** Perl 外部環境を使用する前に、SQL Anywhere Perl DBD ドライバーのバージョンを更新することが非常に重要です。Perl DBD ドライバーを更新しなかった場合、サーバー側 Perl は機能しません。

  また、PHP とは異なり、SQL Anywhere には Perl のさまざまなバージョン用の構築済みバイナリは含まれません。SQL Anywhere Perl DBD ドライバーのソースコードは *%SQLANY11%¥SDK¥perl* にあります。SQL Anywhere Perl DBD ドライバーの構築方法については、「Perl DBI サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Web サーバーでの UTF-8 URL のサポート** これまでは、要求 URL (または要求の本文内の application/x-www-form-urlencoded データ) 内のパーセント記号 (%) でコード化されたデータが、Web サーバーによってデータベースの文字セットに復号化されていました。このリリー

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

スでは、パーセント記号 (%) でコード化されたデータの内容で、UTF-8 シーケンスの有無がテストされ、最大エクステントに基づいてデータベースの文字セットに変換されるようになりました。UTF-8 以外でコード化されたデータは、復号化され、すでにデータベースの文字セットになっているかのように処理されます。

クライアント HTTP アプリケーションではパーセント記号 (%) でコード化された UTF-8 データを排他的に送信する必要があります。ASCII は UTF-8 でそのまま表されます。たとえば、スペースは %20 とコード化されます。

- **新しいクライアントコールバック API**　クライアント側でデータのロードとアンロードを実行する新しい機能のサポートのために、新しいクライアントコールバック API が追加されました。Embedded SQL については、「db_register_a_callback 関数」『SQL Anywhere サーバー プログラミング』の「DB_CALLBACK_VALIDATE_FILE_TRANSFER」を参照してください。ODBC については、「SQLSetConnectAttr 拡張接続属性」『SQL Anywhere サーバー プログラミング』の SA_REGISTER_VALIDATE_FILE_TRANSFER_CALLBACK を参照してください。

- **SQL_ATTR_CONNECTION_DEAD による切断された接続の検出の高速化**　ODBC の SQLGetConnectAttr 呼び出しを使用して SQL_ATTR_CONNECTION_DEAD 属性を取得すると、接続が切断されていた場合、切断後にサーバーに要求が送信されていなくても、値 SQL_CD_TRUE が取得されるようになりました。接続が切断したかどうかの確認は、サーバーに要求を送信しないで行われ、切断された接続は数秒以内に検出されます。接続が切断されるのには、アイドルタイムアウトなどの複数の理由があります。この変更の前は、SQL_ATTR_CONNECTION_DEAD で値 SQL_CD_TRUE が取得されたのは、接続が明示的に切断されたか、接続が切断された後に、SQLExecDirect を呼び出すなどして ODBC ドライバーからサーバーに要求が送信された場合だけでした。「接続属性の取得」『SQL Anywhere サーバー プログラミング』を参照してください。

- **JDBC ドライバーでの ResultSet.getBlob().getBinaryStream() のサポート**　iAnywhere JDBC ドライバーでは、現在 ResultSet.getBlob() メソッドがサポートされています。このメソッドは、JDBC 仕様ではオプションです。オプションの ResultSet.getBlob().getBinaryStream() メソッドのサポートが追加されました。「JDBC 3.0/4.0 API のサポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **iAnywhere JDBC ドライバーで URL ヘッダーとして jdbc:odbc に加えて jdbc:ianywhere を許可**　これまでは、URL ヘッダー jdbc:odbc を使用するアプリケーションでは、この URL への接続時に JDBC ドライバーマネージャーで iAnywhere JDBC ドライバーが使用されると見なすことができました。しかし、Java VM の最近のバージョンでは JDBC ドライバーとして Sun JDBC-ODBC ブリッジが登録されるようになり、Sun JDBC-ODBC ブリッジでは jdbc:odbc で始まる URL も許可されるので、アプリケーションに iAnywhere JDBC ドライバーではなく Sun JDBC-ODBC ブリッジが使用される可能性が高くなりました。JDBC ドライバーマネージャーで、Sun JDBC-ODBC ブリッジではなく iAnywhere JDBC ドライバーが確実に使用されるようにするには、アプリケーションで URL ヘッダー jdbc:ianywhere を使用してください。「JDBC クライアントアプリケーションからの接続」『SQL Anywhere サーバー プログラミング』を参照してください。

- **ODBC ドライバーマネージャーでの driver=iAnywhere Solutions 11 - Oracle の許可**　UNIX ODBC ドライバーマネージャーで driver=iAnywhere Solutions 11 - Oracle が許可されるようになりました。また、アプリケーションがスレッド化されている場合は、スレッド化された Oracle 用 iAnywhere ODBC ドライバーがロードされます。アプリケーションがスレッド化さ

れていない場合、ドライバーはロードされません。スレッド化されていない Oracle 用 iAnywhere ODBC ドライバーはサポートされていません。「iAnywhere Solutions 12 - Oracle ODBC ドライバー」『Mobile Link サーバー管理』を参照してください。

- **ODBC ドライバーマネージャーでの driver=UltraLite 11 の許可** UNIX ODBC ドライバーマネージャーでは、driver=SQL Anywhere 10 が許可され、SQL Anywhere ODBC ドライバーがロードされます (アプリケーションに応じて、スレッドバージョンまたは非スレッドバージョン)。UNIX ODBC ドライバーマネージャーでは driver=SQL Anywhere 11 と driver=UltraLite 11 も許可されるようになりました。Ultra Light ドライバーの場合、ドライバーマネージャーでは Ultra Light ODBC ドライバーのスレッドバージョンだけがロードされます。このドライバーはスレッドバージョンだけが存在します。

- **TDS 接続の強化** SQL Anywhere データベースサーバーで、Open Client のログインサーバー名がデフォルトデータベースの名前と一致しなくてもデフォルトデータベースへの TDS 接続が可能になりました。ただし、接続文字列でデータベースを起動していないこと (DBF=... がないこと)、またデータベースサーバーでデータベースが 1 つだけ実行されていること、という条件を満たしている必要があります。

- **管理ツールのランチャーの再配備の簡素化** データベースツール (Sybase Central、DBISQL、DBConsole、ML Monitor) のランチャー実行プログラムの再配備が簡単になりました。JAR ファイルの場所に関するレジストリのエントリやセットディレクトリ構造が不要になりました。各実行プログラムには、ツールのロード方法を詳細に示す (実行ファイルと同じ名前の) *.ini* ファイルが、実行ファイルと同じディレクトリに必要です。「管理ツールの配備」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL Anywhere の .NET データプロバイダーでの分散トランザクションのエンリストのサポート** .NET 2.0 フレームワークで、トランザクションアプリケーションを記述するためのクラスが含まれる新しいネームスペース System.Transactions が導入されました。クライアントアプリケーションで 1 つまたは複数の参加者が存在する分散トランザクションを作成し、そのトランザクションに参加できます。クライアントアプリケーションでは、TransactionScope クラスを使用して、暗黙的にトランザクションを作成できます。接続オブジェクトでは、TransactionScope によって作成されたアンビエントトランザクションの存在を検出し、自動的にエンリストできます。クライアントアプリケーションでは、CommittableTransaction を作成し、EnlistTransaction メソッドを呼び出してエンリストすることもできます。

  この機能は SQL Anywhere .NET 2.0 データプロバイダーでサポートされています。分散トランザクションには、大きなパフォーマンスのオーバーヘッドがあります。非分散トランザクションにデータベーストランザクションを使用することをおすすめします。「Transaction 処理」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL Anywhere .NET データプロバイダーでの名前付きパラメーターのサポート** SQL Anywhere プロバイダーで、SACommand の名前付きパラメーターがサポートされるようになりました。ユーザーがすべてのパラメーター名を指定すると、コマンドの実行時にプロバイダーによってパラメーター値がマッピングされます。名前付きパラメーターを使用するとき、パラメーターの順序が、ホスト変数の順序と一致している必要はありません。

```
SACommand cmd = new SACommand(
   "UPDATE MyTable SET name = :name WHERE id = :id", conn );

SAParameter p1 = new SAParameter(
```

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

```
    "id", SADbType.Integer );
p1.Direction = ParameterDirection.Input;
p1.Value = 1;
cmd.Parameters.Add( p1 );

SAParameter p2 = new SAParameter(
    "name", SADbType.Char, 40 );
p2.Direction = ParameterDirection.Input;
p2.Value = "asdasd";
cmd.Parameters.Add( p2 );

cmd.ExecuteNonQuery();
```

- **Web サービスの強化**　このリリースでは、次の Web サービスが強化されています。

  - **HTTP:POST タイプの Web クライアントサービスプロシージャーの拡張によるユーザー定義の本文の許可**　CREATE PROCEDURE 文と CREATE FUNCTION 文の TYPE 句が拡張され、MIME タイプを指定できるようになりました。「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』または「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **Web サービスクライアントプロシージャーの拡張による PUT、DELETE、HEAD HTTP の各メソッドのサポート**　Web サービスクライアントのプロシージャーと関数で PUT、DELETE、HEAD HTTP の各メソッドがサポートされるようになりました。CREATE PROCEDURE 文と CREATE FUNCTION 文の TYPE 句が拡張され、これらのメソッドがサポートされるようになりました。POST メソッドと同様に、PUT では TYPE 句内に Content-Type の拡張を必要とし、1 つの (非代入) パラメーターだけが許可されています。「CREATE SERVICE 文 [HTTP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「CREATE SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_http_php_page システムプロシージャーと sa_http_php_page_interpreted システムプロシージャー**　新しい Web サービスシステムプロシージャー sa_http_php_page と sa_http_php_page_interpreted は、PHP インタプリターを経由して渡された PHP スクリプトの結果を返します。「sa_http_php_page システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』と「sa_http_php_page_interpreted システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **HTTP_BODY システム関数**　Web サービスの新しい関数が追加されました。HTTP_BODY 関数は、HTTP 要求の本文をバイナリ形式で返します。「HTTP_BODY 関数 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **WSDLC での Web サービスクライアント SOAP プロシージャー生成のサポート**　C# と JAVA 用の QAnywhere クライアントサイド SOAP インターフェイスの生成に加えて、WSDLC では、SQL Anywhere 用の SQL SOAP (Web サービス) クライアントプロシージャーの生成がサポートされるようになりました。WSDLC では、WSDL1.1 準拠の URL またはファイルが読み込まれ、WSDL 内に記述された対応する各 SOAP 操作にマッピングする、適切なパラメーターと句を持つプロシージャー (または関数) が生成されます。生成された SQL 文は SQL ファイルに書き込まれます。「iAnywhere WSDL コンパイラーユーティリティ (wsdlc)」『SQL Anywhere サーバー プログラミング』を参照してください。

- **FORMAT 句を指定して定義した HTTP SOAP サービスに EXPLICIT OFF または ON を追加可能**　HTTP SOAP サービスを作成するとき、FORMAT 句のデフォルトは EXPLICIT ON です。このため、DISH サービスによって生成される WSDL では、結果セットで返されるカラムごとに明示的な名前とデータ型が指定されます。したがって、結果セットを表すクライアント側のオブジェクトとインターフェイスを SOAP クライアントツールキットで自動的に生成でき、カラム値へのネイティブアクセスが可能です。この機能が追加される前は、カラム値は、抽象 XML データ要素としてのみアクセスできました。EXPLICIT OFF を指定すると、この以前の動作を有効にできます。

  EXPLICIT 応答オブジェクトまたは汎用の SimpleDataset の定義方法の詳細については、「CREATE SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』と「チュートリアル：JAX-WS を使用した SOAP/DISH Web サービスへのアクセス」『SQL Anywhere サーバー プログラミング』を参照してください。

- **JSON Web サービスのサポート**　SQL Anywhere で、JSON 形式の応答を返す Web サービスがサポートされるようになりました。「CREATE SERVICE 文 [HTTP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **Web サービスのクライアントのロギング**　データベースサーバーで、Web サービスクライアントの接続を出力ファイルにロギングできるようになりました。-zoc サーバーオプションを指定するか、WebClientLogFile プロパティと WebClientLogging プロパティを sa_server_option システムプロシージャーとともに使用して、ログを制御し、Web サービスクライアントのログファイルの場所を指定できます。-sf サーバーオプションを使用してこの機能の使用を無効にすることもできます。次の項を参照してください。

  - 「-zoc dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』
  - 「-sf dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』
  - 「sa_server_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』

Windows Mobile の強化

次に、SQL Anywhere バージョン 11.0.0 での Windows Mobile に関する強化を示します。

- **-gss サーバーオプションのサポート**　Windows CE 4 (Pocket PC 2003) 以降で、-gss サーバーオプションを使用して、内部実行スレッドのデフォルトのスタックサイズを指定できます。「-gss dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **チェックサムがデフォルトで有効**　データベースが Windows Mobile で実行されているとき、データベースに対してチェックサムが有効になっているかどうかに関係なく、データベースサーバーによってチェックサムが自動的に有効にされます。この機能を使用するには、既存のデータベースをアップグレードするか、データベースを新規に作成する必要があります。「チェックサムを使用した破損の検出」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

UNIX/Linux の強化

次に、SQL Anywhere バージョン 11.0.0 での UNIX と Linux に関する強化を示します。

- **テンポラリファイルのパーミッションの制御** これまでのリリースでは、データベースサーバーとクライアントによって作成されたテンポラリファイルには、読み込み、書き込み、実行のグローバルパーミッションがありました。SATMP 環境変数を、目的のパーミッションを持つディレクトリに設定することで、テンポラリファイルのパーミッションを制御できます。「SATMP 環境変数」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SELinux のサポート** SELinux のポリシーを使用して、アプリケーションのシステムリソースへのアクセスを制御します。Red Hat Enterprise Linux 5 で SQL Anywhere を使用するときにデフォルトのポリシーを使用できますが、この方法で実行した場合、SQL Anywhere は保護されません。このリリースでは、Red Hat Enterprise Linux 5 で SQL Anywhere を保護するポリシーが含まれるようになりました。ポリシーを使用するには、コンパイルしてインストールする必要があります。ポリシーのソースコードが SQL Anywhere のインストールに含まれます。

  SQL Anywhere の SELinux ポリシーのコンパイルとインストールについては、*$SQLANY11/selinux/readme* を参照してください。

- **Linux の [アプリケーション] メニュー項目** SQL Anywhere 11 を Linux にインストールするときは、**[アプリケーション]** メニュー項目を作成することを選択できます。

Mac OS X の強化

- **Mac OS X での暗号化のサポート** Mac OS のデータベースサーバーとクライアントの両方で通信の RSA 暗号化がサポートされるようになりました。強力な暗号化の使用の詳細については、「トランスポートレイヤーセキュリティ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Mac OS X での HTTPS のサポート** Mac OS X のデータベースサーバーで HTTPS 通信がサポートされるようになりました。HTTPS の使用の詳細については、「-xs dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

その他

次に、SQL Anywhere バージョン 11.0.0 でのその他の強化を示します。

- **無効なビューの変更** これまでは、INVALID ステータスの通常のビューは変更できず、変更が必要な場合はビューを削除してから再作成する必要がありました。このリリースでは、無効なビューの定義を変更して、有効になるようにすることができます。

- **LENGTH 関数への同意語の追加** LENGHT 関数の実行時、LENGTH の同義語として LEN を使用できるようになりました。「LENGTH 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ビッグエンディアンとリトルエンディアンの UTF-16 エンコードのサポート** SQL Anywhere では、プラットフォームのエンディアンに関係なく、すべてのプラットフォームでビッグエン

ディアンとリトルエンディアンの両方の UTF-16 エンコードがサポートされるようになりました。LOAD TABLE 文と UNLOAD 文、また CSCONVERT 関数で UTF-16 エンコードを使用できます。ただし、接続またはデータベースのエンコードとして UTF-16 エンコードを使用することはできません。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「UNLOAD 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

この機能を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」 366 ページを参照してください。

- **インデックスのパフォーマンスの向上** 特にキャッシュが満杯になったときのインデックスのパフォーマンスが向上しました。インデックスのパフォーマンスを向上するには、インデックスを再構築する必要があります。最も簡単な方法は、データベースを再構築することです。再構築したら、データベースファイルが大幅に小さくなっている可能性があります。これは正常な現象であり、問題ではありません。

- **圧縮されたカラムでの INLINE と PREFIX の設定の尊重** これまでは、カラムに対して指定された INLINE と PREFIX の設定は、カラムが圧縮されていた場合は無視され、0 として処理されていました。このリリースでは、カラムが圧縮されていても、カラムに対するこれらの設定が尊重されるようになりました。「BLOB の考慮事項」『SQL Anywhere サーバー データベース管理』と「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **バッチでのホスト変数の使用許可** いくつかの制約付きで、バッチ内でのホスト変数の参照が許可されるようになりました。「バッチ」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **IN リストアルゴリズムの強化** これまでは、IN リスト内のすべての要素が定数値であるか、最適化時に評価すると定数値になる場合にのみ、オプティマイザーで IN リストアルゴリズムが使用されました。このリリースでは、IN リスト述部に、開くときだけに評価される値 (CURRENT DATE、CURRENT TIMESTAMP、非決定的システム関数、ユーザー定義関数) や、クエリブロック 1 回の実行中に定数である値 (外部参照) を含めることができるようになりました。

- **単純な DML 文でのプランのキャッシュ** プランのキャッシュが拡張され、クエリを省略できる SELECT 文 (単純な文) が含まれるようになりました。「プランのキャッシュ」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **新しいデータベースのサイズ縮小** システムテーブルの次のカラムが圧縮され、新しい (空の) データベースのサイズが約 200 KB 縮小されました。これは、Windows Mobile で使用するデータベースを作成するときに役立ちます。

  - ISYSEVENT.action
  - ISYSJARCOMPONENT.contents
  - ISYSPROCEDURE.proc_defn
  - ISYSSOURCE.source
  - ISYSTEXTCONFIG.char_stoplist
  - ISYSTEXTCONFIG.nchar_stoplist
  - ISYSTRIGGER.trigger_defn
  - ISYSVIEW.view_def

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **デフォルトパケットサイズと最小パケットサイズの拡大**　Windows Mobile を除くすべてのオペレーティングシステムで、デフォルトのパケットサイズが 7300 バイトに拡大されました。Windows Mobile では、デフォルトは 1460 バイトのままです。最小パケットサイズは 500 バイトに拡大されました。「CommBufferSize (CBSIZE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』と「-p dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **リモートデータアクセスのための新しい ODBC クラスのサポート**　次の ODBC クラスのサポートが追加されました。

  - msaccessodbc
  - mysqlodbc
  - ulodbc
  - adsodbc

  詳細については、「ODBC ベースのサーバークラス」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  注意
  これまで SQL Anywhere for MS Access 移行ユーティリティ (upsize ツール) を使用して Microsoft Access データベースを SQL Anywhere に移行していた場合、今後は msaccessodbc クラスを使用できます。

- **データベースサーバーのメッセージの強化**　データベースサーバーからのメッセージにカテゴリと重要度が割り当てられるようになりました。この情報には sa_server_messages システムプロシージャーを使用してアクセスできます。また、MessageCategoryLimit プロパティを使用して、維持するメッセージ数を設定できます。「sa_server_messages システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **a_validate_type 列挙体の新しい VALIDATE_COMPLETE パラメーター**　a_validate_type 列挙体に、データベースに対して可能な検証をすべて実行できる新しいパラメーター VALIDATE_COMPLETE が追加されました。a_validate_db 構造体 [データベースツール]『SQL Anywhere サーバー プログラミング』を参照してください。

- **外部アンロードの強化**　データベースの外部アンロードを行うとき、生成される *reload.sql* の先頭に、コメント付き CREATE DATABASE 文が含まれるようになりました。この文を使用して、アンロードしたデータベースと同じデータベースを作成できます。

  アンロードしたデータベースがバージョン 9 以前の SQL Anywhere で作成され、カスタム照合があった場合は、COLLATION 句は次のようになります。

  ```
  COLLATION collation-label DEFINITION collation-definition
  ```

  ここで *collation-definition* はカスタム照合を指定する文字列です。

  アンロードしたデータベースが強力な暗号化を使用して作成されていた場合は、CREATE DATABASE 文の KEY 句の値が疑問符 3 つ (???) として表示されます。

  詳細については、「内部アンロードと外部アンロード、内部再ロードと外部再ロード」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい SQL Anywhere Extension Agent OID**　このリリースには、次の OID が追加されています。

  - saAgent.saRestart
  - saAgent.saInifile

  詳細については、「SQL Anywhere MIB リファレンス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Deadlock システムイベント**　Deadlock システムイベントは、デッドロックが発生したときに起動します。イベントハンドラーでは、sa_report_deadlocks プロシージャーを使用して、デッドロックが発生するに至った状況に関する情報を取得できます。Deadlock システムイベントを使用するには、既存のデータベースをアップグレードする必要があります。「システムイベント」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースの上限の増加**　SQL Anywhere データベースの制限値がいくつか増加しました。「SQL Anywhere のサイズと数の制限」『SQL Anywhere サーバー データベース管理』を参照してください。

- **実行プランの変更**　オプティマイザーによって生成される長いプランに、プラン全体に関連する次のエントリが表示されるようになりました。

  - **見積り済み最良プラン**　オプティマイザーが検出した最良アクセスプランの数。
  - **見積り済みプラン**　オプティマイザーが算出したアクセスプランの数。
  - **最適化時間**　クエリの最適化に要した時間。

  「実行プランのコンポーネント」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  グラフィカルなプランには次のエントリが表示されるようになりました。

  - **見積り済み最良プラン**　このエントリはルートノードの **[オプティマイザー統計]** セクションにあり、オプティマイザーが検出した最良アクセスプランの数を示します。
  - **見積り済みプラン**　このエントリはルートノードの **[オプティマイザー統計]** セクションにあり、オプティマイザーが算出したアクセスプランの数を示します。
  - **最適化時間**　このエントリはルートノードの **[オプティマイザー統計]** セクションにあり、クエリの最適化に要した時間を示します。
  - **FirstRowRunTime**　このエントリは **[ノード統計]** セクションにあり、最初のローのフェッチ時間を示します。
  - **算出されたジョイン**　このエントリはジョイン演算子の **[高度な詳細]** セクションにあり、ジョイン演算子の右側のサブツリーの最適化処理中にオプティマイザーが算出したジョイン演算子のリストを示します。
  - **事前フィルター述部**　このエントリは **[詳細]** ウィンドウ枠の新しいスキャンノードのセクションにあり、スキャンの開始前に評価される述部のリストを示します。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **スキャン述部**　このエントリは **[詳細]** ウィンドウ枠のスキャンノードのセクションにあり、ローからフェッチされるカラムとして評価される述部のリストを示します。スキャン述部でローが拒否された場合、それ以上カラムは読み込まれません。スキャン述部は T.x <= 3 や T.x IS NULL などのシンプルな 1 つのカラムの述部です。
- **ポストスキャン述部**　このエントリは **[詳細]** ウィンドウ枠の新しいスキャンノードのセクションにあり、ローがテーブルページから読み込まれた直後に評価される述部のリストを示します。ポストスキャン述部では、複数のカラムを参照でき、また関数や算術演算を使用できます。
- **未確定述部**　このエントリは **[詳細]** ウィンドウ枠の新しいスキャンノードのセクションにあり、ローのセットがメモリにフェッチされた後に評価される述部のリストを示します。未確定述部は、通常はサブクエリやユーザー定義関数などの複雑な操作が含まれ、スキャン述部やポストスキャン述部として評価できません。
- **算出されたインデックス**　**[高度な詳細]** ウィンドウ枠にあるこのエントリは、このスキャン演算子によって参照されているテーブルの最適化処理中にオプティマイザーが算出したインデックススキャンまたはテーブルスキャンのリストを示します。リスト内の各項目のフォーマットは、**[詳細]** ウィンドウ枠内のアクセスプランで使用されているスキャン演算子の詳細と似ています。
- **Primary Key Table**　このエントリはインデックススキャン演算子の **[インデックス]** セクションにあり、プライマリキーテーブル名を示します。
- **Primary Key Table Estimated Rows**　このエントリはインデックススキャン演算子の **[インデックス]** セクションにあり、プライマリキーテーブル内のロー数を示します。
- **Primary Key Column**　このエントリはインデックススキャン演算子の **[インデックス]** セクションにあり、プライマリキーカラムの名前を示します。
- **連続変換**　このエントリはインデックススキャン演算子の **[インデックス]** セクションにあり、各物理インデックスについて、インデックスのクラスター化の程度を示す統計を示します。
- **ランダム変換**　このエントリはインデックススキャン演算子の **[インデックス]** セクションにあり、各物理インデックスについて、インデックスのクラスター化の程度を示す統計を示します。
- **キー値**　このエントリはインデックススキャン演算子の **[インデックス]** セクションにあり、インデックス内のユニークなエントリの数を示します。

SQL Anywhere の動作の変更

次に、SQL Anywhere バージョン 11.0.0 での動作の変更を、カテゴリごとに示します。

- **カタログの変更**　次の表は、バージョン 11.0.0 でのカタログの変更点を示しています。

  これらの変更点を使用するには、データベースをアップグレードする必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

テーブル名とビュー名	変更点の説明
ISYSTAB/SYSTAB	○新しいカラム dbspace_id が追加されました。既存の file_id カラムは最終的にこのカラムよって置き換えられます。 ○file_id カラムは推奨されなくなりました。今後は dbspace_id を使用してください。グローバルテンポラリテーブルでは、SYSTAB.file_id はシステム DB 領域ではなく、テンポラリ DB 領域を指すようになりました。 ○テーブルを変更したトランザクションのシーケンス番号を格納する last_modified_tsn という新しいカラムが追加されました。
ISYSIDX/SYSIDX	○新しいカラム dbspace_id が追加されました。既存の file_id カラムは最終的にこのカラムよって置き換えられます。 ○file_id カラムは推奨されなくなりました。今後は dbspace_id を使用してください。
ISYSFILE	このシステムテーブルは推奨されなくなりました。lob_map を除くすべてのカラムは新しい ISYSDBSPACE システムテーブルにあります。lob_map カラムは新しい ISYSDBFILE システムテーブルにあります。
ISYSDBFILE/SYSDBFILE	DB 領域に関する情報を格納するための新しいテーブルです。
ISYSDBSPACE/SYSDBSPACE	DB 領域に関する情報を格納するための新しいテーブルです。
SYSDBSPACEPERM/ISYSDBSPACEPERM	DB 領域のパーミッションを格納するための新しいテーブルです。
ISYSOBJECT/SYSOBJECT	file_id カラムの名前が dbspace_id に変更されました。また、object_type カラムに 17 (テキスト設定) と 18 (DB 領域) の 2 つの新しい値を格納できるようになりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

テーブル名とビュー名	変更点の説明
SYSINDEXES	外部キーとプライマリキーのインデックスを他のインデックスと区別するために、indextype フィールドでそれぞれ「外部キー」および「プライマリキー」と識別されるようになりました。
ISYSCAPABILITYNAME	このテーブルはカタログに含まれなくなりました。対応する SYSCAPABILITYNAME システムビューはまだ使用できますが、サーバープロパティを使用して生成されます。
ISYSEVENTTYPE	このテーブルはカタログに含まれなくなりました。対応する SYSEVENTTYPE システムビューはまだ使用できますが、サーバープロパティを使用して生成されます。
ISYSVIEW	マテリアライズドビューをリフレッシュしたトランザクションのシーケンス番号を格納する mv_last_refreshed_tsn という新しいカラムが追加されました。
ISYSLOGINMAP/SYSLOGINMAP	ログインポリシーに関する情報を格納するための新しいテーブルです。
ISYSLOGINPOLICY/SYSLOGINPOLICY	ログインポリシーに関する情報を格納するための新しいテーブルです。
ISYSLOGINPOLICYOPTION/ SYSLOGINPOLICYOPTION	ログインポリシーに関する情報を格納するための新しいテーブルです。
ISYSTEXTCONFIG/SYSTEXTCONFIG	テキスト設定オブジェクトに関する情報を格納するための新しいテーブルです。
ISYSTEXTIDX/SYSTEXTIDX	テキストインデックスに関する情報を格納するための新しいテーブルです。
ISYSTEXTIDXTAB/SYSTEXTIDXTAB	テキストインデックスに関する情報を格納するための新しいテーブルです。

- **PHP 関数名の変更**　PHP 関数名のプレフィクスは、すべて sqlanywhere_ から sasql_ に変更されました。sqlanywhere_ プレフィクスも関数を呼び出すときにまだ使用できますが、廃止される予定です。新しいプレフィクスを使用するようにアプリケーションを変更してください。

- **INSERT...ON EXISTING UPDATE 文でのトリガーの起動**　これまでは、INSERT...ON EXISTING UPDATE 文を実行したとき、データが更新された場合はトリガーは起動しませんでした。このリリースでは、更新に対して文レベルの AFTER トリガーが起動します。

- **REFRESH MATERIALIZED VIEW 文**　リフレッシュの独立性レベルとして STATEMENT SNAPSHOT と READONLY STATEMENT SNAPSHOT を指定できなくなりました。独立性レベルに SNAPSHOT を指定するとこれらのオプションと同じ効果が得られます。「REFRESH MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **REORGANIZE TABLE 文**　同じテーブルに対して複数の REORGANIZE TABLE 文を同時に実行しようとするとエラーが発生します。

- **sa_validate システムプロシージャー**　sa_validate の check_type、エクスプレス、チェックサムの引数が廃止され、指定しても効果はありません。このリリースでは、デフォルトでチェックサム検証が実行されます。また、引数を指定しないで sa_validate システムプロシージャーを呼び出すと、データベースサーバーでは、すべてのテーブル、マテリアライズドビュー、インデックスの検証に加えて、チェックサムを含むデータベース自体が検証されます。「sa_validate システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **-gss サーバーオプション**　Windows XP 以降で -gss サーバーオプションがサポートされるようになりました。これまでのリリースでは、このオプションは Windows オペレーティングシステムではサポートされていませんでした。「-gss dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-gx サーバーオプションのサポート終了**　このリリースでは、-gx サーバーオプションがサポートされなくなりました。SQL Anywhere データベースサーバーの起動時に -gx オプションを指定すると、エラーが発生します。

- **LazyClose 接続パラメーターのデフォルト設定が AUTO に変更**　これまでのバージョンでは、アプリケーションでカーソルを閉じるとき、LazyClose 接続パラメーターが NO に設定されていなかった場合はデータベースサーバーへの往復が必要でした。このリリースでは、カーソルを閉じる要求はデフォルトで複数のカーソル分がキューイングされるようになったので、往復がなくなり、パフォーマンスが向上します。LazyClose 接続パラメーターには、YES、NO、AUTO (デフォルト) の 3 つの値を指定できるようになりました。これまでのリリースでは、YES がデフォルト設定でした。「LazyClose (LCLOSE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Embedded SQL のインポートライブラリの変更**　DBLIB のインポートライブラリの Watcom と Borland のバージョンが含まれなくなりました。これらはそれぞれ *dblibtw.lib* と *dblibtb.lib* です。これらのインポートライブラリの代わりにインポート定義ファイル (*%SQLANY11%¥SDK¥Lib¥Def¥dblib.def* ファイル) が用意されています。

- **データベースツールのインポートライブラリの変更**　データベースツールのインポートライブラリの Watcom と Borland のバージョンが含まれなくなりました。これらはそれぞれ *dbtlstw.lib* と *dbtlstb.lib* です。これらのインポートライブラリの代わりにインポート定義ファイル (*%SQLANY11%¥SDK¥Lib¥Def¥dbtool.def* ファイル) が用意されています。

- **ローを受信しなかったときの DBLIB インジケーターの動作の定義**　フェッチ時または実行時にデータベースサーバー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

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

末尾に到達した場合)、インジケーターの値が変更されなくなりました。「インジケーター変数」『SQL Anywhere サーバー プログラミング』を参照してください。

- **ODBC SQLGetConnectAttr**　ODBC の SQLGetConnectAttr 呼び出しを使用して SQL_ATTR_CONNECTION_DEAD 属性を取得すると、接続が切断されていた場合、切断後にサーバーに要求が送信されていなくても、値 SQL_CD_TRUE が取得されるようになりました。接続が切断したかどうかの確認は、サーバーに要求を送信しないで行われ、切断された接続は数秒以内に検出されます。接続が切断されるのには、アイドルタイムアウトなどの複数の理由があります。

  これまでのリリースでは、SQL_ATTR_CONNECTION_DEAD で値 SQL_CD_TRUE が取得されたのは、接続が明示的に切断されたか、接続が切断された後に、SQLExecDirect を呼び出すなどして ODBC ドライバーからサーバーに要求が送信された場合だけでした。

- **utility_db という名前のデータベースを作成または起動できない**　utility_db という名前は SQL Anywhere サーバーユーティリティデータベースのために予約されています。*utility_db.db* という名前のデータベースを新規に作成するか、この名前の既存のデータベースを起動しようとすると、エラーが返されます。utility_db という名前の既存のデータベースがある場合は、別の名前で起動できます。「ユーティリティデータベース」『SQL Anywhere サーバー データベース管理』を参照してください。

- **計算カラムの依存性**　これまでは、更新操作または挿入操作をエラーなしで続行するには、アプリケーションでトリガーを使用して、NOT NULL と宣言されているカラムに NULL 以外の値を割り当てることができました。これは、このカラムに依存する計算カラムに影響し、意図した計算が計算値に反映されない場合がありました。このリリースでは、計算カラムが依存する NOT NULL カラムに NULL 値を設定しようとするとエラーメッセージが表示され、トリガーは起動しません。「計算カラムへの挿入と更新」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ピリオドを含む DB 領域名のエラー**　これまでのリリースでは、引用符で囲んでいない DB 領域名にピリオドが含まれていた場合、DB 領域名のピリオドの前の部分は、サーバーで通知されることなく無視されていました。このリリースでは、このような名前があった場合、データベースサーバーでエラーが生成されます。

- **SQL Anywhere Web サーバーでの SSL バージョン 2.0 のサポート終了**　SQL Anywhere Web サーバーを使用するとき、SSL バージョン 3.0 と TLS バージョン 1.0 の接続だけがサポートされます。SSL バージョン 2.0 の接続はサポートされません。

- **CREATE SERVICE オプションの DATATYPE のデフォルト値の変更**　DATATYPE 句のデフォルト値が OFF から ON に変更されました。以前の動作が必要な場合は、CREATE SERVICE の定義に DATATYPE OFF を明示的に含めてください。「CREATE SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **一部の保護された機能の名前変更**　このリリースでは、次の保護された機能の名前が変更されています。

以前の名前	新しい名前
xp_read_file	read_file
xp_write_file	write_file
unload_table	write_file
load_table	read_file

詳細については、「-sf dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **チェックサムの動作の変更** バージョン 11 で作成したデータベースまたはバージョン 11 にアップグレードしたデータベースを、ネットワークドライブやリムーバブルドライブなどのメディアで実行した場合、データベースサーバーによってチェックサムが自動的に有効にされます。データベースがこのようなデバイスにあるかぎり、チェックサムは有効なままです。「チェックサムを使用した破損の検出」『SQL Anywhere サーバー データベース管理』を参照してください。

- **HTTP 接続でデータベースが自動停止されない** これまでのリリースでは、データベースを自動的に停止するように設定すると、HTTP 接続が切断され、データベースへの接続が他になかった場合にデータベースが停止していました。このリリースでは、最後の Command Sequence 接続または TDS 接続が切断したときにのみデータベースが自動的に停止します。

  データベースへの唯一の接続が HTTP 接続で、自動的に停止するようにデータベースが設定されている場合、HTTP 接続が切断したときにデータベースは自動的に停止しません。また、自動的に停止するように設定されているデータベースに HTTP 接続と Command Sequence 接続または TDS 接続がある場合は、最後の Command Sequence 接続または TDS 接続が切断したときにデータベースが停止します。このときに HTTP 接続がまだあった場合は切断されます。「-ga dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』と「AutoStop (ASTOP) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースミラーリングの動作の変更** これまでのリリースでは、プライマリサーバーまたはミラーサーバーの -xp オプションで指定した接続パラメーターが無効であった場合、データベースサーバーで接続が繰り返し試行されましたが、接続できませんでした。このリリースでは、-xp オプションで指定したパラメーターが無効であり、サーバーで複数のデータベースが実行されている場合、ミラーデータベースの起動に失敗し、再接続は試行されません。ミラーデータベースが、データベースサーバーで実行されている唯一のデータベースである場合は、データベースサーバーが起動しません。

- **マテリアライズドビューのデフォルトのリフレッシュ動作** これまでは、マテリアライズドビューのデフォルトのリフレッシュ動作は WITH EXCLUSIVE MODE でした。このリリースでは、デフォルトのリフレッシュ動作は、マテリアライズドビューが IMMEDIATE REFRESH と定義されているかどうか、またデータベースでスナップショットアイソレーションレベルが有効になっているかどうかによって異なります。「REFRESH MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **post_login_procedure データベースオプションの動作の変更**　post_login_procedure データベースオプションのデフォルト設定が sa_post_login_procedure システムプロシージャーになりました。「post_login_procedure オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **non_keywords データベースオプション**　これまでのリリースでは、個々のキーワードの指定に加え、次に示すキーワードリストの中から特別値を使用して、特定のリリース以降のすべてのキーワードをオフにできました。

  keywords_4_0_d, keywords_4_0_c, keywords_4_0_b, keywords_4_0_a, keywords_4_0, keywords_5_0_01, keywords_5_0

  これらの特別値はサポートされなくなりました。個々のキーワードをオフにすることはできます。「non_keywords オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **cooperative_commit_timeout データベースオプション**　コミット動作が自動的にチューニングされるため、現在ではこのオプション設定は無視されるようになりました。

- **cooperative_commits データベースオプション**　コミット動作が自動的にチューニングされるため、現在ではこのオプション設定は無視されるようになりました。

- **リモートデータアクセスでの quoted_identifier データベースオプション設定の尊重**　リモートデータアクセスを使用している場合、quoted_identifier オプションのローカル設定を使用して、Adaptive Server Enterprise と Microsoft SQL Server に対する引用符付き識別子の使用を制御できるようになりました。たとえば、quoted_identifier オプションをローカルで Off に設定すると、Adaptive Server Enterprise に対して引用符付き識別子がオフになります。次の項を参照してください。

  ○「サーバークラス aseodbc」『SQL Anywhere サーバー SQL の使用法』
  ○「サーバークラス mssodbc」『SQL Anywhere サーバー SQL の使用法』

- **precision と scale の各データベースオプションのスコープの変更**　これまでのリリースでは、個々のユーザーに precision と scale の各データベースオプションを設定したり、設定のスコープがテンポラリであることを指定したりできました。ただし、これらの設定はデータベースのリカバリ性に影響する可能性があります。テーブルやドメインを作成または変更する DDL 文の実行時に、テンポラリ設定またはユーザーレベルの設定が、対応する PUBLIC 設定と異なる場合、データベースの再構築時に問題が生じる可能性があります。このリリースでは、precision と scale の各データベースオプションの動作は次のようになっています。

データベースサーバーのバージョン	バージョン 10 以前のデータベース	バージョン 11 のデータベース	バージョン 11 にアップグレードしたデータベース	バージョン 10 以前のデータベースのアンロード
11	PUBLIC 設定は可能 ユーザー設定は可能 テンポラリ設定は**不可**	PUBLIC 設定は可能 ユーザー設定は**不可** テンポラリ設定は**不可**	PUBLIC 設定は可能 ユーザー設定は**不可** テンポラリ設定は**不可**	PUBLIC 設定はアンロードされる ユーザー設定はアンロード時に破棄される
10 以前	PUBLIC 設定は可能 ユーザー設定は可能 テンポラリ設定は可能	なし	なし	PUBLIC 設定はアンロードされる ユーザー設定はアンロードされる

バージョン 10 以前のデータベースサーバーでは、引き続き scale と precision の各オプションをテンポラリに、また個々のユーザーに設定できます。

警告
precision と scale の各データベースオプションのユーザーレベル設定またはテンポラリ設定を頼りにしないことをおすすめします。データベースの再構築時に問題が生じる可能性が含まれており、データベースサーバーが予測できない動作をする可能性もあります。

次の項を参照してください。

- 「precision オプション」『SQL Anywhere サーバー データベース管理』
- 「scale オプション」『SQL Anywhere サーバー データベース管理』

● **OPTION 句の動作の変更**　INSERT、UPDATE、DELETE、SELECT、UNION、EXCEPT、INTERSECT の各文の OPTION 句でサポートされていないデータベースオプションを指定すると、エラーが返されるようになりました。次の項を参照してください。

- 「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「UPDATE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「UNION 文」『SQL Anywhere サーバー SQL リファレンス』
- 「EXCEPT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「INTERSECT 文」『SQL Anywhere サーバー SQL リファレンス』

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **読み込み専用データベースのロールバックログの動作の変更**　これまでのリリースでは、トランザクション指向のテンポラリオブジェクトがある読み込み専用データベースの操作はトランザクション指向として処理されず、ロールバックログ情報が維持されませんでした。このリリースでは、読み込み専用データベース内のトランザクション指向テンポラリオブジェクトに完全にトランザクション指向のセマンティックがあります。これらのオブジェクトは、コミット、ロールバック、セーブポイントまでのロールバックの対象となります。

- **Itanium 64 ビットのサポート対象プラットフォームの変更**　これまでのバージョンでは、Itanium II チップ搭載の Windows Server 2003 を対象としたソフトウェアの 64 ビット通常版と、64 ビットの Linux と HP-UX の各オペレーティングシステムを対象としたクライアント／サーバー限定版がありました。

  このリリースでは、64 ビット HP-UX を対象としたクライアント／サーバー限定版だけを使用できます。

- **アンロードユーティリティ (dbunload) の動作の変更**　これまでのリリースでは、dbunload の -ea、-ek、-ep の各オプションを、-an オプションまたは -ar オプションと同時に指定して、新しいデータベースの暗号化を制御する必要がありました。このリリースでは、データベースの全体またはその一部をアンロードした後に再ロードしなかった場合、-ea、-ek、-ep の各オプションによって、作成されるテーブルデータファイルの暗号化が制御されます。これらのファイルを使用して Interactive SQL からデータベースを再ロードするとき、READ 文のパラメーターとして暗号化キーを指定する必要があります。「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

  また、これまでのリリースでは、データベースの抽出に使用する dbunload のバージョンが、データベースを実行しているデータベースサーバーと同じバージョンである必要はありませんでした。このリリースでは、バージョン 10.0.0 以降のデータベースで dbunload を使用する場合、使用する dbunload のバージョンが、データベースへのアクセスに使用するデータベースサーバーのバージョンと一致する必要があります。dbunload のバージョンがデータベースサーバーのバージョンより古いまたは新しい場合は、エラーがレポートされます。

- **抽出ユーティリティ (dbxtract) の動作の変更**　これまでのリリースでは、データベースの抽出に使用する dbxtract のバージョンが、データベースを実行しているデータベースサーバーと同じバージョンである必要はありませんでした。このリリースでは、バージョン 10.0.0 以降のデータベースで dbxtract を使用する場合、使用する dbxtract のバージョンが、データベースへのアクセスに使用するデータベースサーバーのバージョンと一致する必要があります。dbxtract のバージョンがデータベースサーバーのバージョンより古いまたは新しい場合は、エラーがレポートされます。

- **ロックの動作の変更**　これまでのリリースでは、独立性レベル 0 で実行されている UPDATE 文または DELETE 文が、文の処理対象ではないローのローロックをブロックする場合がありました。このリリースでは、UPDATE 文または DELETE 文が、文の処理対象ではないローに意図的ロックまたは排他ロックをかける可能性は低くなっています。アプリケーションを開発する際は、UPDATE 文と DELETE 文に独立性レベル 0 または 1 を使用する場合は注意し、動作がアプリケーションで許容可能であることを確認する必要があります。「更新時のロック」『SQL Anywhere サーバー SQL の使用法』と「削除時のロック」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **プロパティ名の変更**　このリリースでは、次のプロパティの名前が変更されています。

以前の名前	新しい名前
CacheHitsEng	CacheHits
CacheReadEng	CacheRead
DiskReadEng	DiskRead
ReadHint	DiskReadHint
ReadHintScatter	DiskReadHintPages
ReadHintScatterLimit	DiskReadHintScatterLimit

詳細については、「接続、データベース、データベースサーバーのプロパティ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **言語選択ユーティリティ (dblang)** これまでのリリースでは、このユーティリティは、インストール時に多言語リソース開発キット (IRDK) を選択した場合にのみインストールされていました。このリリースでは、すべての多言語リソースと言語選択ユーティリティ (dblang) が常にインストールされます。

- **テンポラリテーブルとインデックスのデフォルトの DB 領域** テンポラリテーブルは、TEMPORARY DB 領域だけに作成できます。CREATE TABLE 文の IN 句で SYSTEM DB 領域を指定すると、IN 句は無視され、テンポラリテーブルはテンポラリ DB 領域に作成されます。CREATE TABLE 文の IN 句でユーザー定義の DB 領域を指定すると、エラーが返されます。また、テンポラリオブジェクト作成時は default_dbspace オプションが無視されます。

- **テンポラリテーブルへのデータのロード** テンポラリテーブルにデータをロードするとき、ON COMMIT DELETE と指定されたローカルテンポラリテーブルをロードできなくなりました。これまでのリリースでは、ON COMMIT DELETE ROWS と指定して定義されたローカルテンポラリテーブルにデータをロードできました。

  このリリースでは、LOAD TABLE 文を実行すると、オートコミットが自動的に実行されます。これまでのリリースではオートコミットは実行されない場合もありました。

- **データベースサーバーオプション** サーバーオプション -uc と -ui が Mac OS X でサポートされるようになりました。これまでは、Linux だけでサポートされていました。Linux では、-ui サーバーオプションを使用すると、**[サーバー起動オプション]** ウィンドウが開き、データベースサーバーメッセージウィンドウが表示され、X-Window Server が起動するかどうかにかかわらず、データベースサーバーが起動します。Mac OS X では、-ui を使用すると、データベースサーバーメッセージが新しいウィンドウに表示され、使用可能な表示がない場合はデータベースサーバーがシェルモードで起動します。-uc サーバーオプションでは、データベースサーバーがシェルモードで起動します。「-uc dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』と「-um dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Unicode 呼び出しをサポートしない ODBC ドライバーでリモートデータアクセスが機能しない** Unicode 呼び出しをサポートしない ODBC ドライバーでリモートデータアクセスが機能

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

しなくなりました。このため、Unicode 以外の ODBC ドライバーの場合、リモートデータアクセスでは ODBC ドライバーからのデータに対して文字セットの変換が行われません。

- **SYSFILE システムビュー**　SYSFILE 互換ビューにテンポラリファイル用のローが含まれるようになりました。

SQL Anywhere の廃止予定機能とサポート終了機能

- **LOAD TABLE、UNLOAD TABLE、INPUT、OUTPUT の各文の FORMAT ASCII 句は廃止予定**　LOAD TABLE、UNLOAD TABLE、INPUT、OUTPUT の各文の FORMAT ASCII 句が推奨されなくなり、FORMAT TEXT に置き換えられました。dbunload などのユーティリティでは、FORMAT ASCII ではなく FORMAT TEXT を含む再ロードスクリプトが生成されるようになりました。

  OUTPUT 文では、FORMAT TEXT 句によって、旧バージョンでの FORMAT ASCII と同じファイル形式でデータが書き込まれます。これまで FORMAT TEXT によって作成されていた出力は使用できなくなりました。

- **データベースプロパティ**　次のデータベースプロパティはサポートされなくなりました。

  - MapPages
  - PreserveSource
  - UniqueIdentifier

- **サーバープロパティ**　このリリースでは、次のサーバープロパティが推奨されなくなっています。

  - MaxMessage
  - Message
  - MessageTime
  - MessageText
  - MessageWindowSize

- **SPX プロトコルのサポート終了**　このリリースでは、SPX プロトコルがサポートされなくなりました。その結果、次のプロトコルオプションがサポートされなくなりました。

  - ExtendedName プロトコルオプション [ENAME]
  - RegisterBindery プロトコルオプション [REGBIN]
  - SearchBindery プロトコルオプション [BINSEARCH]

  SQL Anywhere .NET データプロバイダーの次の機能がサポートされなくなりました。

  - SACommLinksOptionsBuilder クラス：SpxOptionsBuilder プロパティ
  - SACommLinksOptionsBuilder クラス：SpxOptionsString プロパティ
  - SASpxOptionsBuilder クラス

- **dbinit -e オプションのサポート終了**　データベースの作成時に単純暗号化を指定するための dbinit -e オプションはサポートされなくなりました。単純暗号化を指定するには -ea simple

オプションを使用してください。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サポート終了したデータベースオプション**　このリリースでは、次のデータベースオプションと、対応するデータベースプロパティがサポートされなくなりました。

オプション	このリリースでの動作
ansi_integer_overflow	オーバーフローが発生すると必ず SQLSTATE = 22003 - overflow エラーが発生するようになりました。 マテリアライズドビューがある古いデータベースにアンロードまたは接続するときは、このオプションの設定は無視されます。
ansi_substring	SUBSTRING 関数の動作が ANSI/ISO SQL/2003 の動作と一致するようになりました。「SUBSTRING 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
automatic_timestamp	TIMESTAMP データ型の新規カラムに明示的なデフォルト値が定義されていない場合、デフォルト値として Transact-SQL の timestamp が適用されなくなりました。
divide_by_zero_error	ゼロによる除算があると、SQLSTATE 22012 のエラーが発生します。 マテリアライズドビューがある古いデータベースにアンロードまたは接続するときは、このオプションの設定は無視されます。
float_as_double	SQL Anywhere で、精度が指定されていない場合、FLOAT キーワードが Adaptive Server Enterprise の FLOAT キーワードと同じように動作しません。SQL Anywhere では、FLOAT 値は DOUBLE 値と同じように処理されません。 Open Client 接続と jConnect 接続では、この動作は、これまでのリリースのデフォルトの動作と異なります。 マテリアライズドビューがある古いデータベースにアンロードまたは接続するときは、このオプションの設定は無視されます。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

オプション	このリリースでの動作
optimistic_wait_for_commit	このオプションはサポートされなくなりました。
query_plan_on_open	カーソルに対して OPEN を実行したときにプランが返されなくなりました。EXPLAIN 文か PLAN 機能を使用すると詳細な説明が得られます。「EXPLAIN 文 [ESQL]」『SQL Anywhere サーバー SQL リファレンス』と「PLAN 関数 [その他]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
ri_trigger_time	UPDATE または DELETE の後に参照整合性アクションが実行されるようになりました。
truncate_with_auto_commit	TRUNCATE TABLE 文の実行前と実行後の両方に COMMIT が実行されるようになりました。
tsql_hex_constant	16 進定数がバイナリ文字定数として処理されるようになりました。
uuid_has_hyphens	UUID 文字列に 4 つのハイフンが含まれるようになりました。 マテリアライズドビューがある古いデータベースにアンロードまたは接続するときは、このオプションの設定は無視されます。
percent_as_comment	これまでのリリースでは、percent_as_comment データベースオプションの設定によっては、パーセント記号 (%) をコメントマーカーとして使用できました。このリリースでは、% 記号は SQL Anywhere でモジュロ演算子として処理されるようになりました。「MOD 関数 [数値]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SQLANY10 環境変数のサポート終了**　これまでのリリースでは、SQL Anywhere ソフトウェアの一部が共有ディレクトリにインストールされていました。この場所は SQLANYSH10 環境変数で指定できました。このリリースでは、インストールプロセスによってソフトウェアが共有ディレクトリにインストールされなくなり、SQLANYSH10 環境変数は使用されなくなりました。

サイレントインストールを作成するときに SHARED_DIR の場所を設定する必要がなくなりました。「SQL Anywhere インストーラーを使用したサイレントインストール」『SQL Anywhere サーバー プログラミング』を参照してください。

- **sa_get_server_messages システムプロシージャーのサポート終了** これまでのリリースでは、sa_get_server_messages システムプロシージャーを使用して、データベースサーバーメッセージウィンドウから結果セットとして定数を返すことができました。このリリースでは、sa_server_messages システムプロシージャーを使用して同じ情報を取得できるようになりました。「sa_server_messages システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **background_priority オプションは廃止予定** background_priority オプションは推奨されなくなりました。今後は priority オプションを使用してください。「priority オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **encrypt_aes_random_iv オプションのサポート終了** このリリースでは、encrypt_aes_random_iv データベースオプションがサポートされなくなりました。ランダム IV (初期化ベクトル) が常に使用されるようになりました。

- **DLL プロトコルオプションのサポート終了** DLL プロトコルオプションがサポートされなくなりました。Windows データベースサーバーとクライアントでは Winsock 2.2 が使用されます。Windows Mobile クライアントでは Winsock 1.1 が使用されます。「TCP/IP プロトコル」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL Anywhere Broadcast Repeater ユーティリティの名前変更** バージョン 10 では、SQL Anywhere Broadcast Repeater ユーティリティを実行するコマンドは dbns10 でした。このリリースでは、dbns11 です。「Broadcast Repeater ユーティリティ (dbns12)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQLANY10 と SQLANYSAMP10 の各環境変数の名前変更** SQLANY10 と SQLANYSAMP10 の各環境変数は、それぞれ SQLANY11 と SQLANYSAMP11 に名前が変更されました。次の項を参照してください。

  ○「SQLANY12 環境変数」『SQL Anywhere サーバー データベース管理』
  ○「SQLANYSAMP12 環境変数」『SQL Anywhere サーバー データベース管理』

Mobile Link

次の項では、Mobile Link バージョン 11.0.0 の新機能、動作の変更、廃止予定機能について説明します。

Mobile Link の新機能

次に、バージョン 11.0.0 で導入された Mobile Link の追加機能を示します。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

統合データベース

- **DB2 メインフレームを統合データベースとしてサポート**　Mobile Link では、これまで DB2 LUW (Linux、UNIX、Windows) が統合データベースとしてサポートされていました。このリリースでは、DB2 メインフレームもサポートされるようになりました。

- **MySQL を統合データベースとしてサポート**　Mobile Link で、統合データベースとして MySQL がサポートされるようになりました。

  「MySQL 統合データベース」『Mobile Link サーバー管理』を参照してください。

- **Mobile Link システムデータベース (MLSD)**　個別データベースに Mobile Link システムデータ (MLSD - Mobile Link システムデータベース) を保持できるようになりました。この機能は Microsoft DTC (分散トランザクションコーディネーター) とともに使用してください。「-cs mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

新しいシステムオブジェクト

- **新しい Mobile Link サーバーシステムテーブルとスキーマ**　Mobile Link システムテーブルは、次のように変更されています。

  ○複数の新しい Mobile Link システムテーブルが追加されました。

  - ml_qa_delivery_archive
  - ml_qa_repository_archive
  - ml_qa_repository_props_archive
  - ml_qa_status_history_archive
  - ml_server
  - ml_active_remote_id
  - ml_passthrough
  - ml_passthrough_repair
  - ml_passthrough_script
  - ml_passthrough_status

  ○複数の新しい Mobile Link システムプロシージャーが追加されました。次の項を参照してください。

  - 「ml_add_passthrough システムプロシージャー」『Mobile Link サーバー管理』
  - 「ml_add_passthrough_repair システムプロシージャー」『Mobile Link サーバー管理』
  - 「ml_add_passthrough_script システムプロシージャー」『Mobile Link サーバー管理』
  - 「ml_delete_passthrough システムプロシージャー」『Mobile Link サーバー管理』
  - 「ml_delete_passthrough_repair システムプロシージャー」『Mobile Link サーバー管理』
  - 「ml_delete_passthrough_script システムプロシージャー」『Mobile Link サーバー管理』

iAnywhere Solutions Oracle ドライバー

- **Oracle DSN が暗号化されたパスワードを格納**　Windows ODBC アドミニストレーターで Oracle ODBC データソースを作成するときに、ODBC データソースに格納されたパスワード

を暗号化するよう選択できるようになりました。「iAnywhere Solutions 12 - Oracle ODBC ドライバー」『Mobile Link サーバー管理』を参照してください。

- **Oracle ODBC ドライバーが Microsoft 分散トランザクションをサポート** Oracle ODBC ドライバーで Microsoft 分散トランザクションがサポートされるようになりました。Windows ODBC アドミニストレーターで、[Microsoft 分散トランザクションを有効にする] を選択し、Oracle クライアントとともに適切な DLL がインストールされていることを確認してください。「iAnywhere Solutions 12 - Oracle ODBC ドライバー」『Mobile Link サーバー管理』を参照してください。

Mobile Link サーバー

- **リレーサーバー** リレーサーバーは、Mobile Link サーバー、Afaria サーバー、および OneBridge サーバーと、モバイルデバイスとの間の、安全で負荷分散された Web サーバー経由の通信を実現する、Web 拡張機能セットです。「Relay Server の概要」『Relay Server』を参照してください。

- **Sybase リレーサーバーのホスティングサービス** Sybase リレーサーバーのホスティングサービスは、Sybase をホストとするリレーサーバーのファームです。Mobile Link データ同期を使用するモバイルアプリケーションの開発を容易にすること、また特に公共無線ネットワークを使用してデータを送信する場合に開発者による評価プロセスを簡素化することを目的としています。「Sybase ホストのリレーサービス」『Relay Server』を参照してください。

- **Mobile Link サーバーファーム** Mobile Link サーバーを同一のサーバーファームに明示的にグループ化できるようになりました。

  同一のリモート ID による冗長な同時同期をファーム全体から自動的に検出できるようになりました。これにより、負荷分散によって同一のリモート ID を Mobile Link サーバー に送信し続ける必要がなくなりました。

  各 Mobile Link サーバーに対して同一の設定ができるようになります。

  Notifier と QAnywhere コネクターに対するフェールオーバーが自動的にサポートされています。ファームにより、自動的に Mobile Link サーバーが指定されて Notifier と QAnywhere コネクターが実行されす。最初に指定されたコンピューターでフェールした場合は、新しいサーバーが選択されて Notifier と QAnywhere コネクターが実行されます。

- **64 ビットのプラットフォーム** いくつかの 64 ビットプラットフォームで Mobile Link サーバーが完全に 64 ビット対応になりました。サポートされる 64 ビットプラットフォームのリストについては、http://www.ianywhere.jp/sas/os.html を参照してください。

- **Java DownloadTableData インターフェイスの新しいメンバー** getLastDownloadTime メソッドによってテーブルの最終ダウンロード時刻が返されます。DownloadTableData.getLastDownloadTime メソッド [Mobile Link サーバー Java]『Mobile Link サーバー管理』を参照してください。

- **SQL パススルー** SQL パススルー機能によって、統合データベースから SQL Anywhere または Ultra Light のクライアントに SQL 文のスクリプトをダウンロードし、クライアント上で適切な時間に SQL 文を実行させることができます。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **情報メッセージの待機**　Java と .NET API では、プレフィクス "I" が付けられた情報の行がログに出力されたときに通知を受け取るよう登録できるようになりました。

  次の項を参照してください。

  - LogMessage.INFO 変数 [Mobile Link サーバー Java]『Mobile Link サーバー管理』
  - ServerContext.addInfoListener メソッド [Mobile Link サーバー Java]『Mobile Link サーバー管理』
  - ServerContext.removeInfoListener メソッド [Mobile Link サーバー Java]『Mobile Link サーバー管理』
  - LogMessage.MessageType 列挙体 [Mobile Link サーバー .NET]『Mobile Link サーバー管理』
  - ServerContext.InfoListener イベント [Mobile Link サーバー .NET]『Mobile Link サーバー管理』

mlsrv11 の新機能

- **-cs オプション**　システムテーブル、プロシージャー、トリガー、ビューなどの Mobile Link サーバーシステムオブジェクトを、統合データベース以外のデータベースに格納できるようになりました。Mobile Link システム オブジェクトを格納するデータベースは、Mobile Link システムデータベース (MLSD) と呼ばれます。

  MLSD の接続パラメーターを指定するには、-cs を使用します。「-cs mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-lsc オプション**　ローカルサーバーの接続情報を指定します。この情報は、サーバーファームのその他のサーバーに渡されます。「-lsc mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-ss オプション**　Mobile Link サーバーをサーバーファームで実行できます。

  注意
  -ss オプションは、Mobile Link の高可用性オプションの機能であり、別途ライセンスが必要です。「別途ライセンスが必要なコンポーネント」『SQL Anywhere 12 紹介』を参照してください。

- **-tc オプション**　SQL スクリプト実行のカウントダウンタイマーを設定します。「-tc mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-tf オプション**　-tc を使用して指定されたカウントダウンタイマーの期限が切れたら、SQL スクリプトを実行できないようにします (Oracle は対象外)。「-tf mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

Mobile Link の新しいスクリプト機能

- **非ブロッキングダウンロード ACK スクリプト**　「nonblocking_download_ack 接続イベント」『Mobile Link サーバー管理』と「publication_nonblocking_download_ack 接続イベント」『Mobile Link サーバー管理』を参照してください。

Mobile Link リダイレクターの強化

- **リダイレクターは廃止予定**　リダイレクターは推奨されなくなり、リレーサーバーに置き換えられました。「Relay Server の概要」『Relay Server』を参照してください。

- **新しいリレーサーバー**　リレーサーバーは、Mobile Link サーバー、Afaria サーバー、および OneBridge サーバーと、モバイルデバイスとの間の、安全で負荷分散された Web サーバー経由の通信を実現する、Web 拡張機能セットです。「Relay Server の概要」『Relay Server』を参照してください。

Mobile Link クライアント

SQL Anywhere クライアント

- **C++ および .NET 用の dbmlsync API**　dbmlsync API は、C++ または .NET で記述された Mobile Link クライアントアプリケーションが同期を起動し、要求した同期の進行状況に関するフィードバックを受け取れるようにするプログラミングインターフェイスです。「Dbmlsync API」『Mobile Link クライアント管理』を参照してください。

- **同期プロファイル**　dbmlsync コマンドラインを同期プロファイルとしてデータベースに格納できるようになりました。dbmlsync の -sp オプションを使用すると、同期プロファイルからコマンドライン同期オプションにオプションを追加できます。「-sp dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **挿入操作でロードされたローをロギングする LOAD TABLE のオプション機能**　新しい WITH ROW LOGGING 句によって、SQL Anywhere リモートデータベースで LOAD TABLE 文を使用してデータをロードできるようになりました。これまでは、LOAD TABLE ではローのログが取られずに dbmlsync から無視されていたため、LOAD TABLE は同期環境において必ずしも有益ではありませんでした。「LOAD TABLE 文を使用したデータのインポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **dbmlsync ログスキャンの最適化**　重複のないパブリケーション用に最適化されています。

セキュリティ

このリリースでは、次のセキュリティ機能が追加されています。

- **エンドツーエンド暗号化**　Mobile Link 同期ストリームおよびクライアントで、プロトコルレベルでエンドツーエンド暗号化がサポートされるようになりました。「エンドツーエンド暗号化」『SQL Anywhere サーバー データベース管理』を参照してください。

- **キーペアジェネレーターユーティリティ (createkey)**　このユーティリティでは、Mobile Link エンドツーエンド暗号化で使用する RSA と ECC のキーペアを作成します。「キーペアジェネレーターユーティリティ (createkey)」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

サーバー起動同期

- **Mobile Link サーバーファームのサポート**　SIS が強化され、Mobile Link サーバーファーム環境での動作が改善されました。ファーム内のすべての Mobile Link サーバーで Notifier を実行できるようになりました。これらの Notifier によって、同じ Listener への冗長な通知がなくなります。「-lsc mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

Listener の強化

- **新しい dblsn オプション**　Windows 用の Mobile Link Listener に次のオプションが追加されています。
  - **-ni**　IP 追跡を無効にします。
  - **-pc-**　永続的な接続を無効にします。
  - **-nu**　デフォルトの UDP 受信を無効にします。

Mobile Link の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された Mobile Link の変更点を示します。

Mobile Link サーバーの変更

- **デフォルトのダウンロード確認の変更**　-nba オプションのデフォルト値が -nba+ になり、デフォルトで非ブロッキングダウンロード確認が実行されるようになりました。既存の配備環境で、ダウンロード確認をブロックするダウンロードを使用する場合、次のいずれかの操作を行ってください。
  - -nba- オプションを使用する。
  - 同期スクリプトを確認して nonblocking_download_ack スクリプトか publication_nonblocking_download_ack スクリプトまたはその両方を使用する (推奨)。
- **certificate と certificate_password の各プロトコルオプションの名前変更**　TLS および HTTPS の certificate と certificate_password の各プロトコルオプションは、それぞれ identity と identity_password に名前が変更されました。「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

Mobile Link クライアントの変更

- **dbmlsync 統合コンポーネントは廃止予定**　dbmlsync 統合コンポーネントは推奨されなくなり、dbmlsync API に置き換えられました。

  「Dbmlsync API」『Mobile Link クライアント管理』を参照してください。
- **dbmlsync StreamCompression 拡張オプションのサポート終了**　このオプションはサポートされなくなりました。

- **-lt 拡張オプションのデフォルトは OFF に変更**　dbmlsync の LockTables (-lt) 拡張オプションのデフォルトは以前は ON でした。このリリースでは、デフォルトが OFF になり、デフォルトでは dbmlsync によって同期テーブルがロックされません。「LockTables (lt) 拡張オプション」『Mobile Link クライアント管理』を参照してください。

その他の Mobile Link の動作の変更

サーバー起動同期

- **Sierra Wireless Aircards のサポート終了**　Sierra Wireless Aircards の SMS 受信ライブラリは、サポートされなくなりました。
- **-g オプションは廃止予定**　-g Listener オプションは -ni オプションに置き換えられました。
- **-ga オプションは廃止予定**　-ga Listener オプションは推奨されなくなりました。非同期の IP 追跡は、暗黙的に行われます。
- **-gi デフォルトの変更**　デフォルトのポーリング間隔が、10 秒から 60 秒に変更されました。

QAnywhere

次の項では、QAnywhere バージョン 11.0.0 の新機能、動作の変更、廃止予定機能について説明します。

QAnywhere の新機能

次に、バージョン 11.0.0 で導入された QAnywhere の追加機能を示します。

新しい QAnywhere クライアント API

- **.NET API**　次の .NET API が追加されました。
  - CreateQAManager メソッド
  - CreateQAManager(Hashtable String)
  - CreateQATransactionalManager()
  - CreateQATransactionalManager(Hashtable String)
- **Java QAnywhere API の新しいインターフェイスクラス**　新しいクラスが追加されました。QAMessageListener2 インターフェイス [QAnywhere Java]『QAnywhere』を参照してください。
- **C# QAnywhere API の新しいデリゲート**
  - ExceptionListener デリゲート [QAnywhere .NET]『QAnywhere』
  - ExceptionListener2 デリゲート [QAnywhere .NET]『QAnywhere』
  - MessageListener2 デリゲート [QAnywhere .NET]『QAnywhere』

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

QAnywhere Agent の新機能

- **メッセージストアとして機能する Ultra Light**　QAnywhere で、メッセージストアとして Ultra Light がサポートされるようになりました。「qauagent ユーティリティ」『QAnywhere』を参照してください。

- **qastop -id オプション**　このオプションを使用すると、メッセージストア ID を qastop に指定して、指定したストア ID でデータベースに接続している QAnywhere Agent を停止できます。

- **新しい qauagent ユーティリティ**　Ultra Light 用の QAnywhere Agent です。「qauagent ユーティリティ」『QAnywhere』を参照してください。

その他の QAnywhere の強化

- **メッセージストアとして機能する Ultra Light**　QAnywhere で、メッセージストアとして Ultra Light がサポートされるようになりました。「qauagent ユーティリティ」『QAnywhere』を参照してください。

- **インクリメンタルアップロード／ダウンロードの強化**　インクリメンタルアップロードとインクリメンタルダウンロードでは、大きなメッセージを小さなメッセージに分割できます。「-iu qaagent オプション」『QAnywhere』と「-idl qaagent オプション」『QAnywhere』を参照してください。

- **設定プロパティとしてのデータベースタイプの指定**　QAnywhere Manager の設定プロパティとしてクライアントメッセージストアのデータベースタイプを指定できるようになりました。「QAnywhere Manager の設定プロパティ」『QAnywhere』を参照してください。

- **最終ステータス到達時におけるメッセージのアーカイブ**　メッセージが最終ステータスになり、永久的に削除されるまで待機しているメッセージを格納するためだけに使用されるアーカイブメッセージストアが用意されました。「アーカイブメッセージストア要求」『QAnywhere』を参照してください。

- **サーバー側の削除ルール**　サーバー側の削除ルールがアーカイブメッセージストアに適用されるようになりました。

- **サーバー管理要求の強化**　サーバー管理要求が拡張され、<actions> タグ内にオプションで追加可能な <actionsResponseId> タグが含まれるようになりました。

QAnywhere の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された QAnywhere の変更点を示します。

QAnywhere Agent の変更

- **qastop ユーティリティ**　qastop にオプションが指定されていないと、同じコンピューター上で実行中のすべての QAnywhere Agent が停止します。

- **Windows Mobile の予約スタックサイズ**　Windows Mobile のための、*qaagent.exe*、*dblsn.exe*、*dbmlsync.exe* の全スレッドの予約スタックサイズが縮小されました。

QAnywhere のその他の変更

- **非ブロッキングダウンロード確認**　正常にダウンロードされたメッセージのステータスは、同期の直後にサーバー内で「転送中」から「転送済み」に更新されます。

SQL Remote

次の項では、SQL Remote バージョン 11.0.0 の新機能、動作の変更、廃止予定機能について説明します。

SQL Remote の新機能

- **抽出ユーティリティ (dbxtract) の -ea オプションの強化**　dbxtract の -ea オプションでは、暗号化タイプとして、なし (none) と単純 (simple) の両方を使用できるようになりました。none を指定すると、暗号化は行われません。simple を指定すると、単純暗号化が行われます。また、-ea とともに -ek、-et、-ep のどのオプションが指定されたかに応じて、デフォルトの暗号化タイプが変更されました。「抽出ユーティリティ (dbxtract)」『SQL Remote』を参照してください。

- **抽出ユーティリティ (dbxtract) への -ap、-er、-et、-nl オプションの追加**　dbxtract で -ap、-er、-et、-nl の各オプションを使用できるようになりました。「抽出ユーティリティ (dbxtract)」『SQL Remote』を参照してください。

SQL Remote の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された SQL Remote の変更点を示します。

- **挿入操作でロードされたローをロギングする LOAD TABLE のオプション機能**　新しい WITH ROW LOGGING 句によって、SQL Anywhere リモートデータベースで LOAD TABLE 文を使用してデータをロードできるようになりました。これまでは、LOAD TABLE ではローのログが取られずに dbremote または dbltm から無視されていたため、LOAD TABLE は同期環境やレプリケート環境において必ずしも有益ではありませんでした。「LOAD TABLE 文を使用したデータのインポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **VIM と MAPI の各メッセージタイプのサポート終了**　このリリースでは、VIM と MAPI の各メッセージシステムがサポートされなくなりました。VIM または MAPI を使用するデータベースを SQL Anywhere バージョン 11.0.0 にアップグレードする場合は、メッセージタイプを File、FTP、または SMTP に変更してください。メッセージタイプが MAPI または VIM の場合、*dbremote.exe* は起動しません。

  最も簡単な変更方法は、SMTP/POP に切り替えることです。メッセージタイプを変更すると、SMTP/POP をサポートするためにメールサーバーの変更が必要になる場合があります。

  SQL Remote メッセージタイプの選択については、「SQL Remote メッセージシステム」『SQL Remote』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Ultra Light

次の項では、Ultra Light バージョン 11.0.0 の新機能、動作の変更、廃止予定機能について説明します。

Ultra Light の新機能

次に、バージョン 11.0.0 で導入された Ultra Light の追加機能を示します。

主な機能

- **同期プロファイルのサポート** Ultra Light 11.0.0 と 11.0.1 で同期プロファイルがサポートされます。「ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』、「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』、「DROP SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light SELECT 文** Ultra Light の SELECT 文に明示的な FOR 句がない場合、デフォルトで FOR READ ONLY になります。この変更によって、Ultra Light では、更新が許可されていないクエリにより適したプランを選択できるようになります。「SELECT 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light SYNCHRONIZE 文** Ultra Light 同期プロファイルまたは特定の同期オプションを同期するための新しい文です。「SYNCHRONIZE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light CREATE SYNCHRONIZATION PROFILE 文** Ultra Light 同期プロファイルを作成するための新しい文です。同期プロファイルによって、Ultra Light データベースが Mobile Link サーバーと同期する方法を定義します。「CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light ALTER SYNCHRONIZATION PROFILE 文** Ultra Light 同期プロファイルを変更するための新しい文です。「ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light DROP SYNCHRONIZATION PROFILE 文** Ultra Light 同期プロファイルを削除するための新しい文です。「DROP SYNCHRONIZATION PROFILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **SQL Anywhere パススルースクリプトのサポート** この機能は Ultra Light 12 ではサポートされていません。

Ultra Light ユーティリティで、SQL Anywhere パススルースクリプトがサポートされるようになりました。この変更点は次のユーティリティに適用されます。

- ulcond11
- ulunload
- ulload
- ulinfo
- ulsync

次の項を参照してください。

- 「Ultra Light データベースのアンロードユーティリティ (ulunload)」『Ultra Light データベース管理とリファレンス』
- 「Ultra Light データベースへの XML のロードユーティリティ (ulload)」『Ultra Light データベース管理とリファレンス』
- 「Ultra Light 情報ユーティリティ (ulinfo)」『Ultra Light データベース管理とリファレンス』
- 「Ultra Light 同期ユーティリティ (ulsync)」『Ultra Light データベース管理とリファレンス』

- **Ultra Light データベース検証**　ulvalid ユーティリティまたは ValidateDatabase API を使用して Ultra Light データベースを検証できるようになりました。検証によって、データベースファイル内に特定タイプの破損がないかテストされます。コマンドラインパラメーターを使用して結果を絞り込むことができます。「Ultra Light データベース検証ユーティリティ (ulvalid)」『Ultra Light データベース管理とリファレンス』と「Ultra Light データベースの検証」『Ultra Light データベース管理とリファレンス』を参照してください。

  Ultra Light .NET で ValidateDatabase 関数がサポートされるようになりました。接続の有無に関わらず、データベースや特定のテーブルを検証できるようになりました。ULDatabaseManager クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』と ULConnection クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  Sybase Central の**データベース検証ウィザード**を使用して、Ultra Light データベースを検証できるようになりました。**[データベースの検証]** オプションは、**[ツール]** メニューから選択できます。

- **イベントと通知のサポート**　Ultra Light でイベントと通知がサポートされるようになりました。イベントの発生時に、通知メッセージが登録されたキューまたは接続に送信されます。アプリケーションによるユーザーイベントの定義やトリガーも可能です。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。

- **Ultra Light における独立性レベルのサポート**　接続が複数ある場合、デフォルトで互いに独立するようになりました。他の接続によるコミットされていない変更およびダウンロードは、コミットされるまで参照できません。

  独立性レベルを READ_COMMITTED または READ_UNCOMMITTED に設定できるようになりました。「独立性レベルの変更」『Ultra Light データベース管理とリファレンス』と「Ultra Light でのトランザクション処理」『Ultra Light データベース管理とリファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Ultra Light .NET で ReadUncommitted 独立性レベルがサポートされるようになりました。オートコミットモードでの接続の独立性レベルは、デフォルトで ReadCommitted になります。「Ultra Light でのトランザクション処理」『Ultra Light データベース管理とリファレンス』と「独立性レベルの変更」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Ultra Light ALTER DATABASE SCHEMA FROM FILE 文** ALTER DATABASE SCHEMA FROM FILE 文を使用して Ultra Light スキーマを変更できるようになりました。ALTER DATABASE SCHEMA FROM FILE 文は、現在は削除されている UpgradeSchemaFromFile メソッドまたは ApplyFile メソッドによって実装されたバージョン 9.0.2 におけるスキーマアップグレード機能を置き換えるものとなります。ulinit ユーティリティまたは ulunload ユーティリティのいずれかを使用して、必要な DDL 文が構文的に正しいことを確認してください。

  次の項を参照してください。

  - 「ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light データベーススキーマのアップグレードの配備」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』
  - 「Ultra Light データベースのアンロードユーティリティ (ulunload)」『Ultra Light データベース管理とリファレンス』

- **データベース抽出ウィザードの動作の変更** 抽出プロセスからテーブルを除外できるようになりました。**データベース抽出ウィザード**では、使用可能なパブリケーションのリストから重複する名前の付いたパブリケーションが省略されます。「以前のバージョンの Ultra Light で作成したデータベースのアップグレード」 397 ページを参照してください。

- **Ultra Light クライアントバージョンとビルド番号の Mobile Link ログファイルへの追加** 同期中に、Ultra Light クライアントがバージョンとビルド番号を Mobile Link サーバーログに追加するようになりました。次の項を参照してください。

  - 「Mobile Link サーバーログの表示」『Mobile Link サーバー管理』
  - 「Ultra Light の同期パラメーター」『Ultra Light データベース管理とリファレンス』

- **Ultra Light LOAD TABLE 文** LOAD TABLE 文をデスクトップコンピューター上で実行できるようになりました。Ultra Light LOAD TABLE 文を参照してください。「LOAD TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **バックグラウンド同期のサポート** アプリケーションのどの時点からでも別のスレッドの同期を開始できるようになりました。Ultra Light では、アップロードの開始時にコミットされたローのみがアップロードされます。アップロード中にデータベースを変更し、アップロードに影響を及ぼさずにその変更をコミットできるようになりました。アップロード中にコミットされたローは、アップロードからは無視されます。「Ultra Light の同時実行性」『Ultra Light データベース管理とリファレンス』を参照してください。

- **強化された GUID 識別子のサポート** Ultra Light のこれまでのバージョンでは、16 バイトのバイナリまたは文字列で表した UUID (ユニバーサルユニーク識別子) または GUID (グローバルユニーク識別子) 識別子の入出力がランタイムにより許可されていました。これらの識別

子はエンディアン変換によって GUID 構造体と互換にされていました。Ultra Light 11 では、GUID 構造体は、ランタイムから明示的に入出力できるため、エンディアン変換が不要になりました。

- **ul_stream_error 構造体** Ultra Light 11 では、stream_id フィールド、stream_context フィールド、および error_string_length フィールドが削除されました。また、error_string フィールドが、ユーザー定義の char * から静的 char 配列に変更されました。

プラットフォームとデバイス

- **64 ビット Windows プラットフォーム (64 ビットの XP 以降) へのネイティブ amd64/x64 ESQL および C++ アプリケーション配備のサポート** Ultra Light で、64 ビット Windows プラットフォーム (64 ビットの XP 以降) に対するネイティブ amd64/x64 ESQL および C++ アプリケーションの配備がサポートされるようになりました。ただし、64 ビットのコンピューターで Ultra Light アプリケーションを開発するには、Ultra Light のユーティリティの 32 ビットバージョンを使用する必要があります。また、64 ビットのコンピューターで Ultra Light に接続する場合は、DBISQL と Sybase Central の 32 ビットバージョンを使用する必要があります。
- **Linux (32 ビット) にポートされた Ultra Light ユーティリティ** 「ユーティリティ」198 ページを参照してください。

セキュリティ

- **エンドツーエンド暗号化** Ultra Light で、プロトコルレベルでエンドツーエンド暗号化がサポートされるようになりました。データは、暗号ブロック連鎖 (CBC) モードで 128 ビット AES を使用して暗号化され、キー交換は RSA または ECC によって処理されます。

ユーティリティ

- **Ultra Light データベースの XML へのアンロードユーティリティ (ulunload)** -s オプションを使用してスキーマをアンロードし、SQL Anywhere 互換のフォーマットでデータを保存できるようになりました。「Ultra Light データベースのアンロードユーティリティ (ulunload)」『Ultra Light データベース管理とリファレンス』を参照してください。
- **Ultra Light データベース初期化ユーティリティ (ulinit)** -d オプションを使用して SQL Anywhere データベースから新しい Ultra Light データベースにデータをコピーできるようになりました。「Ultra Light データベース初期化ユーティリティ (ulinit)」『Ultra Light データベース管理とリファレンス』を参照してください。
- **ulerase** Ultra Light データベース (データベースに関連するテンポラリファイルを含む) を消去するための新しいユーティリティプログラムです。このユーティリティには、データベースへのアクセスを確認するためのユーザー ID とパスワードが必要です。「Ultra Light ユーティリティ消去 (ulerase)」『Ultra Light データベース管理とリファレンス』を参照してください。
- **ulvalid** Ultra Light データベースを検証するための新しいユーティリティプログラムです。検証によって、データベースファイル内に特定タイプの破損がないかテストされます。コマ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ンドラインパラメーターに従って検証を設定できます。「Ultra Light データベース検証ユーティリティ (ulvalid)」『Ultra Light データベース管理とリファレンス』を参照してください。

- **Linux にポートされた Ultra Light ユーティリティ (32 ビットバージョンのみで使用可能)**
  - **ulcreate** 新しい空の Ultra Light データベースを作成します
  - **ulerase** Ultra Light データベースと関連するチェックポイントファイルを永久的に消去します
  - **ulinfo** 既存の Ultra Light データベースに関する情報を表示します
  - **ulinit** SQL Anywhere リファレンスデータベースで使用可能なスキーマから新しい Ultra Light データベースを作成します
  - **ulload** ulunload によって保存された XML データから Ultra Light データベースを作成し、ロードします
  - **ulsync** Mobile Link を転送エージェントとして使用して、Ultra Light データベースを統合データベースと同期します
  - **ulunload** Ultra Light データベースを XML にアンロードします
  - **ulvalid** Ultra Light データベースに妥当性検査を実行します

ulunloadold ユーティリティは Linux では使用できません。

SQL

- **IF 文、CASE 文、CASE 式の強化** 互換性の向上のため、IF 式を ENDIF または END IF で終了できるようになりました。また、CASE 式を END または END CASE で終了できるようになりました。「IF 式」『Ultra Light データベース管理とリファレンス』と「CASE 式」『Ultra Light データベース管理とリファレンス』を参照してください。

プログラミングインターフェイス

一般的な向上点

パブリケーションマスクは、パブリケーションリストに置き換えられました。キーワード Publications は、パブリケーション名をカンマで区切ったリストで指定します。

Ultra Light for C/C++

新しいメソッド、UltraLite_Table* OpenTableEx() は、UltraLite_Connection オブジェクトの一部になりました。このメソッドによって、SQL 以外のアプリケーションからテーブルを開いたり、ローのダイレクトスキャンを実行する際の汎用性が向上します。「ダイレクトページスキャン」『Ultra Light データベース管理とリファレンス』を参照してください。

このメソッドを使用すると、次のいずれかの方法を指定してテーブルを開くことができます。

●プライマリキーのローを返すには、ul_table_open_primary_key を使用します。

- 任意の順序でローを返すには、ul_table_open_no_index を使用します。
- インデックスによって指定された順序でローを返すには、ul_table_open_with_index を使用します。

Ultra Light Embedded SQL

- エラー解釈に関連する 2 つの関数のマニュアルについては、ULGetErrorParameter メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』と ULGetErrorParameterCount メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』を参照してください。
- パブリケーションマスクからパブリケーションリストへの変更を反映するため、ULGetLastDownloadTime、ULResetLastDownloadTime、および ULCountUploadRows の各関数の構文が変更されました。
- 関数 ULGetPublicationMask は使用できなくなりました。

Ultra Light.NET

- ULDataReader クラスに、指定された最大ロー数までローカウントを取得できる新しいメソッド GetRowCount(threshold) が追加されました。
- ULDataReader クラスで IListSource インターフェイスが実装されるようになりました。

Ultra Light for M-Business Anywhere

- Connection クラスにイベント処理と通知のための新しいメソッド cancelGetNotification、createNotificationQueue、declareEvent、destroyNotificationQueue、getNotification、getNotificationParameter、registerForEvent、sendNotification、triggerEvent が追加されました。

Ultra Light J

Ultra Light J は Java による Ultra Light の実装で、Java SE 環境や BlackBerry スマートフォンを含む Java ME 環境をサポートします。「Ultra Light J」『Ultra Light - Java プログラミング』を参照してください。

Ultra Light の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された Ultra Light の変更点を示します。

廃止予定のプラットフォーム

- Ultra Light C++ インターフェイスで Symbian OS がサポートされなくなりました。Symbian 向けの Ultra Light アプリケーションの開発者は、Ultra Light J を使用してください。「Ultra Light J」『Ultra Light - Java プログラミング』を参照してください。
- Ultra Light.NET インターフェイスでは、.NET 1.0 コンポーネントがサポートされなくなりました。.NET 1.0 API リファレンスは、マニュアルから削除されました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 11 では、Ultra Light for AppForge はサポートされていません。

データベースプロパティ

このリリースでは、次のデータベースプロパティが推奨されなくなっています。

●CollationName

接続パラメーター

このリリースでは、Ultra Light の次の接続パラメーターが推奨されなくなりました。

●ORDERED_TABLE_SCAN

削除されたユーティリティ

C++ API 移行ウィザードは Sybase Central で使用できなくなりました。

削除、廃止予定、変更された関数

●ESQL インターフェイスの ULGetPublicationMask 関数は削除されました (パブリケーションマスクはパブリケーションリストに置き換えられました)。

●パブリケーションマスクからパブリケーションリストへの変更を反映するため、ESQL インターフェイスの ULGetLastDownloadTime、ULResetLastDownloadTime、および ULCountUploadRows の各関数の構文が変更されました。

●M-Business Anywhere API で、DatabaseSchema.getTableCountInPublications、DatabaseSchema.SYNC_ALL_DB、DatabaseSchema.SYNC_ALL_PUBS、PublicationSchema.getMask、SyncParms.getPublicationMask、SyncParms.setPublicationMask の各関数は推奨されなくなりました。

●M-Business Anywhere および .NET API では、次のメソッドはパラメーターとしてパブリケーションマスクを取っていました。このリリースでは、パラメーターがパブリケーションリスト (文字列) に変更されました。影響を受けるメソッドは、Connection クラスの countUploadRows、getLastDownloadTime、および resetLastDownloadTime です。

●ul_sync_info 構造体が変更され、**disable_concurrency**、**checkpoint_store**、**table_order** の各フィールドが削除されました。これらのオプションは、**additional_parms** という新しいフィールドに、キーワードと値の組み合わせをセミコロンで区切って指定します。

その他

● **CustDB の 2 番目のインスタンスの実行**　これまでのバージョンでは、CustDB の 2 番目のインスタンスを実行すると、エラーが発生しました。このバージョンでは、CustDB の 2 番目のインスタンスを起動すると、最初のインスタンスがフォアグランドに移され、2 番目のインスタンスは終了します。

Sybase Central と Interactive SQL

次の項では、バージョン 11.0.0 の Sybase Central と Interactive SQL での新機能、動作の変更、廃止予定機能について説明します。

Sybase Central と Interactive SQL の新機能

次に、バージョン 11.0.0 で導入された Sybase Central と Interactive SQL の追加機能を示します。

- **新しい高速ランチャー方式**　これまで、Interactive SQL および Sybase Central の高速ランチャーは、ユーザーのログイン時に起動していました。このバージョンでは、高速ランチャーは Interactive SQL または Sybase Central の起動時にのみ起動します。起動されると、高速ランチャーはアプリケーションのシャットダウン後 30 分間動作するよう設定されています。この新しい方式によって、30 分以内に実行される後続の Interactive SQL および Sybase Central の高速ランチャーをより短時間で起動できるようになります。「高速ランチャーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[接続] ウィンドウの強化と変更**　Sybase Central および Interactive SQL の **[接続]** ウィンドウが、次のように強化されています。

  - **新しい [ODBC データソースとして保存] ツール**　**[ODBC データソースとして保存]** ツールを使用すると、**[接続]** ウィンドウで指定した接続パラメーターを使用して ODBC データソースを生成できます。このツールを使用するには、**[接続]** ウィンドウの **[ツール]** ボタンをクリックして、**[ODBC データソースとして保存]** をクリックし、画面の指示に従います。「ODBC データソースの作成 (Sybase Central および Interactive SQL の場合)」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **新しい [接続文字列をクリップボードにコピー] ツール**　**[接続文字列をクリップボードにコピー]** ツールを使用すると、接続文字列をクリップボードにコピーして、データベースサーバーに渡されている値を確認できます。このツールを使用するには、**[接続]** ウィンドウの **[ツール]** ボタンをクリックして、**[接続文字列をクリップボードにコピー]** をクリックします。接続文字列をテキストエディターに貼り付けて表示します。

  - **接続アシスタント**　**[接続アシスタント]** ツールが、**[接続]** ウィンドウの右半分に表示されるようになりました。**[接続アシスタント]** は、データベースへの接続を支援するように設計された、ウィザードのような機能です。

  - **[詳細] タブの強化**　**[詳細]** タブに、プロパティの表とプロパティの簡単な説明が表示されるようになりました。

  - **新しい [ネットワーク] タブ**　**[ネットワーク]** タブを使用して、サポートされているプロトコルの共有メモリと TCP/IP のオプションを指定できます。この新しいタブが、これまで **[データベース]** タブの **[ネットワーク上でデータベースサーバーを検索]** オプションによって提供されていた機能を置き換えます。

- **キーボードショートカット**　Sybase Central のコードエディターと、Interactive SQL の **[SQL 文]** ウィンドウ枠で、キーボードショートカットを使用してコードとコメントのインデントを増減できるようになりました。また、キーボードショートカットを使用して、二重ハイフン

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

コメントと二重スラッシュコメントの両方のインジケーターを追加および削除できます。「Sybase Central キーボードショートカット」『SQL Anywhere サーバー データベース管理』を参照してください。

Sybase Central の新機能

次に、バージョン 11.0.0 で導入された Sybase Central のプラグインの追加機能を示します。

SQL Anywhere プラグインの新機能

- **データベースの [概要] タブ**　データベースサーバーとその機能の正常性と統計情報の概要を確認できるようになりました。「データベースの正常性と統計情報」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースドキュメント**　**データベースドキュメントウィザード**を使用してデータベースのドキュメントを生成できるようになりました。「データベースのドキュメント化」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいウィザード**　SQL Anywhere プラグインに、次の新しいウィザードが含まれるようになりました。

  - **ログインポリシー作成ウィザード**
  - **データベースドキュメントウィザード**
  - **テキストインデックス作成ウィザード**
  - **テキスト設定オブジェクト作成ウィザード**
  - **データベース作成ウィザード**

- **新しいプロパティウィンドウ**　SQL Anywhere プラグインに、次の機能のプロパティウィンドウが含まれるようになりました。

  - テキストインデックス
  - テキスト設定オブジェクト
  - 外部環境
  - マテリアライズドビュー
  - データベース監査

- **Sybase Central 内の監査ユーザー設定**　Sybase Central を使用して、データベース監査のユーザー設定を設定し、監査情報を表示できるようになりました。「監査の制御」『SQL Anywhere サーバー データベース管理』と「監査情報の取り出し」『SQL Anywhere サーバー データベース管理』を参照してください。

- **照合の適合化設定のサポート**　**[データベースのプロパティ]** ウィンドウの **[設定]** タブを使用して、照合の適合化オプションを指定できるようになりました。**データベース作成ウィザード**を使用して、新しいデータベースを作成するときに照合の適合化オプションを指定することもできます。

Ultra Light プラグインの新機能

- **新しいウィザード**　Ultra Light プラグインに、次の新しいウィザードが含まれるようになりました。

  - **データベース検証ウィザード**
  - **同期プロファイル作成ウィザード**

- **新しいプロパティウィンドウ**　Ultra Light プラグインに、次の機能のプロパティウィンドウが含まれるようになりました。

  - 同期プロファイル

Mobile Link プラグインの新機能

- **同期プロファイルのサポート**　Sybase Central で同期プロファイルを作成し、管理できます。

- **新しいウィザード**　Mobile Link プラグインに、次のウィザードが含まれるようになりました。

  - **パススルースクリプト作成ウィザード**
  - **パススルーダウンロード作成ウィザード**

- **新しいプロパティウィンドウ**　Mobile Link プラグインに、次の機能のプロパティウィンドウが含まれるようになりました。

  - パススルースクリプト
  - パススルーダウンロード

QAnywhere プラグインの新機能

- **Ultra Light のサポート**　QAnywhere プラグインで、クライアントメッセージストアとして Ultra Light がサポートされるようになりました。また、新しいメッセージアーカイブ機能のサポートも追加されています。「クライアントメッセージストアの設定」『QAnywhere』を参照してください。

Interactive SQL の新機能

次に、バージョン 11.0.0 で導入された Interactive SQL の追加機能を示します。

- **データベースロックの自動解放**　Interactive SQL でテーブルを表示すると、テーブルを変更しなくても、テーブルに対するスキーマロックがデータベースサーバーによって作成されます。

  このリリースでは、結果セットが表示されるときに作成されるデータベーススキーマロックを Interactive SQL が解放しようとします。

  結果セットを返す文を実行した後、データベースにコミットされていない変更が接続にあるかどうかが Interactive SQL によって確認されます。コミットされていない変更がなかった場合は Interactive SQL によってスキーマロックが解放されます。それ以外の場合、スキーマロックは解放されません。つまり、データベースにコミットされていない変更がある場合、スキーマロックは解放されません。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **読み込み専用結果セットのサポート**　Interactive SQL で結果セットを読み込み専用にできるようになりました。これを行うには、**[オプション]** » **[結果]** をクリックして、**[編集の無効化]** オプションをクリックします。この設定は、それ以降にフェッチされた結果に適用されます。「Interactive SQL の結果セットからテーブル値を編集する」『SQL Anywhere サーバー データベース管理』を参照してください。

  Interactive SQL を展開するときは、*OEM.ini* ファイル内の lockedPreferences のエントリに disableResultSetEditing を追加することで、ユーザーがこの設定を変更できないようにすることができます。「管理ツールの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL 文を 1 文ずつ実行する**　これまで、SQL 文を 1 文ずつ実行する場合、SQL 文を選択してから **[選択の実行]** を実行する操作を繰り返す必要がありました。このリリースでは、**[シングルステップ]** をクリックすると、選択した文が実行され、次に実行される文が選択されます。**[選択の実行]** と同様に、**[シングルステップ]** は Interactive SQL の **[SQL]** メニューから選択できます。「Interactive SQL での SQL 文の実行」『SQL Anywhere サーバー データベース管理』を参照してください。

- **お気に入りの SQL ファイルとデータベース接続**　よく使用するデータベース接続のリストと SQL スクリプトファイルのリストを、Interactive SQL の **[お気に入り]** メニューで作成および管理できるようになりました。「SQL スクリプトファイルと接続のお気に入りリストへの保存 」『SQL Anywhere サーバー データベース管理』を参照してください。

- **結果セットのテーブル編集を無効にする**　Interactive SQL を展開するとき、Interactive SQL からの SQL Anywhere と Ultra Light の結果セットのテーブル編集を無効にできるようになりました。「管理ツールの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

- **新しいキーボードショートカット**　Interactive SQL に新しいキーボードショートカットが追加されました。「Interactive SQL のキーボードショートカット」『SQL Anywhere サーバー データベース管理』を参照してください。

- **クライアントアプリケーションでのオプション変更を防止するためのサポート**　ユーザーが Interactive SQL のオプション設定を変更しないよう、*OEM.ini* ファイルで一部の設定をロックできるようになりました。「管理ツールの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

- **dbisql の新しい -version オプション**　コマンドプロンプトで dbisql -version と入力すると、Interactive SQL のバージョン番号が表示されます。「Interactive SQL ユーティリティ (dbisql)」『SQL Anywhere サーバー データベース管理』と「Ultra Light 用 Interactive SQL ユーティリティ (dbisql)」『Ultra Light データベース管理とリファレンス』を参照してください。

Sybase Central と Interactive SQL の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された Sybase Central と Interactive SQL の変更点を示します。

- **データベースツールのランチャーの再配備の簡素化**　Sybase Central、Interactive SQL、データベースコンソールユーティリティ、Mobile Link モニターの各ランチャー実行プログラムの再

配備が簡単になりました。JAR ファイルの場所に関するレジストリのエントリやセットディレクトリ構造が不要になりました。各実行プログラムには、(*filename.exe* ファイルと同じ名前の) 対応する *filename.INI* ファイルが、exe ファイルと同じディレクトリに必要です。*.INI* ファイルには、ツールをロードする方法の詳細が含まれます。「管理ツールの配備」『SQL Anywhere サーバー プログラミング』を参照してください。

- ***OEM.ini* [help] セクションのサポート終了**　*OEM.ini* ファイルの [help] セクションはサポートされなくなりました。詳細については、「管理ツールの設定」『SQL Anywhere サーバー プログラミング』を参照してください。

Sybase Central の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された Sybase Central の変更点を示します。

- **Sybase Central の設定ファイルの名前変更**　*.screpository* ファイルの名前が *.screpository600* に変更されました。

SQL Anywhere プラグインの変更機能

- **プロパティウィンドウの強化**　次のプロパティページが更新されました。

  - **Web サービスのプロパティ** ウィンドウ
  - **ユーザーのプロパティ** ウィンドウ
  - **ビューのプロパティ** ウィンドウ
  - **マテリアライズドビューのプロパティ** ウィンドウ

- **特定ユーザーのデバッグ**　SQL Anywhere プラグインで **[デバッグ]** モードを選択すると、デバッグの対象となるユーザーを指定する必要があります。「チュートリアル : デバッガーの使用開始」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **更新されたウィザード**　次のウィザードが更新されました。

  - **ユーザー作成ウィザード**
  - **Web サービス作成ウィザード**
  - **Deployment ウィザード**

Mobile Link プラグインの変更機能

- **更新されたウィザード**　次のウィザードが更新されました。

  - **同期モデル展開ウィザード**

SQL Anywhere プラグインの廃止予定機能

- **削除されたプロパティタブ**　次のプロパティウィンドウが更新されました。

  - **[テーブルのプロパティ]** ウィンドウ
  - **[Ultra Light プロジェクトのプロパティ]** ウィンドウ
  - **[Ultra Light 文のプロパティ]** ウィンドウ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Interactive SQL の動作の変更と廃止予定機能

次に、バージョン 11.0.0 で導入された Interactive SQL の変更点を示します。

- **プランビューアーでグラフィカルプランを表示可能**　Interactive SQL で、プランビューアーと呼ばれるサイズ変更可能な別ウィンドウで SQL Anywhere データベースのグラフィカルなプランを表示できるようになりました。この変更によって、[プランビューアー] ウィンドウを一度に複数開くことができるので、プランの表示と比較が簡単になります。プランビューアーにアクセスするには、**[ツール]** » **[プランビューアーを開く]** をクリックします。Ultra Light データベースのテキストプランもプランビューアーに表示されます。「Interactive SQL のプランビューアーとグラフィカルなプラン」『SQL Anywhere サーバー データベース管理』を参照してください。

  また、Interactive SQL の isql_plan オプションはサポートされていません。

- **長いプランと短いプランの表示はサポート終了**　SQL Anywhere データベースのテキストプランを Interactive SQL で表示できなくなりました。ただし、EXPLANATION 関数と PLAN 関数を使用してテキストプランを取得することはできます。Ultra Light データベースのテキストプランは、Interactive SQL でプランビューアーを使用して表示できます。

- **実行プランと結果セットの印刷**　[Ctrl + P] を押すか、**[ファイル]** » **[印刷]** をクリックして、**[SQL 文]** ウィンドウ枠の内容と結果セットを印刷できるようになりました。これまでは、**[SQL 文]** ウィンドウ枠の内容しか印刷できませんでした。プランビューアーで印刷するには、**[印刷]** ボタンをクリックします。「Interactive SQL での SQL 文、実行プラン、結果セットの印刷」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[SQL 文] ウィンドウ枠への行番号の追加**　**[SQL 文]** ウィンドウ枠の左側に行番号が表示されるようになりました。この行番号によって、構文エラーの場所を特定できるようになります。

- **[SQL 文の実行] ツールバーボタンの強化**　これまでは、[Interactive SQL] ツールバーで、**[SQL 文の実行]** ボタンをクリックするとすべての SQL 文が実行されていました。このリリースでは、ボタンのクリック時に、すべての文を実行するか選択されている文のみを実行するかを指定できます。

  [SQL 文の実行] ボタンの動作を設定するには、**[ツール]** » **[オプション]** » **[ツールバー]** をクリックします。「Interactive SQL での SQL 文の実行」『SQL Anywhere サーバー データベース管理』を参照してください。

- **バッチ文の実行の強化**

  ○Interactive SQL で文のバッチが実行されるときのフィードバックが改善されました。**[SQL 文]** ウィンドウ枠から SQL 文を実行する場合、実行されている文が選択され、全体が見えるようにスクロールされます。**[ファイル]** » **[スクリプトの実行]** をクリックしてスクリプトファイルを実行する場合、スクリプトの進行状況を示すステータスウィンドウが表示されます。「Interactive SQL での複数の SQL 文の実行」『SQL Anywhere サーバー データベース管理』を参照してください。

- **[結果] ウィンドウ枠の強化**

○**[結果]** ウィンドウ枠で、[Ctrl + A] を押すと、すべての結果を選択できるようになりました。現在フェッチされている結果だけでなく、結果セット全体も選択できます。**[結果]** ウィンドウ枠に結果セット全体が含まれていない場合、残りの結果をフェッチするよう要求されます。残りをフェッチしないと、現在フェッチされている結果のみが選択されます。

○**[結果]** タブからセルをコピーすると、コピーされたデータが isql_field_separator、isql_quote、isql_escape_character の各 Interactive SQL オプションに基づいてフォーマットされるようになりました。また、結果セットから選択した値、ロー、およびカラムをクリップボードにコピーすることもできます。「Interactive SQL の結果セットからカラム、ロー、セルをコピーする」『SQL Anywhere サーバー データベース管理』を参照してください。

○**[結果]** タブのカラムヘッダーをクリックすると、そのカラムを基準にして結果がソートされます。**[結果]** ウィンドウ枠に結果セット全体が含まれていない場合、残りの結果をフェッチするよう要求されます。残りをフェッチしないと、現在フェッチされている結果のみがソートされます。

○結果セットの選択されたローに基づいて INSERT 文、DELETE 文、UPDATE 文を生成し、クリップボードにコピーできるようになりました。「Interactive SQL での結果セットからの SQL 文の生成」『SQL Anywhere サーバー データベース管理』を参照してください。

○Interactive SQL の **[結果]** ウィンドウ枠が強化され、次の機能が追加されました。これらの機能は右クリックメニューから選択できます。

- **[コピー] » [セルのコピー]**　選択されたセルの内容をコピーします。
- **[コピー] » [カラムのコピー]**　選択されたセルのカラム値をコピーします。
- **[生成] » [INSERT 文]**　選択されている各ローに対して INSERT 文を生成し、クリップボードにコピーします。
- **[生成] » [DELETE 文]**　選択されている各ローに対して DELETE 文を生成し、クリップボードにコピーします。
- **[生成] » [UPDATE 文]**　選択されている各ローに対して UPDATE 文を生成し、クリップボードにコピーします。生成された文によって、カラムの値が現在の値に設定されます。したがって、文を実行しても、実際にはカラムの値は変更されません。この機能は、実行に先立って、編集可能なテンプレートの UPDATE 文を提供する際に便利です。

「Interactive SQL の結果セットからカラム、ロー、セルをコピーする」『SQL Anywhere サーバー データベース管理』と「Interactive SQL での結果セットからの SQL 文の生成」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Interactive SQL 文の強化**

○ **DESCRIBE 文の強化**　DESCRIBE 文によって、Interactive SQL に接続されているデータベースまたはデータベースサーバーに関する情報を返せるようになりました。「DESCRIBE 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **INPUT 文と READ 文の強化**　INPUT 文と READ 文で相対パスの解決を試行する際に、2 つの方法が取られるようになりました。「INPUT 文 [Interactive SQL]」『SQL Anywhere サー

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バー SQL リファレンス』と「READ 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **INPUT 文と OUTPUT 文の強化**

- **ODBC ソースからのインポートとエクスポートの新規サポート** INPUT 文と OUTPUT 文を使用してデータベースからインポートしたりデータベースにエクスポートしたりするときに、ODBC データソースを指定できるようになりました。これを行うには、新しい USING 句を使用します。「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』と「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  インポートウィザードと**エクスポートウィザード**を使用するときにも ODBC データソースを指定できます。「インポートウィザード (Interactive SQL) でのデータのインポート」『SQL Anywhere サーバー SQL の使用法』と「エクスポートウィザードを使用したデータのエクスポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **バイト順マーク (BOM) の新規サポート** データ中のバイト順マーク (BOM) を処理するかどうかを制御できるようになりました。これを行うには、新しい BYTE ORDER MARK 句を使用します。「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』と「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **INPUT 文のサポートフォーマットの変更** INPUT 文では、dBase、Lotus、Excel および FoxPro の各ファイルフォーマットがサポートされなくなりました。TEXT と FIXED は引き続きサポートされています。サポートされなくなったファイルフォーマットを引き続き使用したい場合は、ODBC ドライバーを使用する必要があります。「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **OUTPUT 文のサポートフォーマットの変更** OUTPUT 文では、dBase、Lotus、Excel および FoxPro の各ファイルフォーマットがサポートされなくなりました。TEXT、FIXED、HTML、SQL、および XML は引き続きサポートされています。「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **INPUT 文と OUTPUT 文で ASCII フォーマットの名前を TEXT に変更** INPUT 文と OUTPUT 文で TEXT が使用されるようになりました。ASCII の使用は下位互換性のためだけにサポートされています。

● **インポートウィザードとエクスポートウィザードの変更** **インポートウィザード**または**エクスポートウィザード**が終了するとき、ウィザードによって生成された SQL 文がコマンド履歴に保存されます。生成された SQL 文を表示するには、**[SQL] » [履歴]** をクリックします。

● **Interactive SQL オプション**

○ **isql_allow_read_client_file と isql_allow_write_client_file** これらの 2 つのオプションは、クライアント側のファイルの読み込み／書き込み要求に対する Interactive SQL の応答方法を記述します。「isql_allow_read_client_file オプション [Interactive SQL]」『SQL

Anywhere サーバー データベース管理』と「isql_allow_write_client_file オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-codepage オプションは廃止予定**　Interactive SQL に特定のコードページを含むファイルを読み込ませる場合、INPUT 文、OUTPUT 文、または READ 文の ENCODING 句を使用してください。次の項を参照してください。
  - 「Interactive SQL ユーティリティ (dbisql)」『SQL Anywhere サーバー データベース管理』
  - 「Ultra Light 用 Interactive SQL ユーティリティ (dbisql)」『Ultra Light データベース管理とリファレンス』
  - 「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』
  - 「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』
  - 「READ 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』
- **isql_plan オプションのサポート終了**　Interactive SQL の isql_plan オプションはサポートされなくなりました。下位互換のため、このオプションを設定しようとすると通知されずに無視されます。「Interactive SQL のプランビューアーとグラフィカルなプラン」『SQL Anywhere サーバー データベース管理』を参照してください。
- **SET OPTION 文の PUBLIC キーワードの削除**　SET OPTION 文を使用して Interactive SQL のオプションを設定するための PUBLIC キーワードはサポートされなくなりました。「Interactive SQL オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Interactive SQL ランチャーの変更**　Windows バージョンの Interactive SQL ランチャーの実行プログラムは、*dbisqlg.exe* から *dbisql.exe* に変更されました。

  コマンドラインバージョンの Interactive SQL ランチャーの実行プログラムは、*dbisql.exe* から *dbisql.com* に変更されました。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。

SQL Anywhere コンソールユーティリティの動作の変更

次に、バージョン 11.0.0 で導入された SQL Anywhere コンソールユーティリティの変更点を示します。

- **新しいコンソールオプション**　日付と時刻を指定して、**[オプション]** ウィンドウからデータベースサーバーを停止できるようになりました。**[ファイル]** メニューで **[オプション]** » **[コンソール]** をクリックします。

Mobile Link モニターの動作の変更

次に、バージョン 11.0.0 で導入された Mobile Link モニターの変更点を示します。

Mobile Link モニターでは、バージョン 9 以前の Mobile Link サーバーで作成されたモニターファイルを読み込むことができません。Mobile Link モニターは、同じメジャーバージョンの Mobile Link サーバーだけと使用してください。また、ワーカーカラムが削除され、Mobile Link モニターの次のプロパティの名前が変更されました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

古いプロパティ名	新しいプロパティ名
preload_upload	sync_request
verify_upload	authenticate_user

マニュアルの強化

- **マニュアルディレクトリの強化**　以前は、マニュアルは *%SQLANY11%¥docs* にありました。現在、マニュアルは *%SQLANY11%¥documentation* にあります。個々の HTML ヘルプファイルと PDF ファイルのファイル名も更新されました。

  HTML ヘルプの新しいファイル名は次のとおりです。

  - *dbadmin_ja11.chm*
  - *dbprogramming_ja11.chm*
  - *dbreference_ja11.chm*
  - *dbusage_ja11.chm*
  - *mlclient_ja11.chm*
  - *mlserver_ja11.chm*
  - *mlsisync_ja11.chm*
  - *mlstart_ja11.chm*
  - *qanywhere_ja11.chm*
  - *sachanges_ja11.chm*
  - *saerrors_ja11.chm*
  - *saintro_ja11.chm*
  - *scplugin_ja11.chm*
  - *sqlanywhere_ja11.chm*
  - *sqlremote_ja11.chm*
  - *uladmin_ja11.chm*
  - *ulc_ja11.chm*
  - *uldotnet_ja11.chm*
  - *ulj_ja11.chm*
  - *ulmbus_ja11.chm*

- **サポートされているプラットフォームのページに Web サイトからアクセス可能**　サポートされているプラットフォームに関する情報は、iAnywhere の Web サイト http://www.ianywhere.jp/sas/os.html にあります。

- **[Locate] ボタン**　HTML ヘルプブラウザーで、**[Locate]** ボタンを使用して、現在のヘルプページが目次のどこに含まれているかを確認できるようになりました。

製品全体の機能

次の項では、SQL Anywhere バージョン 11.0.0 のすべてのコンポーネントに影響する新機能、動作の変更、廃止予定機能について説明します。

製品全体の新機能

次に、バージョン 11.0.0 で導入された製品全体の追加機能を示します。

- **エラーレポートの強化** Windows、Windows Mobile、Linux では、エラーレポートが生成されるとウィンドウが表示されます。iAnywhere に送信するか選択する前に、このウィンドウでエラーレポートの内容を確認できます。「SQL Anywhere のエラーレポート」『SQL Anywhere サーバー データベース管理』を参照してください。

  -ce オプションを使用してサポートユーティリティ (dbsupport) を設定できるようになりました。このオプションを使用すると、dbsupport でアプリケーションを監視しているときにアプリケーションがクラッシュすると、電子メールが送信されます。「サポートユーティリティ (dbsupport)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **追加の Windows Mobile プラットフォームのサポート** SQL Anywhere で、スマートフォン用 Windows Mobile 5 と Windows Mobile 6 Standard エディションがサポートされるようになりました。Windows Mobile での SQL Anywhere サーバーの実行については、「SQL Anywhere for Windows Mobile」『SQL Anywhere サーバー データベース管理』と「インストール時の考慮事項 : Windows Mobile 5.0 for Smartphone に関する制限」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL Anywhere サンプルデータベース (*demo.db*) の新しいテーブル** SQL Anywhere サンプルデータベースに新しいテーブル MarketingInformation が追加されました。このテーブルの各ローには、Products テーブル内の製品を説明する HTML ページが格納されています。このテーブルが追加されたことで、機能をテストしたり試したりするときにより複雑な文字データに対してクエリを実行できます。「SQL Anywhere サンプルデータベース」『SQL Anywhere 12 紹介』を参照してください。

製品全体の動作の変更

次に、バージョン 11.0.0 で導入された製品全体の変更点を示します。

- **Windows CE から Windows Mobile への変更** Windows CE の名前を使用した方が正確である場合を除き、マニュアルとソフトウェアで Windows CE という名前が Windows Mobile に変更されました。

- **readcert、gencert、reqtool の削除** ユーティリティの readcert、gencert、および reqtool は削除されました。これらは、これまで廃止予定になっていました。これらに代わり、createcert と viewcert を使用できます。「証明書ユーティリティ」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **createcert と viewcert の各ユーティリティの Mac OS X でのサポート**　証明書ユーティリティの createcert と viewcert が、Mac OS X でサポートされるようになりました。「証明書ユーティリティ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **certificate と certificate_password の各プロトコルオプションの名前変更**　TLS および HTTPS の certificate と certificate_password の各プロトコルオプションは、それぞれ identity と identity_password に名前が変更されました。次の項を参照してください。

  - SQL Anywhere データベースサーバー：「-ec dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  - SQL Anywhere Web サーバー：「-xs dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  - SQL Anywhere プロトコルオプション：「Identity プロトコルオプション」『SQL Anywhere サーバー データベース管理』と「Identity_Password プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  - Mobile Link サーバー：「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **サンプル ID ファイルの変更**　サンプル証明書を含む ID ファイルと、対応する TLS のプライベートキーを含む ID ファイルの名前が、このリリースでそれぞれ変更されています。ファイル *rsaserver.crt* の名前は *rsaserver.id* に、ファイル *sample.crt* の名前は *eccserver.id* に変更されました。これらの ID ファイルのパスワードはともに、**tJ1#m6+W** から **test** に変更されました。

- **インストールディレクトリの変更**　このリリースでは、32 ビットのソフトウェアのインストール先が *win32* ディレクトリから *bin32* ディレクトリに、64 ビットのソフトウェアのインストール先が *X64* ディレクトリから *bin64* ディレクトリに変更されました。たとえば、これまでのバージョンで *C:¥Program Files¥SQL Anywhere 11¥win32* にインストールされたソフトウェアは、このリリースでは *C:¥Program Files¥SQL Anywhere 11¥bin32* にインストールされます。

- **サンプルデータベースの ODBC データソースの変更**　これまでのリリースでは、ソフトウェアとともにインストールされたサンプルデータベースの ODBC データソースはユーザーデータソースでした。このリリースでは、SQL Anywhere 11 Demo、SQL Anywhere 11 CustDB、QAnywhere 11 Demo の各データソースがシステムデータソースになりました。

- **.NET 1.0 のサポート終了**　SQL Anywhere 11 では Microsoft Visual Studio .NET 2002 または Visual Studio .NET 2003 はサポートされません。Microsoft Visual Studio 2005 (.NET 2.0) と Visual Studio 2008 (.NET 3.x) はサポートされています。

バージョン 10.0.1 の新機能

SQL Anywhere のバージョン 10 より前における新機能や動作の変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

SQL Anywhere

- 「新機能」215 ページ
- 「動作の変更と廃止予定機能」222 ページ

Mobile Link

- 「新機能」227 ページ
- 「動作の変更と廃止予定機能」229 ページ

QAnywhere

- 「新機能」230 ページ
- 「動作の変更と廃止予定機能」230 ページ

Ultra Light

- 「新機能」231 ページ
- 「動作の変更と廃止予定機能」233 ページ

SQL Remote

- 「動作の変更と廃止予定機能」231 ページ

製品全体の機能

- 「新機能」233 ページ
- 「動作の変更」235 ページ
- 「Windows Vista サポートの問題」235 ページ

SQL Anywhere

次の項では、SQL Anywhere のバージョン 10.0.1 の新機能、動作の変更、廃止される機能について説明します。

新機能

次に、バージョン 10.0.1 で導入された SQL Anywhere データベースとデータベースサーバーの新機能を示します。

暗号化の強化

暗号化のサポートを強化するために、次のような変更が加えられています。

- **CREATE DATABASE 文の ENCRYPTION 句の拡張**　CREATE DATABASE 文の ENCRYPTION 句の構文が拡張され、SIMPLE を暗号化タイプとして指定できるようになりました。また、暗号化キーとアルゴリズムを任意の順番で指定できます。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **dbinit と dbunload の -ea オプションの強化**　dbinit と dbunload の -ea オプションでは、暗号化タイプとして、なし (none) と単純 (simple) の両方を使用できるようになりました。none を指定すると、暗号化は行われません。simple を指定すると、単純暗号化が行われます。また、-ea とともに -ek、-et、-ep のどのオプションが指定されたかに応じて、デフォルトの暗号化タイプが変更されました。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』と「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

  -e オプションは廃止される予定です。「動作の変更と廃止予定機能」222 ページを参照してください。

- **Mac OS X での強力な暗号化**　Mac OS X で RSA 暗号化を使用して、クライアントとサーバー間の通信を暗号化できるようになりました。「SQL Anywhere クライアント／サーバー通信の暗号化」『SQL Anywhere サーバー データベース管理』を参照してください。

クライアントでの文のキャッシュのサポート

クライアントでの文のキャッシュがサポートされるようになり、デフォルトで有効です。この機能では、同じ SQL 文の準備と削除が繰り返し行われると、クライアントはこの文をキャッシュして、アプリケーションによって文が削除された後もサーバーで準備された状態にしておきます。これにより、データベースサーバーは、文の削除や再準備などの余分な作業を節約できます。クライアントでの文のキャッシュを使用するには、バージョン 10.0.1 のクライアントライブラリとバージョン 10.0.1 のデータベースサーバーの両方が必要です。

クライアントでの文のキャッシュをサポートするために、次のような変更が加えられています。

- **max_client_statements_cached オプション**　このオプションは、アプリケーションによって文が削除された後であっても、データベースサーバーでキャッシュ (準備) されたまま維持できる文の最大数を指定します。「max_client_statements_cached オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい接続プロパティとサーバープロパティ**　ClientStmtCacheHits、ClientStmtCacheMisses、max_client_statements_cached の各プロパティが追加されました。「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』と「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい要求の統計値**　文キャッシュヒットと文キャッシュミスの統計値が追加されました。「要求の統計値」『SQL Anywhere サーバー SQL の使用法』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

SQL FLAGGER の強化

SQL FLAGGER 機能が強化されて、互換性の検出が向上したほか、以降の標準のサポートが追加されました。たとえば、特定の SQL 標準との互換性や Ultra Light SQL との互換性をテストできるようになりました。

これらの強化をサポートするために、次のような変更が加えられています。

- **新しい SQLFLAGGER 関数**　新しい SQLFLAGGER 関数を使用すれば、実際に文を実行しなくても、SQL 文が指定された SQL 標準に準拠しているかどうかをテストできます。「SQLFLAGGER 関数 [その他]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい sa_ansi_standard_packages システムプロシージャー**　新しい sa_ansi_standard_packages システムプロシージャーを使用して、SQL 標準や SQL 文を指定したり、文の実行中に使用される非コア SQL 拡張の一覧を取得したりすることができます。「sa_ansi_standard_packages システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  この機能を使用するには、データベースをアップグレードする必要があります。「バージョン 10 以降のデータベースのアップグレード手順」 379 ページを参照してください。

- **sql_flagger_error_level データベースオプションと sql_flagger_warning_level データベースオプションの新しい値**　sql_flagger_error_level データベースオプションと sql_flagger_warning_level データベースオプションで新しい値がいくつか追加され、SQL/1999 標準と SQL/2003 標準がサポートされるようになりました。「sql_flagger_error_level オプション」『SQL Anywhere サーバー データベース管理』と「sql_flagger_warning_level オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL プリプロセッサー (sqlpp) の -e オプションと -w オプションの新しい値**　SQL プリプロセッサー (sqlpp) の -e オプションと -w オプションで新しい値がいくつか追加され、SQL/1999 標準と SQL/2003 標準がサポートされるようになりました。「SQL プリプロセッサー」『SQL Anywhere サーバー プログラミング』を参照してください。

SQL 文

SQL 文と関数には、次のように強化が行われています。

- **START DATABASE 文の強化**　START DATABASE 文では、データベースの DB 領域ファイルが配置されているディレクトリを指定できる DIRECTORY 句がサポートされるようになりました。「START DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **INSERT 文、UPDATE 文、DELETE 文、SELECT 文、UNION 文、EXCEPT 文、INTERSECT 文に OPTION 句が含まれる**　INSERT、UPDATE、DELETE、SELECT、UNION、EXCEPT、INTERSECT の各文には OPTION 句があるため、その文がマテリアライズドビューを使用す

る方法とクエリを最適化する方法を指定したり、次のデータベースオプションについて設定を上書きしたりすることができます。

- isolation_level
- max_query_tasks
- optimization_goal
- optimization_level
- optimization_workload

次の項を参照してください。

- 「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「UPDATE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「UNION 文」『SQL Anywhere サーバー SQL リファレンス』
- 「EXCEPT 文」『SQL Anywhere サーバー SQL リファレンス』
- 「INTERSECT 文」『SQL Anywhere サーバー SQL リファレンス』

- **HTML_DECODE 関数**　HTML_DECODE 関数は、トレードマーク記号 (™) など、数値エンティティとして指定された Unicode コードポイントをさらに多く復号化するようになりました。コードポイントがデータベースの文字セットで表すことができない場合は、コードポイント形式のままになります。以前は、0x7F より小さいコードポイントは文字に変換され (一部の文字セットでは 0xFF より小さいコードポイントが文字に変換されていました)、それ以外のコードポイントはすべてコードポイント形式のままでした。「HTML_DECODE 関数 [その他]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

照合の適合化のサポート

SQL Anywhere では、データベースの作成時に照合の適合化がサポートされるようになりました。照合の適合化をサポートするために、次のような変更が加えられています。

- **CREATE DATABASE 文の強化**　CREATE DATABASE 文または初期化ユーティリティ (dbinit) を使用してデータベースを作成するときに、文字のソートや比較を詳細に制御する適合化オプションを指定できるようになりました。

  CREATE DATABASE 文の場合は、COLLATION 句および NCHAR COLLATION 句を使用して照合の適合化がサポートされます。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  初期化ユーティリティの場合は、-z オプションおよび -zn オプションを使用して照合の適合化がサポートされます。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

注意
照合の適合化オプションを使用して作成したデータベースは、10.0.1 より前のデータベースサーバーでは起動できません。

既存のデータベースに対して照合の適合化を使用するには、照合の適合化をサポートする新しいバージョン 10.0.1 のデータベースを作成し、既存のデータベースをアンロードして、そのデータベースを新しいバージョン 10.0.1 のデータベースに再ロードします。「バージョン 10 以降のデータベースの再構築手順」371 ページを参照してください。

- **新しい HasCollationTailoring データベースプロパティ**　新しいデータベースプロパティ HasCollationTailoring は、データベースの作成時に適合化サポートが有効だったかどうかを示します。「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しく拡張されたプロパティ値**　Collation、NCHARCollation、CatalogCollation の各データベースプロパティを問い合わせるときに、新しい DB_EXTENDED_PROPERTY 値である CaseSensitivity、AccentSensitive、PunctuationSensitivity、Properties、Specification を利用できます。「DB_EXTENDED_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SORTKEY 関数と COMPARE 関数の拡張**　照合名をパラメーターとして使用できるだけでなく、SORTKEY 関数と COMPARE 関数では、CREATE DATABASE 文と同じ照合の適合化オプションをカッコで囲んで使用できるようになりました。「SORTKEY 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』と「COMPARE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Web サービスの強化

HTTP ヘッダーと SOAP ヘッダーの設定可能性を改善するために、次のような強化が行われています。

- **設定可能性の改善**　CREATE PROCEDURE 文と CREATE FUNCTION 文の新しい SET 句では、HTTP プロトコルや SOAP プロトコルのオプションを修正できます。修正できるオプションは、クライアントが使用する HTTP バージョン、チャンクを使用するかどうか、SOAP 要求で呼び出す SOAP 処理の名前 (プロシージャーや関数の名前と異なる場合) です。「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **HTTP ヘッダー仕様**　CREATE PROCEDURE 文と CREATE FUNCTION 文の HEADER 句の構文が拡張され、指定された HTTP 要求ヘッダーを抑制したり、空の値を指定したりすることができるようになりました。この機能は、以前のリリースでは修正できなかった自動的に生成された HTTP 要求ヘッダーにも適用されます。「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』と「HTTP 要求ヘッダーの管理」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SOAP:RPC クライアントでのデータ型のサポート**　CREATE SERVICE 文の DATATYPE 句を使用して、データ型指定を有効にできます。データ型情報は、すべての SOAP サービスフォーマットでパラメーター入力と結果セット出力 (応答) の XML エンコードに含まれます。これにより、パラメーターを String に明示的に変換するクライアントコードが不要になるため、

SOAP ツールキットからのパラメーター受け渡しが簡単になります。「SOAP のデータ型」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Mac OS X での HTTPS サポート**　以前のリリースでは、Mac OS X では HTTP プロトコルのみがサポートされていました。今回のリリースでは、Mac OS X 上で SQL Anywhere データベースサーバーを Web サーバーとして実行しているときに、HTTPS を使用できるようになりました。「-xs dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』と「HTTP Web サーバーとしての SQL Anywhere」『SQL Anywhere サーバー プログラミング』を参照してください。

データベースミラーリングの強化

データベースのミラーリング機能に、次の機能強化が追加されています。

- **データベースミラーリングにおける優先データベースサーバーの指定**　ミラーリングシステムでプライマリサーバーのロールを引き受けるデータベースサーバーを指定できるようになりました。「-xp dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』と「優先データベースサーバー」『SQL Anywhere サーバー データベース管理』を参照してください。

- **プライマリサーバーからミラーサーバーへのデータベースミラーリングのフェールオーバーの起動**　ALTER DATABASE 文の SET PARTNER FAILOVER 句を使用して、プライマリサーバーからミラーサーバーへのデータベースミラーリングのフェールオーバーを起動できるようになりました。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』と「プライマリサーバーのフェールオーバー」『SQL Anywhere サーバー データベース管理』を参照してください。

SQL Anywhere プラグイン

- **[トリガー] フォルダーのカラム名の変更**　[トリガー] フォルダーの [テーブル名] カラムと [テーブル所有者] カラムは、[オブジェクト名] カラム、[オブジェクト所有者] カラム、[オブジェクトタイプ] カラムに置き換えられました。[オブジェクトタイプ] カラムはデフォルトで表示されませんが、[ビュー] » [カラムの選択] を選択すると表示することができます。

- **[ビューのプロパティ] ウィンドウに追加された [トリガー] タブ**　非マテリアライズドビューのプロパティウィンドウに、ビューの INSTEAD OF トリガーをリストする [トリガー] タブが追加されました。

- **トリガー作成ウィザードに追加された INSTEAD OF トリガーのサポート**　トリガー作成ウィザードでいくつか機能が強化され、INSTEAD OF トリガーがサポートされるようになりました。これには、テーブルと非マテリアライズドビューのどちらに対するトリガーを作成するかを選択するオプションの追加も含まれます。「トリガーの作成」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **データベース作成ウィザードに追加された照合の適合化のサポート**　選択したデータベースサーバーがバージョン 10.0.1 以降のサーバーである場合、または新しいデータベースサーバーを起動してローカルコンピューターにデータベースを作成することにした場合、データベース作成ウィザードに照合の適合化のページが含まれるようになりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

その他の機能強化

- **単純な DML 文でのプランのキャッシュ**　プランのキャッシュが拡張され、クエリを省略できる INSERT 文、UPDATE 文、DELETE 文 (単純な文) が含まれるようになりました。「プランのキャッシュ」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **左外部ジョインや右外部ジョインを含むマテリアライズドビューがコストベースの最適化中に使用可能**　以前は、左外部ジョインや右外部ジョインは、マテリアライズドビューの定義で使用できました。しかし、これが原因で、マテリアライズドビューをコストベースの最適化で使用することができませんでした。このバージョンでは、左外部ジョインや右外部ジョインを含むマテリアライズドビューがコストベースの最適化中に使用できるようになりました。「マテリアライズドビューを使ったクエリのパフォーマンス改善」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **INSTEAD OF トリガーのサポート**　BEFORE トリガーまたは AFTER トリガーは、それぞれトリガー操作の前または後に起動されます。INSTEAD OF トリガーは、トリガー操作を置き換えます。INSTEAD OF トリガーを使用することで、挿入、更新、削除の各操作中にトリガーの動作をより細かく制御できます。「CREATE TRIGGER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **DBTools の強化**　DBCreatedVersion 関数を使用することで、データベースが SQL Anywhere 10.0.0 またはそれ以前のバージョンを使用して作成されたかどうかをデータベースを起動せずに判別できるようになりました。DBCreatedVersion メソッド [データベースツール]『SQL Anywhere サーバー プログラミング』を参照してください。

- **OLAP の強化**　新しい 2 つの Window 集合関数 FIRST_VALUE と LAST_VALUE がサポートされるようになりました。これらの関数は、ウィンドウの最初または最後の値をそれぞれ返すため、セルフジョインを使用してこれらの値を返す必要がありません。ウィンドウ上で実行される以後の計算では、これらの値をベースラインとして使用できます。「FIRST_VALUE 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』と「LAST_VALUE 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **強化された UNIX での IPv6 サポート**　UNIX では、IPv6 アドレスの一部としてインターフェイス識別子かインターフェイス名を指定できます。Linux (カーネル 2.6.13 以降) では、クライアントまたはサーバーで IP アドレスを指定するとき (たとえば HOST=、MYIP=、または BROADCAST= TCP プロトコルオプションを使用するとき) に、インターフェイス識別子が必要です。「SQL Anywhere での IPv6 サポート」『SQL Anywhere サーバー データベース管理』を参照してください。

- **TDS DATE データ型と TDS TIME データ型のサポート**　TDS DATE データ型と TDS TIME データ型は、最近になって TDS クライアントに導入されました。Open Client 15 以降のバージョンまたは jConnect の EBF を使用するアプリケーションでは、日付カラムや時刻カラムを TDS DATETIME ではなく TDS DATE 値または TDS TIME 値としてフェッチできるようになりました。

  SQL Anywhere は、TDS ベースのアプリケーションで日付や時刻のデータを TDS DATE 値や TDS TIME 値としてフェッチできるよう強化されています。Open Client や jConnect の古いバージョンを使用するアプリケーションでは、日付や時刻のデータを引き続き TDS DATETIME としてフェッチします。TDS ベースでないアプリケーション (Embedded SQL、

ODBC、iAnywhere JDBC ドライバーを使用するアプリケーション) では、これまでも日付や時刻のデータを日付値や時刻値としてフェッチ可能でした。

- **使用可能な文字セットエンコードをリストする新しい dbinit オプション**　データベースで使用可能な文字セットエンコードをリストするには、初期化ユーティリティ (dbinit) の -le オプションを使用します。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい -ds サーバーオプション**　-ds サーバーオプションを使用して、データベースの DB 領域ファイルが配置されている場所を指定できます。「-ds dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SADbType.Xml データ型**　SADbType.Xml 列挙定数が SQL Anywhere .NET プロバイダーに追加されました。

- **SQL Anywhere SNMP Extension Agent の動的トラップにおける単位のサポート**　動的トラップを設定するとき、k、m、g、t を使用して、トラップの数値をそれぞれキロバイト、メガバイト、ギガバイト、テラバイトの単位で指定できるようになりました。「動的トラップ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **iAnywhere Solutions Oracle ドライバー用 ODBC データソースの作成**　データソースユーティリティ (dbdsn) で -or オプションを指定して、iAnywhere Solutions Oracle ドライバー用の ODBC データソースを作成できるようになりました。「データソースユーティリティ (dbdsn)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **エラーレポートを送信するウィンドウの強化**　エラーレポートを iAnywhere に送信するかどうかを確認する dbsupport ウィンドウに、[エラーレポートの表示] ボタンが追加され、エラーレポートに含まれる情報を送信前に確認できるようになりました。

動作の変更と廃止予定機能

次に、バージョン 10.0.1 で導入された SQL Anywhere データベースとデータベースサーバーに加えられた変更を、カテゴリごとに示します。

動作の変更

- **クエリ内並列処理を使用するときの変更**　クエリ内並列処理は、background_priority が on に設定されている接続には使用されなくなりました。また、現在要求を処理しているサーバースレッド数 (ActiveReq サーバープロパティ) が、データベースサーバーの使用ライセンスがあるコンピューターの CPU コア数を最近超えた場合は、クエリ内並列処理は使用されません。「高度：クエリ実行時の並列処理」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  新しいサーバープロパティである ExchangeTasksCompleted は、データベースサーバーの起動後にクエリ内並列処理に使用された内部タスクの総数を返します。「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **暗号化の結果は決定的ではなくなった**　以前は、ENCRYPT 関数を使用した値の暗号化は決定的でした。2 つの同一な入力文字列と 2 つの同一な暗号化キーを入力すると、同一の出力データ (暗号文) が返されていました。新しい encrypt_aes_random_iv データベースオプションでは、暗号化が決定的であるかどうかを制御できるようになりました。新しいデフォルトの動作は、非決定的です。

  注意
  このデータベースオプションのないデータベースサーバー (バージョン 10.0.0 以前) では、たとえ Off であってもこのオプションが設定されたデータベースからはデータを復号化できません。

- **ALTER DBSPACE RENAME では、DB 領域が開かれていない場合に開こうとする**　以前は、DB 領域を使用するデータベースが起動し、そのうちのいずれかの DB 領域が見つからなかった場合、その DB 領域に対して ALTER DBSPACE...RENAME 文を実行すると、カタログで DB 領域名が更新されましたが、DB 領域の起動は試みられませんでした。今回のリリースでは、カタログの更新後に、データベースサーバーが DB 領域を開こうとするようになりました。「ALTER DBSPACE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE DBSPACE 文、ALTER DBSPACE 文、DROP DBSPACE 文に対する変更**　CREATE DBSPACE 文と DROP DBSPACE 文では、事前に定義された DB 領域の名前 (SYSTEM、TEMPORARY、TEMP、TRANSLOG、TRANSLOGMIRROR) を受け入れなくなりました。以前のバージョンの SQL Anywhere データベースサーバーで作成されたデータベースのユーザー DB 領域が、事前に定義された DB 領域のいずれかと名前が同じ場合、データベースサーバーは常にユーザー DB 領域を参照します。「事前定義の DB 領域」『SQL Anywhere サーバー データベース管理』を参照してください。

- **sa_conn_info システムプロシージャーの出力に対する変更**　sa_conn_info システムプロシージャーの出力が変更され、接続が待機しているロックに関して詳細な情報が得られるようになりました。LockName フィールドは削除されました。それに代わって、LockRowID と LockIndexID という新しい 2 つのフィールドが追加されました。特定のロー識別子に関連付けられているロックで接続が待機している場合は、LockRowID にそのロー識別子が含まれます。特定のインデックスに関連付けられているロックで接続が待機している場合は、LockIndexID にそのインデックスの識別子が含まれます。「sa_conn_info システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  この機能を使用するには、データベースをアップグレードする必要があります。「バージョン 10 以降のデータベースのアップグレード手順」 379 ページを参照してください。

- **一部のシステムプロシージャーで DBA パーミッションは不要になった**　sa_dependent_views、sa_get_dtt、sa_check_commit、sa_materialized_view_info の各システムプロシージャーでは、DBA パーミッションを必要とせず実行できるようになりました。

- **CREATE DATABASE 文のデフォルトの測定単位の削除**　CREATE DATABASE 文を使用してデータベースを作成するときに DATABASE SIZE の値を指定する場合、オプションだった測定単位の指定は必須になりました。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **default_timestamp_increment オプションの新しい最大値**　default_timestamp_increment オプションの最大値が 1000000 (1 秒) になりました。「default_timestamp_increment オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- ***dbdata10.dll* の削除**　dbdata10 DLL (Dynamic Link Library) で提供されていた機能は、SQL Anywhere .NET プロバイダー DLL に組み込まれました。その結果、Windows CE では、SQL Anywhere .NET プロバイダー DLL にプラットフォーム固有のバージョンが用意されました。

- **NetWare におけるデフォルトキャッシュサイズの増加**　NetWare におけるデータベースサーバーのデフォルトキャッシュサイズが 2 MB から 8 MB に増加されました。「-c dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **クライアントでの文のキャッシュによる動作の変更**　クライアントでの文のキャッシュをサポートした結果、次のような動作の変更が導入されました。

  ○同じ SQL 文で結果セットがないと記述された後になって結果セットが存在するような場合は、不正な記述が発生することがあります。次に例を示します。

  ```
  CREATE PROCEDURE p() NO RESULT SET BEGIN ... END
  Prepare, Describe, Drop "call p"
  ALTER PROCEDURE p() RESULT( ... ) BEGIN ... END
  Prepare, Describe, Drop "call p"  // describe returns no result set
  ```

  ○クライアントでの文のキャッシュが有効で、RememberLastStatement が有効な場合 (-zl サーバーオプション)、キャッシュされた文を再使用するときの LastStatement プロパティは空の文字列になります。

  ○クライアントでの文のキャッシュが有効な場合、sa_get_request_times または sa_get_request_profile を使用して要求レベルログを処理する場合、文の実行回数が不正である可能性があります。「max_client_statements_cached オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **言語ユーティリティ (dblang) に管理者権限は不要になった**　以前のバージョンの SQL Anywhere では、dblang ユーティリティを使用して SQL Anywhere のローカライズされたバージョンの言語設定を変更するために、ユーザーは管理者としてログインする必要がありました。この要件はなくなりました。

- **DISH サービス名にスラッシュを使用できなくなった**　DISH サービス名が誤って解釈されないように、スラッシュ (/) をサービス名の一部として使用できなくなりました。「CREATE SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SACommand.UpdateRowSource のデフォルト値の変更**　これまで SACommand.UpdatedRowSource のデフォルト値は UpdatedRowSource.Both でしたが、UpdatedRowSource.OutputParameters に変更されました。SACommand.UpdatedRowSource プロパティ [SQL Anywhere .NET]『SQL Anywhere サーバー プログラミング』を参照してください。

- **PrefetchRows 接続パラメーターのデフォルト値の変更**　.NET データプロバイダーを使用するとき、PrefetchRows 接続パラメーターのデフォルト値が 10 から 200 に変更され、パフォー

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

マンスが向上しました。SAConnectionStringBuilder.PrefetchRow のデフォルト値も 200 に変更されました。結果セットに BLOB カラムが含まれる場合、プリフェッチは無効です。「PrefetchRows (PROWS) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **認証アプリケーションでの *saopts.sql* から *authenticate.sql* の使用への変更**　以前のリリースの SQL Anywhere OEM 版では、データベースの作成、再構築、または更新を行うたびに適用されるように、認証文をファイル *%SQLANY10%¥scripts¥saopts.sql* に格納するように推奨されていました。

  今回のリリースでは、認証文字列をファイル *%SQLANY10%¥scripts¥authenticate.sql* に格納するように推奨されるようになりました。「認証データベースのアップグレード」『SQL Anywhere サーバー データベース管理』を参照してください。

- **HP-UX で長いホスト名を使用できるようになった**　HP-UX 11i v2 September 2004 Update より、システム管理者はカーネルパラメーターを設定することで 255 バイトのホスト名のサポートを有効にできるようになりました。しかし、長いホスト名のサポートが有効な HP-UX コンピューター上の SQL Anywhere サーバーで、MachineName プロパティと AppInfo HOST キーは最長で 64 バイトのホスト名を返していました。今回のリリースでは、MachineName と AppInfo の両方で、255 バイトのホスト名を返せるようになりました。

- **iAnywhere JDBC ドライバーの URL ヘッダー**　以前のリリースでは、アプリケーションが iAnywhere JDBC ドライバーを使用して SQL Anywhere に接続した場合、JDBC ドライバーに渡される URL はヘッダー **jdbc:odbc:** で始まっていました。今回のリリースでは、**jdbc:ianywhere:** で始まる URL ヘッダーも使用できるようになりました。Sun JDBC-ODBC ブリッジとの競合を避けるため、**jdbc:ianywhere:** を使用することをおすすめします。「ドライバーへの URL の指定」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Remarks 値が 128 文字よりも長い場合の jConnect を使用したテーブルリストの取得**　以前は、JDBC アプリケーションが jConnect を使用して接続し、テーブルのリストを要求した場合、テーブルが存在していたとしても、結果は空になる可能性がありました。string_rtruncation オプションが On に設定され、アプリケーションが DatabaseMetaData.getTables メソッドを使用し、任意のテーブルの Remarks 値が 128 文字より長い場合に、このような状況が発生していました。今回のリリースでは、長すぎる Remarks 値は 128 文字にトランケートされるようになり、テーブルのリストが返されます。この変更を使用するには、*jcatalog.sql* を実行するか、データベースをアップグレードする必要があります。「jConnect システムオブジェクトのデータベースへのインストール」『SQL Anywhere サーバー プログラミング』または「アップグレードユーティリティ (dbupgrad)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **CHAR 値と NCHAR 値の比較**　SQL Anywhere 10.0.0 では、CHAR ドメインと NCHAR ドメインを組み合わせると、NCHAR の比較になっていました。しかしこれにより、SQL_C_WCHAR としてバインドされたホスト変数を使用すると、10.0.0 にアップグレードしたアプリケーションでは異なる結果が得られたり、パフォーマンスが低下したりする可能性があります。SQL Anywhere 10.0.0 では、SQL_C_WCHAR としてバインドされた変数は NCHAR として表されます。SQL Anywhere 10.0.1 では新しい推定規則が導入されて、既存のアプリケーションとの互換性が向上し、CHAR ドメインと NCHAR ドメインを組み合わせたときに一貫性のある予測可能な結果を提供するようになりました。「CHAR と NCHAR の比較」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **2.6.12 より前の Linux カーネルで非同期 I/O が無効**　2.6.12 より前の Linux カーネルでのバグにより、影響のあるカーネルのいずれかで SQL Anywhere データベースサーバーを実行すると、デフォルトで非同期 I/O が無効になります。非同期 I/O を使用する場合は、カーネルを 2.6.12 以降にアップグレードする必要があります。

- **OEM 版マニュアルの削除**　以前のリリースの SQL Anywhere OEM 版では、認証アプリケーションを設定するための手順は、*.pdf* または *.html* ファイルに別途記載されていました。今回のリリースでは、この情報は次の場所に記載されています。

  ○「認証 SQL Anywhere アプリケーション」『SQL Anywhere サーバー データベース管理』
  ○「connection_authentication オプション」『SQL Anywhere サーバー データベース管理』
  ○「database_authentication」『SQL Anywhere サーバー データベース管理』

- **アンロードユーティリティの -e と -t オプションで、大文字と小文字が区別されるデータベースにおけるテーブル名の大文字と小文字の区別は不要になった**　以前のリリースでは、dbunload ユーティリティを使用し、-e オプションまたは -t オプションを指定して大文字と小文字が区別されるデータベースをアンロードする場合、これらのオプションには大文字と小文字を区別したテーブル名を指定する必要がありました。今回のリリースでは、テーブル名は大文字と小文字が区別されません。

- **テンポラリテーブルへのデータのロード**　テンポラリテーブルにデータをロードする際の動作が変更されました。ON COMMIT DELETE ROWS と指定して定義された LOCAL TEMPORARY TABLE を除いて、コミットは、テンポラリテーブルで LOAD TABLE を実行する前後に自動的に実行されます。ロードに失敗すると、テンポラリテーブル内のすべてのロー (ロードの前に存在したローを含む) が削除されるようになりました。

  ON COMMIT DELETE ROWS で定義された LOCAL TEMPORARY TABLE の場合、動作の変更はなく、コミットは実行されません。つまり、ロード中に発生した障害によって部分的になったロードの場合は、この種のテンポラリテーブルには、ロードされたローの一部しか含まれず、ロード以前に存在したローが見つからないこともあります。

  また、別のテーブルの外部キーから参照されるローがテーブルに含まれている場合、テンポラリテーブルへのロードは失敗します。

  ON COMMIT DELETE ROWS を指定して定義された GLOBAL TEMPORARY TABLE にはロードできません。

- **UCA 照合を使用した日本語のデータベースに対する大文字と小文字の区別のデフォルト設定**　日本語のデータベースを作成する場合の UCA 照合では、デフォルトで、大文字と小文字およびアクセント記号が区別されます。日本語のデータベースとは、OS の言語または文字セットが日本語であるコンピューターで作成されたデータベースや、932JPN や EUC_JAPAN などの日本語の CHAR 照合で作成されたデータベースです。

  日本語以外のデータベースを作成する場合の UCA 照合では、デフォルトで、大文字と小文字が区別されません。

  大文字と小文字およびアクセント記号の区別のデフォルト設定は、dbinit の -c と -a (または -c- と -a-) オプションを指定して、それぞれ上書きすることができます。また、照合の適合化構文や、CREATE DATABASE 文の CASE 句と ACCENT 句を使用して上書きすることもできます。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』と

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

廃止予定機能とサポート終了機能

- **SQL FLAGGER での SQL/1992 標準サポート**　SQL FLAGGER では、SQL:1992 (全レベル) はサポートされなくなりました。
- **dbinit -e オプションの廃止**　データベースの作成時に単純暗号化を指定するための dbinit -e オプションは廃止される予定です。単純暗号化を指定するには -ea オプション (実際には -ea simple) を使用してください。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』を参照してください。
- **SADbType.oldbit データ型の削除**　SADbType.oldbit 列挙定数が SQL Anywhere .NET プロバイダーから削除されました。
- **-gx サーバーオプションの廃止**　Windows デスクトッププラットフォームでは、データベースサーバースケジューラが CPU キャッシュを使用できるように要求の類似性の維持を試みるようになりました。その結果、可能なかぎり 1 CPU 上で要求が実行されます。また、データベースサーバーが使用するオペレーティングシステムスレッドの数を指定する -gx サーバーオプションも廃止される予定です。データベースサーバーではこのオプションは無視されます。
- **CREATE DATABASE 文の CASE 句と ACCENT 句の廃止**　CREATE DATABASE 文の COLLATION 句および NCHAR COLLATION 句を使用して照合を適合化できるようになったので、CREATE DATABASE 文の CASE 句と ACCENT 句は廃止される予定です。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Mobile Link

次の項では、Mobile Link のバージョン 10.0.1 での新機能、動作の変更、廃止される機能について説明します。

新機能

Mobile Link サーバー

- **Oracle 用の新しい ODBC ドライバー**　iAnywhere Solutions 10 - Oracle という名前の Oracle 用の ODBC ドライバーが含まれるようになりました。このドライバーは、Mobile Link アプリケーション用にカスタマイズされています。

  「Oracle 用の新しい ODBC ドライバー」233 ページを参照してください。
- **再構築された暗号化レイヤー**　Mobile Link の暗号化レイヤーが再構築されて改善されました。この変更は透過的であるため、アプリケーションに変更を加える必要はありません。
- **Mobile Link サーバーログファイルビューアー**　新しいウィンドウが追加され、Mobile Link のログファイルを表示することができるようになりました。ログファイルビューアーでは、ロ

グに記録された情報をフィルターしたり概要や統計値を表示したりといった、強化された機能が提供されます。

「Mobile Link サーバーログファイルビューアー」『Mobile Link サーバー管理』を参照してください。

- **メモリの使用量の向上** Windows では、必要に応じて、Mobile Link サーバー (32 ビットプロセス) がより多くのメモリを使用するようになりました。以前のバージョンでは、2 GB 未満に制限されていましたが、必要な場合に使用可能なメモリがあれば、今までよりかなり多いメモリを使用できるようになりました。サーバーのメモリキャッシュの最大サイズを設定するには、mlsrv10 -cm オプションを使用します。メモリを多く使用した場合はディスクへのページングが減り、パフォーマンスが向上します。

Sybase Central の Mobile Link プラグイン

- **モデルモードでの新機能**
  - **テーブルマッピング** テーブルとカラムのマッピングを作成および変更するのが簡単になりました。テーブルの [マッピング方向] カラムのドロップダウンリストから選択することで、テーブルマッピングを有効にできます。同様に、[カラムマッピング] タブで、カラムをマッピングするかどうかを指定できるようになりました。テーブルマッピング新規作成ウィザードは削除されました。

    「テーブルマッピングとカラムマッピングの変更」『Mobile Link クイックスタート』を参照してください。

  - **新しいリモートテーブルの作成** 統合データベーススキーマに基づいて新しいリモートテーブルを作成するのが簡単になりました。[新しいリモートテーブル] ウィンドウを使用して、統合データベースのテーブルと同じ名前とカラムを持つ新しいリモートテーブルを作成できます。このウィンドウを使用して、リモートデータベーステーブルをモデルの統合テーブルにマッピングすることもできます。

  - **リモートデータベースのテーブルとカラムの削除** [マッピング] タブで、リモートデータベースのテーブルとカラムを削除できるようになりました。テーブルまたはカラムを選択し、[編集] » [削除] を選択することで、モデルのリモートデータベーススキーマからリモートテーブルまたはカラムを削除できます。

Mobile Link クライアント

- **簡単になった Windows CE 上のネットワーク名の指定方法** default_internet や default_work キーワードを network_name プロトコルオプションで名前として指定することで、デフォルト設定が使用される名前になるようにすることができるようになりました。

  「network_name」『Mobile Link クライアント管理』を参照してください。

- **delete_old_logs の機能の強化** データベースオプション delete_old_logs で、日数を指定できるようになりました。作成されたログは、この期間を過ぎると削除されます。

  「delete_old_logs オプション [SQL Remote]」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しいフック**　新しいフック sp_hook_dbmlsync_set_ml_connect_info を使用すると、dbmlsync が Mobile Link サーバーへの接続を試みる直前に、ネットワークプロトコルとネットワークプロトコルオプションを設定することができます。

  「sp_hook_dbmlsync_set_ml_connect_info」『Mobile Link クライアント管理』を参照してください。

セキュリティ

トランスポートレイヤーセキュリティを管理するための、createcert と viewcert という名前の 2 つの新しいユーティリティが追加されました。次の項を参照してください。

- 「トランスポートレイヤーセキュリティ」234 ページ
- 「証明書ユーティリティ」『SQL Anywhere サーバー データベース管理』

動作の変更と廃止予定機能

次に、バージョン 10.0.1 で導入された Mobile Link の変更を示します。

Sybase Central の Mobile Link プラグイン

- **テーブルマッピング追加ウィザードの削除**　リモートデータベーススキーマに対して新しいテーブルを追加するための新機能がモデルモードに追加されました。

  「テーブルマッピングとカラムマッピングの変更」『Mobile Link クイックスタート』を参照してください。

- **デフォルトの dbmlsync バッチファイルの改善**　Mobile Link モデルを展開するときは、*model-name_dbmlsync.bat* という名前のファイルを作成します。以前のバージョンでは、このファイルのデフォルトの dbmlsync コマンドに -qc オプションが含まれており、このオプションによって同期後に dbmlsync ウィンドウが閉じられていました。そのため、同期が成功したかどうかを判断するのが難しくなっていました。今回のバージョンでは、デフォルトの dbmlsync コマンドラインから -qc オプションが削除されました。

セキュリティ

TLS ユーティリティの readcert、gencert、reqtool は廃止される予定です。createcert と viewcert という名前のユーティリティに置き換えられました。「証明書ユーティリティ」『SQL Anywhere サーバー データベース管理』を参照してください。

QAnywhere

次の項では、QAnywhere のバージョン 10.0.1 での新機能、動作の変更、廃止される機能について説明します。

新機能

次に、バージョン 10.0.1 で導入された QAnywhere の追加機能を示します。

- **動的アドレス指定**　QAnywhere Agent は、有効なネットワークを検出して、再起動しなくても Mobile Link サーバーの通信プロトコルとアドレスを自動的に調整できるようになりました。

  「-xd qaagent オプション」『QAnywhere』を参照してください。

- **最大ダウンロードサイズ**　メッセージをダウンロードする際の最大サイズを設定できるようになりました。

  「-idl qaagent オプション」『QAnywhere』と、「事前に定義されたクライアントメッセージストアプロパティ」『QAnywhere』の ias_MaxDownloadSize を参照してください。

- **QAnywhere サーバーログファイルビューアー**　新しいビューアーが追加され、QAnywhere サーバーのログファイルを表示することができるようになりました。ログファイルビューアーでは、ログに記録された情報をフィルターしたり概要や統計値を表示したりといった、強化された機能が提供されます。

  「QAnywhere サーバーのログ」『QAnywhere』を参照してください。

クライアント API の強化

- **.NET API でのメッセージリスナー処理中の例外処理機能**　ExceptionListener デリゲートが .NET API に追加されました。この機能は Java API にすでに存在します。

  次の項を参照してください。

  - ExceptionListener デリゲート [QAnywhere .NET]『QAnywhere』
  - ExceptionListener2 デリゲート [QAnywhere .NET]『QAnywhere』

- **メッセージの所有 QAManager をリスナーに渡す機能**　QAManagerBase API をリスナー内部から呼び出すのに便利な新しいインターフェイスが、.NET と Java の API に追加されました。これは、たとえばメッセージの受信確認時に役立ちます。新しいインターフェイスでは、QAManagerBase をリスナーに渡すために、QAManagerBase のグローバルインスタンスを参照したり、別のコーディング方法を使用したりする必要がありません。

  次の項を参照してください。

  - MessageListener2 デリゲート [QAnywhere .NET]『QAnywhere』
  - QAMessageListener2 インターフェイス [QAnywhere Java]『QAnywhere』

動作の変更と廃止予定機能

次に、バージョン 10.0.1 で導入された QAnywhere の変更を示します。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **クライアントメッセージストアのトランザクションログ**　デフォルトでは、クライアントメッセージストアのトランザクションログの内容は、チェックポイントで削除されるようになりました。

  「-m dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

  qaagent -c オプションで StartLine パラメーターを指定すると、この動作を変更できます。

  「-c qaagent オプション」『QAnywhere』を参照してください。

SQL Remote

次の項では、SQL Remote のバージョン 10.0.1 での動作の変更と廃止される機能について説明します。

動作の変更と廃止予定機能

次に、バージョン 10.0.1 で導入された SQL Remote の変更を示します。

- **VIM と MAPI の廃止**　VIM および MAPI メッセージシステムのサポートは、このリリースで廃止される予定です。file、ftp、SMTP のメッセージタイプは引き続きサポートされます。「SQL Remote メッセージシステム」『SQL Remote』を参照してください。

Ultra Light

次の項では、Ultra Light のバージョン 10.0.1 での新機能、動作の変更、廃止される機能について説明します。

新機能

次に、バージョン 10.0.1 で導入された Ultra Light の追加機能を示します。

- **新しいプラットフォームとデバイス**　このリリースでは Windows Vista がサポートされています。

- **SQL パフォーマンスの改善**　これまで Ultra Light では、クエリに既存のインデックスを使用することに利点がないと Ultra Light クエリオプティマイザーが判断した場合に、プライマリキーインデックスがデフォルトで使用されていました。

  このバージョンでは、プライマリキーを使用するのではなく、データベースページから直接ローにアクセスします。これによりローはプライマリキーインデックスの順でなくなるため、これまでのリリースとは異なる順序でクエリの結果が返されることがあります。データの順序が重要である場合、クエリで ORDER BY 句を使用してください。「ダイレクトページ

スキャン」『Ultra Light データベース管理とリファレンス』と「インデックススキャンの使用」『Ultra Light データベース管理とリファレンス』を参照してください。

- **SQL によるデータベースプロパティとオプションのアクセシビリティ**　以前のバージョンでは、各 Ultra Light API のメソッドを使用した場合だけ、データベースプロパティやオプションにアクセスすることが可能でした。今回のバージョンでは、Ultra Light SQL に次の文と関数が導入されたため、SQL (Interactive SQL を含む) を使用してプロパティやオプションを設定したり取得したりすることができるようになりました。

  ○SET OPTION 文。「SET OPTION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

  ○DB_PROPERTY 関数。「DB_PROPERTY 関数 [システム]」『Ultra Light データベース管理とリファレンス』を参照してください。

- **同時接続数の増加**　Ultra Light では、より多くの同時接続数をサポートするようになりました。データベース 1 つあたりの接続数は 14 のままです。しかし、全体の同時データベース接続数は次のように増えました。

  ○Palm OS と Symbian OS の場合はデータベース数が 8、同時接続数が 16

  ○その他のすべてのプラットフォームの場合はデータベース数が 32、同時接続数が 64

  注意
  エンジンへの接続に使用できる SQLCA の合計数は 31 です。データベースマネージャーあたり 1 つの SQLCA と接続あたり 1 つの SQLCA を使用するコンポーネントがあることをふまえて、この値を使用してください。つまり、Ultra Light.NET API を使用するアプリケーションでは、実際の接続制限は合計で 30 に減ってしまいます。

- **コミットフラッシュ操作**　これまでは、COMMIT 文または API 呼び出しによって実行されたコミット操作は、Ultra Light がトランザクションを安全に記憶領域にフラッシュした後でのみ完了していました。このリリースでは、これらの動作を設定したり、独立した操作として論理的に分けたりすることができるようになりました。

  ○論理コミットでは、アプリケーションでトランザクションをロールバックできるようになりました。

  ○チェックポイントでは、エラー後のリカバリポイントを使用できるようになりました。これにより、メモリにフラッシュ済みの、最後にコミットされたトランザクションまでリカバリすることができます。

  ただし、オートコミット機能を使用する Ultra Light アプリケーションのパフォーマンスを強化することもできます。特にグループコミットフラッシュを使用する場合に有効です。「Ultra Light および Ultra Light Java Edition データベースのバックアップとリカバリ」『Ultra Light データベース管理とリファレンス』と「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」『Ultra Light データベース管理とリファレンス』を参照してください。

  この新しい動作は、次の機能でサポートされます。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○CHECKPOINT 文。「CHECKPOINT 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

Ultra Light Embedded SQL API と C++ API コンポーネントでの Checkpoint メソッド。その他の言語では、代わりに CHECKPOINT 文を使用する必要があります。次の項を参照してください。

- ULCheckpoint メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
- ULConnection.Checkpoint メソッド [Ultra Light C++]『Ultra Light C/C++ プログラミング』

○COMMIT_FLUSH 接続パラメーター。「Ultra Light COMMIT_FLUSH 接続パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

○commit_flush_timeout データベースオプションと commit_flush_count データベースオプション。「Ultra Light commit_flush_timeout オプション [テンポラリ]」『Ultra Light データベース管理とリファレンス』と「Ultra Light commit_flush_count オプション [テンポラリ]」『Ultra Light データベース管理とリファレンス』を参照してください。

動作の変更と廃止予定機能

次に、バージョン 10.0.1 で導入された Ultra Light の変更点を示します。

- **ulafreg**　ulafreg は変更され、標準出力は利用できなくなりました。このユーティリティは、これまでと同様に、必要なオプションを指定してコマンドラインから実行します。しかし、クリップボードに出力をコピーするには、これからは **[編集]** － **[コピー]** をクリックする必要があります。任意のテキストエディターを使用して、これをファイルに保存できます。

- **Windows CE での FIPS 認定のサポート**　Windows CE デバイスで FIPS 認定のアルゴリズムをサポートするために、実行プログラム *%SQLANY10%¥ultralite¥ce¥arm¥fips¥setup.exe* を実行する必要がなくなりました。これからは、*%SQLANY10%¥ultralite¥ce¥arm¥sbgse2.dll* ファイルを展開するだけで済みます。

製品全体の機能

次の項では、SQL Anywhere バージョン 10.0.1 のすべてのコンポーネントに影響する新機能、動作の変更、廃止される機能について説明します。

新機能

次に、バージョン 10.0.1 で導入された製品全体の追加機能を示します。

Oracle 用の新しい ODBC ドライバー

iAnywhere Solutions 10 - Oracle と呼ばれる Oracle 用のネイティブ ODBC ドライバーが用意されました。

以前のバージョンの iAnywhere では、サードパーティ製の Oracle ドライバーをリブランドしていました。今回のバージョンでは、iAnywhere には SQL Anywhere アプリケーションで使用するための独自の Oracle ODBC ドライバーが用意されています。このドライバーを使用すると、バグフィックスの速度が向上し、国際文字データの処理機能が改善されます。Oracle 統合データベースを使用して Mobile Link を配備する場合やリモートデータアクセスを使用して Oracle に接続する場合は、この新しいドライバーに切り替えることを強くおすすめします。

「iAnywhere Solutions 12 - Oracle ODBC ドライバー」『Mobile Link サーバー管理』を参照してください。

トランスポートレイヤーセキュリティ

- **新しい証明書ユーティリティ**　新しい 2 つのユーティリティ createcert と viewcert を使用すると、セキュリティ証明書を作成、修正、表示できます。これまでは、この目的で gencert、reqtool、readcert の各ユーティリティを使用していましたが、これらのユーティリティは廃止される予定です。

  viewcert を使用すると、いくつかの種類の PKI オブジェクトを表示できます。これまでは証明書を表示することしかできませんでした。viewcert では、PEM オブジェクトと DER オブジェクトの両方を表示することもできます。これまでは PEM オブジェクトだけを表示することしかできませんでした。viewcert を使用して PEM と DER の間で変換したり、パスワードの暗号化や復号化を実行したりすることもできます。

  createcert は、以前の gencert ユーティリティと reqtool ユーティリティの機能を組み合わせ、さらに新しい機能を追加したユーティリティです。ECC 曲線を作成する場合、これまでは sect163k1 を使用する必要がありましたが、曲線を選択できるようになりました。使用できるキーのサイズは、512 ～ 2048 ビットの制限がありましたが、512 ～ 16384 ビットになりました。GUID シリアル番号は、デフォルトはありませんでしたが、デフォルトが用意されました。必要に応じて、別の証明書に署名できる証明書を作成することもできるようになりました。また、証明書のプライベートキーの使用方法を決定する詳細なオプションを指定できます。最後に、すべてのプライベートキーにはパスワードが必要でしたが、暗号化されていないプライベートキーを使用できるようになりました。

  1

- **Windows CE でサポートされる FIPS 認定の Certicom Security Builder のアップグレード**　SQL Anywhere 製品では、2 つの FIPS 認定モジュールのうちの 1 つを使用できますが、どちらも Certicom 製です。

  ○Palm OS の場合、これまでと同様に Security Builder Government Services Edition v1.0.1 を使用する必要があります。

  ○Windows CE では、Security Builder Government Services Edition v2.0.0 を使用する必要があります。これらのライブラリは、Windows Mobile で使用するように署名できるためです。

「FIPS 認定の暗号化テクノロジ」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

動作の変更

次に、バージョン 10.0.1 で導入された製品全体の変更を示します。

トランスポートレイヤーセキュリティ

- **gencert、readcert、reqtool の廃止**　セキュリティユーティリティの gencert、reqtool、readcert は廃止される予定です。gencert と reqtool は createcert で置き換えられました。readcert は viewcert で置き換えられました。

  「証明書ユーティリティ」『SQL Anywhere サーバー データベース管理』を参照してください。

ライセンス

- ***.lic* ファイルに格納されるようになったサーバーのライセンス情報**　以前のリリースでは、SQL Anywhere パーソナルデータベースサーバー、SQL Anywhere ネットワークデータベースサーバー、Mobile Link サーバーのライセンス情報は、サーバーの実行プログラムに格納されていました。この情報が *.lic* ファイルに格納されるようになりました。このファイルは、サーバーの実行プログラムと同じフォルダーにあります。実行プログラムで *.lic* が見つからないと、プログラムは起動しません。

  この変更によって、a_dblic_info 構造体の exename メンバーが実行プログラムまたはライセンスファイル名を指定できるようになりました。

  次の項を参照してください。

  - 「サーバーライセンス取得ユーティリティ (dblic)」『SQL Anywhere サーバー データベース管理』
  - 「データベースサーバーの配備」『SQL Anywhere サーバー プログラミング』
  - 「Mobile Link サーバーの配備」『Mobile Link サーバー管理』
  - a_dblic_info 構造体 [データベースツール]『SQL Anywhere サーバー プログラミング』

Windows Vista サポートの問題

SQL Anywhere バージョン 10.0.1 では、Windows Vista オペレーティングシステムをサポートします。Vista で SQL Anywhere ソフトウェアを実行する場合には、次のような問題があります。

- **Windows Vista のセキュリティ**　Windows Vista には、新しいセキュリティモデルが組み込まれています。ユーザーアカウント制御 (UAC) はデフォルトで有効に設定され、ファイルに書き込み可能とされるプログラムの動作に影響する可能性があります (特に、コンピューターが複数のユーザーをサポートしている場合)。ファイルとディレクトリを作成した場所と作成方法によって、あるユーザーが作成したファイルを、他のユーザーが読み込んだり、書き込むことが許可されなくなる場合があります。SQL Anywhere をデフォルトのディレクトリにインストールした場合は、複数のユーザーに読み込み／書き込みアクセスを許可する必要があるファイルおよびディレクトリが適切に設定されます。

- **SQL Anywhere 昇格操作エージェント**　Vista では、ユーザーアカウント制御がアクティブな状態で実行する場合に、特定のアクションで権限の昇格が必要になります。SQL Anywhere では、次のプログラムで昇格が必要になることがあります。*dbdsn.exe*、*dbelevate10.exe*、

dblic.exe、*dbsvc.exe*、*installULNet.exe*、*mlasinst.exe*、*SetupVSPackage.exe*、*ulcond10.exe*、*ulafreg.exe*。

次の DLL では、登録または登録の解除時に昇格が必要です。*dbctrs10.dll*、*dbodbc10.dll*、*dboledb10.dll*、*dboledba10.dll*。

ユーザーアカウント制御がアクティブな Vista システムでは、SQL Anywhere の昇格操作エージェントに対して昇格を確認するメッセージが表示されることがあります。このメッセージは、識別されたプログラムの実行を継続するかどうかを確認したり (管理者としてログオンしている場合)、管理者のクレデンシャルを提供するように求める (管理者以外でログオンしている場合) ため、Vista のユーザーアカウント制御システムによって発行されるものです。

- **配備環境の変更**　プログラム *dbelevate10.exe* は、昇格された権限が必要な操作を実行するために SQL Anywhere コンポーネントによって内部的に使用されます。この実行プログラムは、SQL Anywhere の配備環境に含まれている必要があります。

- **ActiveSync のサポート**　Microsoft ActiveSync ユーティリティは、Vista ではサポートされていません。これは、Windows Mobile Device Center で置き換えられました。SQL Anywhere ActiveSync プロバイダーインストールユーティリティを Windows Mobile Device Center で使用できます。

- **署名された SQL Anywhere 実行プログラム**　Vista での SQL Anywhere 実行プログラムは、iAnywhere Solutions, Inc. によって署名されています。

- **新しいライセンスファイル**　10.0.1 のインストールには、SQL Anywhere 用の新しいライセンスファイルを作成する手順が含まれます。既存のインストールからのライセンス情報は、実行ファイル内の以前置かれていた場所から抽出され、新しい場所 (実行プログラムと同じディレクトリにある *dbsrv10.lic*、*dbeng10.lic*、*mlsrv10.lic* ファイル) に移動されます。

  「サーバーライセンス取得ユーティリティ (dblic)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サンプル**　サンプルで、1 つ以上のスペースを含む SQL Anywhere インストールパス名が正しく処理されるようになりました。

- **Windows サービス**　Vista に準拠したサービスでは、デスクトップとの対話が許可されていません。Windows Vista では、(サービス定義で **[デスクトップとの対話をサービスに許可]** が有効になっている場合でも) SQL Anywhere サービスはデスクトップと対話しません。SQL Anywhere データベースサーバーは、dbconsole ユーティリティを使用するか Sybase Central からモニタリングできます。

  Sybase Central を Windows Vista で実行している場合、サービスがデスクトップと対話できるオプションは無効になります。

- **PowerDesigner、InfoMaker、DataWindow .NET**　SQL Anywhere に含まれる PowerDesigner、InfoMaker、DataWindow .NET の各コンポーネントは、Windows Vista では正式にサポートされていません。そのため、Vista 環境でこれらのコンポーネントを実行すると、問題が発生する可能性があります。Vista 環境での実行方法、およびこれらの製品の Vista 対応版の入手方法については、それぞれの製品マニュアルを参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 10.0.0 の新機能

SQL Anywhere のバージョン 10 より前における新機能や動作の変更については、ftp://ftp2.ianywhere.jp/public/tech/dbwnja10.pdf を参照してください。

注意
廃止予定機能のリストはあくまでも予定であって完全なものとは限らず、変更される可能性があります。

SQL Anywhere

- 「新機能」238 ページ
- 「動作の変更」274 ページ
- 「廃止予定機能とサポート終了機能」300 ページ

Mobile Link

- 「新機能」308 ページ
- 「動作の変更と廃止予定機能」322 ページ

QAnywhere

- 「新機能」332 ページ
- 「動作の変更と廃止予定機能」335 ページ

SQL Remote

- 「新機能」337 ページ
- 「動作の変更と廃止予定機能」337 ページ

Ultra Light

- 「新機能」338 ページ
- 「動作の変更と廃止予定機能」351 ページ

Sybase Central と Interactive SQL

- 「新機能」354 ページ
- 「動作の変更と廃止予定機能」356 ページ

マニュアルの強化

- 「マニュアルの強化」359 ページ

SQL Anywhere

次の項では、SQL Anywhere のバージョン 10.0.0 の新機能、動作の変更、廃止される機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された SQL Anywhere データベースとデータベースサーバーの追加機能を示します。

主な機能

- **パフォーマンスを向上させる並列処理のサポート**　データベースサーバーで、単一のクエリを処理するのに複数のプロセッサーを使用できるようになりました。クエリ内並列処理は、同時実行されるクエリ数が使用可能なプロセッサー数よりも少ない場合に便利です。「高度 : クエリ実行時の並列処理」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **データベースのミラーリングのサポート**　SQL Anywhere で、データベースの可用性を高めるためのメカニズムである、データベースのミラーリングがサポートされるようになりました。データベースのミラーリングでは、別々のコンピューターで実行され、同期モードか非同期モードで相互通信する 2 ～ 3 のデータベースサーバーを使用します。「データベースミラーリング」『SQL Anywhere サーバー データベース管理』を参照してください。

  データベースのミラーリングをサポートするために、次の機能が追加されています。

  ○「synchronize_mirror_on_commit オプション」『SQL Anywhere サーバー データベース管理』

  ○データベースサーバーの代替サーバー名。「-sn dbsrv12 オプション」『SQL Anywhere サーバー データベース管理』と「START DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○ServerName プロパティ

  ○AlternateServerName プロパティ

  ○RetryConnectionTimeout プロパティ

  ○ALTER DATABASE *dbname* FORCE START。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○MirrorServerDisconnect システムイベントと MirrorFailover システムイベント。「システムイベント」『SQL Anywhere サーバー データベース管理』を参照してください。

  ○「-xf dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』

  ○「-xp dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』

  ○SQL Anywhere SNMP Extension Agent 用の新しいトラップ。「トラップ」『SQL Anywhere サーバー データベース管理』を参照してください。

  ○EVENT_PARAMETER 関数に追加された MirrorServerName パラメーター。「EVENT_PARAMETER 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

データベースのミラーリングに加え、SQL Anywhere では、Veritas Cluster Server エージェントがデータベース用 (SADatabase エージェント) およびデータベースサーバー用 (SAServer エージェント) に提供されるようになりました。「SQL Anywhere Veritas Cluster Server エージェント」『SQL Anywhere サーバー データベース管理』を参照してください。

- **スナップショットアイソレーションのサポート**　スナップショットアイソレーションを使用すると、ユーザーがデータを変更する間、データベースでは元のデータのコピーが保持されるので、他のユーザーも元のデータを読むことができます。スナップショットアイソレーションはユーザーに対して完全に透過的であり、デッドロックやロック競合の発生を抑えるのに役立ちます。「スナップショットアイソレーション」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  スナップショットアイソレーションをサポートするために、次の機能が追加または強化されました。

  - 「allow_snapshot_isolation オプション」『SQL Anywhere サーバー データベース管理』
  - 「isolation_level オプション」『SQL Anywhere サーバー データベース管理』
  - 「sa_snapshots システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
  - 「sa_transactions システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
  - LockCount、SnapshotCount、SnapshotIsolationState、VersionStorePages の各データベースプロパティ
  - allow_snapshot_isolation、LockCount、SnapshotCount の各接続プロパティ
  - バージョンストアページ (パフォーマンスモニターの統計)
  - 「SET 文 [T-SQL]」『SQL Anywhere サーバー SQL リファレンス』
  - 「OPEN 文 [ESQL] [SP]」『SQL Anywhere サーバー SQL リファレンス』
  - 「ValuePtr パラメーター」『SQL Anywhere サーバー SQL の使用法』

- **アプリケーションプロファイリングと診断トレーシングのサポート**　ストアドプロシージャープロファイリングや要求ロギングなどの既存のアプリケーションプロファイリング機能は、単一の対話型インターフェイスである Sybase Central 用 SQL Anywhere プラグインに統合されました。Sybase Central からアプリケーションをプロファイリングするときは、データベースのパフォーマンスを向上させるためのアドバイスが提供されます。

  Sybase Central でのアプリケーションプロファイリングの詳細については、「アプリケーションプロファイリング」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **マテリアライズドビューのサポート**　データベースのサイズが大きく、頻繁にクエリが行われるために大量のデータで繰り返し集約やジョイン操作が発生するような環境において、パフォーマンスを向上させるために、SQL Anywhere ではマテリアライズドビューがサポートされました。「マテリアライズドビュー」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  データベースサーバーの機能が強化され、クエリの一部に応答するために使用できるマテリアライズドビューを自動的に決定できるようになりました。この決定はコストを基に行われ、クエリが直接参照するベーステーブルを使用する必要はありません。

  マテリアライズドビューの情報を格納するために、2 つの新しいシステムテーブル ISYSMVOPTION と ISYSMVOPTIONNAME が追加されました。

● **NCHAR データのサポート**　SQL Anywhere で NCHAR データ型がサポートされるようになりました。NCHAR データ型は、Unicode 文字データを格納するのに使用されます。「NCHAR データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

NCHAR をサポートするために、次の新しい関数が追加されました。

○ UNISTR 関数 [文字列] :「UNISTR 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
○ CONNECTION_EXTENDED_PROPERTY 関数 [文字列] :「CONNECTION_EXTENDED_PROPERTY 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
○ UNICODE 関数 [文字列] :「UNICODE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
○ NCHAR 関数 [文字列] :「NCHAR 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
○ TO_CHAR 関数 [文字列] :「TO_CHAR 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
○ TO_NCHAR 関数 [文字列] :「TO_NCHAR 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

NCHAR データ型をサポートするために、次の関数 SOFTKEY と COMPARE に新しいパラメーターが追加されました。

○ SORTKEY 関数 [文字列] :「SORTKEY 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
○ COMPARE 関数 [文字列] :「COMPARE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Unicode 照合アルゴリズム (UCA) を使用するときに、マルチバイトの文字セットを正しくソートできるようになりました。

NCHAR データ型をサポートするために、初期化ユーティリティ (dbinit) と Unload (dbunload) ユーティリティにも新しいオプションが追加されました。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』と「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

Unicode のサポートのため、International Components for Unicode (ICU) が使用されるようになりました。「国際言語と文字セット」『SQL Anywhere サーバー データベース管理』を参照してください。

ICU の使用と NCHAR データの処理をサポートするために、次のプロパティが変更されました。

○新しいデータベースと接続の拡張プロパティ NcharCharSet が追加されました。このプロパティは、データベースまたは接続で使用されている NCHAR 文字セットを返します。

○新しいデータベースプロパティ AccentSensitive が追加されました。このプロパティは、アクセントを区別する機能のステータスを返します。

○CharSet データベースと接続のプロパティが、拡張プロパティになりました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』と「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **内部パフォーマンスの強化**　データベースサーバーのパフォーマンスを向上させるために、仮想マシンテクノロジを使用して、SQL 式の表現と評価を再構成するようになりました。これにより、スループットが大幅に向上します。

- **ビューの依存性のサポート**　ビューの依存性の情報がカタログに格納されるようになりました。カタログは特に、データベースの各ビューが依存するビュー、テーブル、カラムを追跡します。ビューが依存するオブジェクトを変更すると、ビュー定義が不適切な結果を返す状態のままにならないように、データベースサーバーは自動的に追加処理を実行します。「ビューの依存性」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  システムオブジェクトとそれらの依存性の情報を格納するために、2 つの新しいシステムテーブル ISYSDEPENDENCY と ISYSOBJECT が追加されました。

- **チェックポイントアルゴリズムの向上**　データベースサーバーは、チェックポイントを開始し、チェックポイントの発生中にその他の操作を実行できるようになりました。以前は、チェックポイントが発生すると、すべてのアクティビティが停止していました。チェックポイントがすでに進行中の場合、新しくチェックポイントを開始する ALTER TABLE や CREATE INDEX などの操作は、現在のチェックポイントが完了するまで待機する必要があります。「チェックポイントログ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ロックの強化**　ロックが次のように強化されています。

  - **ロックのクラス**　SQL Anywhere で、スキーマロック、テーブルロック、ローロック、位置ロックの 4 つの固有のクラスが使用できるようになりました。ロックの問題をより厳密に分析できるように、各トランザクションが保持するロックの種類を明確に記述するように sa_locks システムプロシージャーが変更されました。「ロックの仕組み」『SQL Anywhere サーバー SQL の使用法』と「sa_locks システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **意図的ロックのサポート**　新しい種類のロックである意図的ロックが、テーブルロックとローロックに導入されました。意図的ロックは、アプリケーションがテーブルまたはそのテーブル内のローセットの更新意図を通知するために使用されます。アプリケーションで SELECT FOR UPDATE または FETCH FOR UPDATE 文 (または各種プログラミングインターフェイスで同様の構成) を使用するときに、意図的ロックが取得されるようになりました。意図的ロックは、その他の意図的ロックや書き込みロックをブロックしますが、読み込みロックはブロックしません。このため、明示的な同時制御メカニズムとしてロックを使用するアプリケーションで、高度な同時実行性が実現されます。「カーソルの使い方」『SQL Anywhere サーバー プログラミング』と「意図的ロック」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  - **一部の状況におけるキー範囲ロックの省略**　インデックス管理アルゴリズムが変更され、データベースサーバーがキーの範囲ではなく個別のインデックスエントリに、書き込みロックを設定できるようになりました。これにより、同時実行性が向上し、さまざまな環境で同時に発生する INSERT 操作による不要なブロックを避けることができます。「ロックの仕組み」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **インデックス処理の強化**　インデックス処理には、次のような強化が行われています。
  - **新しいインデックスの実装**　SQL Anywhere の以前のリリースには、2 種類のインデックス処理が実装されていました。これらは宣言されたインデックスカラムのサイズに基づいて自動的に選択されていました。SQL Anywhere 10 で、圧縮 B ツリーインデックスの新しい実装が全体に使用されるようになり、以前の B ツリーインデックス処理テクノロジが廃止されました。新しいインデックスには、ローの値とは全く別の、インデックスエントリのインデックスキー値が圧縮形式で格納されます。スナップショットアイソレーションをサポートするにはこの機能が必要です。
  - **スナップショットアイソレーションのサポート**　以前の SQL Anywhere リリースでは、UPDATE または DELETE 文によりインデックスエントリがすぐに削除されていました。スナップショットアイソレーションをサポートするために、異なるインデックスキー値を持つ同じ論理ローを指すような複数のインデックスエントリが存在する可能性があります。これらの複数のインデックスエントリは、データベースサーバーによって管理されるため、どの接続でも、任意のローに対してエントリの 1 つしか確認できません。サーバー内のデーモンは、定期的にこれらの余分なインデックスエントリが (トランザクション COMMIT または ROLLBACK で) 不要になった時点で物理的に削除します。コミットされていない DELETE ではインデックスエントリを保持することにより、SQL Anywhere の同時実行性制御メカニズムのセマンティック一貫性も向上します。「スナップショットアイソレーション」『SQL Anywhere サーバー SQL の使用法』を参照してください。
- **BLOB 記憶域制御とパフォーマンスの向上**　テーブルのローに (インラインで) 格納される BLOB 値の量を制御できるようになりました。また、BLOB 値をインデックス処理するかどうかも制御できます。これらの強化により BLOB の検索とアクセスが向上します。これらの強化機能を使用するには、CREATE TABLE と ALTER TABLE 文の 3 つの新しい句 INLINE、PREFIX、[NO] INDEX を使用します。BLOB 値は同じテーブルの行同士または行内で共有できるようになりました。重複した BLOB 値を格納する必要がなくなるため、記憶域の要件が抑えられます。「BLOB の考慮事項」『SQL Anywhere サーバー データベース管理』、「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』、「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **カラム圧縮のサポート**　テーブルの個別カラムを圧縮できるようになりました。圧縮には deflate 圧縮アルゴリズムが使用されます。これは、COMPRESS 関数で使用される圧縮方式と同じであり、Windows の *.zip* ファイルで使用されるアルゴリズムでもあります。「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **テーブルの暗号化のサポート**　データをセキュリティ保護するためにデータベース全体を暗号化するのではなく、データベースの個別のテーブルを暗号化できるようになりました。テーブルの暗号化は、データベースを初期化するとき有効にする必要があります。「テーブル暗号化」『SQL Anywhere サーバー データベース管理』を参照してください。

データベース接続

- **一重引用符または二重引用符のサポート**　接続文字列内の値を、一重引用符または二重引用符で囲むことができるようになりました。これによって、接続文字列の値でスペースやセミコロンなどの文字を使用できるようになりました。「接続文字列として渡される接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **接続文字列でのブール値としての T、Y、F、N の使用**　接続文字列で接続パラメーターとプロトコルのオプションを指定するときに、true を示すために T または Y を、false を示すために F または N を指定できるようになりました。「接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **一部の接続文字列とプロトコルのオプションでの k、m、g サフィックスの付いた値の使用**　次の接続パラメーターとプロトコルのオプションで、キロバイト、メガバイト、ギガバイトを表すサフィックスとして k、m、g を使用できるようになりました。

  - CommBufferSize (CBSIZE) 接続パラメーター：「CommBufferSize (CBSIZE) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。
  - Compression Threshold (COMPTH) 接続パラメーター:「CompressionThreshold (COMPTH) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。
  - PrefetchBuffer (PBUF) 接続パラメーター：「PrefetchBuffer (PBUF) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。
  - LogMaxSize (LSIZE) プロトコルオプション：「LogMaxSize (LSIZE) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  - MaxRequestSize (MAXSIZE) プロトコルオプション：「MaxRequestSize (MAXSIZE) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  - ReceiveBufferSize (RCVBUFSZ) プロトコルオプション：「ReceiveBufferSize (RCVBUFSZ) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  - SendBufferSize (SNDBUFSZ) プロトコルオプション：「SendBufferSize (SNDBUFSZ) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **AppInfo が Windows クライアントの IP アドレスを返す**　以前のリリースでは、AppInfo 接続パラメーターは、UNIX と NetWare クライアントのクライアントコンピューターの IP アドレスを返すだけでした。Windows クライアントの IP アドレスも返すようになりました。「AppInfo (APP) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **個々の接続の監査**　conn_auditing テンポラリデータベースオプションがログインプロシージャーで設定されているときは、特定の接続の監査を有効または無効にできます。データベースの監査ステータスの情報を取得できるように、auditing データベースプロパティが追加されました。「conn_auditing オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **RetryConnectionTimeout 接続パラメーター**　RetryConnectionTimeout (RetryConnTO) 接続パラメーターは、サーバーが見つからない場合に、指定された期間、クライアントライブラリに対して接続の試行をリトライするように通知します。「RetryConnectionTimeout (RetryConnTO) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **IPv6 のサポート**　IPv6 は Windows、Linux、Mac OS X、Solaris、AIX、HP-UX でサポートされるようになりました。これらのオペレーティングシステムを実行しているサーバーで、使用可能なすべての IPv4 と IPv6 のアドレスを受信するようになりました。また、クライアントやサーバーで IP アドレスを指定できる箇所で (HOST=、MYIP=、BROADCAST= の各 TCP プロトコルオプションなど)、IPv6 アドレスを指定できるようになりました。「SQL Anywhere での IPv6 サポート」『SQL Anywhere サーバー データベース管理』を参照してください。

- **LDAP 登録の新しいパラメーター**　データベースサーバーが Active Directory サーバーである場合に、read_authdn と read_password パラメーターを使用して、LDAP にデータベースサーバーを登録できるようになりました。「LDAP サーバーを使用した接続」『SQL Anywhere サーバー データベース管理』を参照してください。

バックアップとリカバリ

- **重要なデータベースページでのチェックサム自動計算**　データベースでチェックサムが有効であるかどうかに関係なく、データベースサーバーは、重要なデータベースページのチェックサムを記録します。その結果、データベースでチェックサムが有効でない場合でも、データベースを検証するとチェックサム違反の警告が表示されることがあります。また、破壊された重要なページにアクセスしようとすると、致命的なエラーが発生してデータベースサーバーは停止します。「チェックサムの検証」『SQL Anywhere サーバー データベース管理』を参照してください。

- **リカバリの開始時に複数のトランザクションログを適用**　デフォルトでは、データベースのリカバリ時にトランザクションログを正しい順序で個々に適用する必要があります。データベースサーバーの開始時に新しいリカバリオプション -ad、-ar、-as を指定した場合は、トランザクションログをデータベースに適用する順序を手動で指定する必要はありません。データベースサーバーとデータベースはトランザクションログの適用中も実行しているため、サーバーのキャッシュはウォーミングステータスのままになり、総リカバリ時間を減らすことができます。「-ad dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』、「-ar dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』、「-as dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **並列データベースバックアップのサポート**　SQL Anywhere データベースサーバーで、サーバー側のイメージバックアップで並列バックアップがサポートされるようになりました。並列データベースバックアップでは、物理 I/O を利用して、直列ではなく並列に情報を読み書きすることで、パフォーマンスが向上します。並列バックアップは次のいずれかの方法で実行できます。

  ○「バックアップユーティリティ (dbbackup)」『SQL Anywhere サーバー データベース管理』
  ○「BACKUP 文」『SQL Anywhere サーバー SQL リファレンス』
  ○「db_backup 関数」『SQL Anywhere サーバー プログラミング』

- **最後のバックアップの情報の追跡**　最後のバックアップの情報を格納するために、新しいカラム LAST_BACKUP が ISYSHISTORY システムテーブルに追加されました。

セキュリティ

この項では、セキュリティを向上させるために SQL Anywhere に加えられた強化について説明します。

- **RSA が SQL Anywhere に付属**　RSA 暗号化を使用するためのライセンスを別途購入する必要がなくなりました。「別途ライセンスが必要なコンポーネント」『SQL Anywhere 12 紹介』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **FIPS 認定の暗号化のサポートの強化**　データベースサーバーに、次のような変更がありました。

  ○DLL の名前が *dbfips10.dll* に変更されました。バージョン 9.0 では、*dbrsa9f.dll* でした。

  ○HASH 関数で 2 つの新しいアルゴリズム SHA1_FIPS と SHA256_FIPS が使用できるようになりました。これらは SHA1 アルゴリズムや SHA256 アルゴリズムと同じですが、FIPS 認定の Certicom バージョンです。

  ○-fips サーバーオプションを指定したときに FIPS 認定でないアルゴリズムを HASH 関数に指定すると、データベースサーバーでは SHA1 の代わりに SHA1_FIPS が、SHA256 の代わりに SHA256_FIPS が使用されます。また、MD5 を使用した場合はエラーが返されます (MD5 は FIPS 認定のアルゴリズムではありません)。

  ○-fips オプションを指定した場合は、パスワードハッシュ処理に SHA256_FIPS が使用されます。

  また、-fips オプションと FIPS 認定の機能をより多くのプラットフォームで使用できるようになりました。-fips オプションをサポートするプラットフォームのリストについては、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

- **Kerberos 認証**　SQL Anywhere で、Kerberos 認証がサポートされるようになりました。Kerberos 認証では、ユーザー ID やパスワードを指定しなくても、Kerberos クレデンシャルを使用してデータベースに接続できます。「Kerberos 認証」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい権限の追加**　次の権限が追加されました。

  ○ **バックアップ権限**　ユーザーがバックアップを実行できるようにするために、ユーザー DBA 権限を付与する代わりに、バックアップ権限をユーザーに割り当てることができます。「バックアップ権限」『SQL Anywhere サーバー データベース管理』を参照してください。

  ○ **VALIDATE 権限**　検証操作用の新しい権限 VALIDATE が追加されました。さまざまな VALIDATE 文による操作 (データベース、テーブル、インデックス、チェックサムなどの検証) を実行するには、VALIDATE 権限が必要です。「VALIDATE 権限」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースサーバーのセキュリティ保護機能**　-sf データベースサーバーオプションを使用して、データベースサーバーで実行しているデータベースに対してセキュリティ保護されている (無効である) 機能または機能グループを指定できます。「-sf dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

  -sk サーバーオプションでは、secure_feature_key データベースオプションを使用するときに、無効になっている機能を有効にするためのキーを指定できます。sa_server_option プロシージャーで SecureFeatures プロパティを使用すると、無効になっている機能のセットを変更することもできます。「-sk dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースユーティリティ

- **@filename を複数のユーティリティで再使用可能**　コマンドパラメーターファイルを使用するユーティリティで、そのパラメーターファイルを個々に解析できるようになりました。パラメーターファイル内に配置された簡単な条件ディレクティブを基に解析が行われます。「設定ファイルでの条件付き解析」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データソースユーティリティ (dbdsn) の強化**　dbdsn ユーティリティに次のオプションが追加されています。

  - **-dr**　データソースの作成に使用したコマンドをリストするときは、DRIVER= パラメーターを指定します。これによって、データソースを再作成して、現在のバージョンのソフトウェアに含まれる ODBC ドライバーとはバージョンの異なる ODBC ドライバーを使用できるようになります。

  - **-f**　使用されているシステム情報ファイル (通常は *.odbc.ini*) の名前を表示します。

  - **-ns**　dbdsn に対して、システム情報ファイル (通常 *.odbc.ini*) を検索せずに、既存の環境変数を使用してファイルの場所を判断するように指定します。1 つまたは複数の環境変数によって指定されたファイルが存在せず、ODBC データソースを作成している場合に、この機能が役立ちます。

  - **-pe**　データソースのパスワードフィールドを暗号化します。

  「データソースユーティリティ (dbdsn)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ヒストグラムユーティリティ (dbhist) の強化**　dbhist で作成される Excel 出力ファイル内のシートに対して、Sheet1、Sheet2 などではなく、シートの適用先カラム名を反映した名前が付くようになりました。「ヒストグラムユーティリティ (dbhist)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **情報ユーティリティ (dbinfo) の強化**　-u オプションにマテリアライズドビューの情報が含まれるようになりました。「情報ユーティリティ (dbinfo)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **初期化ユーティリティ (dbinit) の強化**　初期化ユーティリティ (dbinit) で、次の新しいオプションがサポートされるようになりました。

  - **-a**　UCA 文字列比較でアクセントを区別します。

  - **-af**　UCA 文字列比較でフランス語用のアクセント区別ルールを使用します。

  - **-dba**　新しいデータベースで、デフォルト DBA データベースユーザーのユーザー ID やパスワードを変更します。

  - **-dbs**　データベースファイルの初期サイズを指定します。

  - **-ze**　CHAR データ型の文字セットエンコーディングを指定します。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○ **-zn**　NCHAR データ型の照合順を指定します。

「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』を参照してください。

● **Log Transfer Manager (LTM) の強化**　Log Transfer Manager (LTM) ユーティリティ (Replication Agent とも呼ばれる) では、Replication Server 15.0 と Open Server/Open Client 15.0 環境で Replication Agent を使用する場合に、テーブル、カラム、プロシージャー、関数、パラメーターの名前に 128 バイトまでの識別子を使用できるようになりました。以前のバージョンでは、識別子は 30 バイトまでに制限されていました。

dbltm が生成する情報、警告、エラーの各メッセージのタイムスタンプは、明確に定義された ISO 8601 日時フォーマット ({I|W|E} yyyy-mm-dd hh:mm:ss message) を使用するようになりました。

注意
Log Transfer Manager は、SQL Anywhere バージョン 12 の時点でサポートされなくなりました。

● **Ping ユーティリティ (dbping) の強化**　Ping ユーティリティ (dbping) で -s または -st オプションを指定すると、Embedded SQL 接続のパフォーマンスやネットワークのパフォーマンスに付いての情報を取得できます。これらのオプションは、dbping を実行するコンピューターと、データベースサーバーを実行するコンピューターとの間のパフォーマンスに関する統計をレポートします。「Embedded SQL とネットワーク接続パフォーマンスのテスト (dbping)」『SQL Anywhere サーバー データベース管理』を参照してください。

-pd オプションを使用して、プロパティ値の取得元データベースの名前を指定できるようになりました。「Ping ユーティリティ (dbping)」『SQL Anywhere サーバー データベース管理』を参照してください。

● **サーバー列挙ユーティリティ (dblocate) の強化**　サーバー列挙ユーティリティ (dblocate) で、データベースを検索するための新しいオプションがサポートされるようになりました。

○ **-d**　サーバーの名前とアドレス、および各サーバーで実行しているすべてのデータベースのリスト (カンマ区切り) を表示します。

○ **-dn**　サーバーが指定された名前のデータベースを実行している場合にかぎり、そのサーバーの名前とアドレスを表示します。

○ **-dv**　サーバーの名前とアドレス、および各サーバーで実行しているすべてのデータベースのリスト (1 行に 1 つずつ) を表示します。

○ **-p**　指定した TCP/IP ポート番号を使用しているサーバーを表示します。

○ **-s**　指定した名前のサーバーを表示します。

○ **-ss**　指定したサブ文字列を含むサーバーの名前を表示します。

「サーバー列挙ユーティリティ (dblocate)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サービスユーティリティ (dbsvc) の強化**　サービスユーティリティ (dbsvc) では、Log Transfer Manager のサービスを管理できる DBLTM サービスタイプと、Mobile Link Listener ユーティリティのサービスを管理できる dblsn サービスタイプがサポートされるようになりました。

  サービスユーティリティでは、ユーティリティの出力のログをファイルに保存することができる -o オプションも使用できます。「Windows 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』と「Linux 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい SQL Anywhere Broadcast Repeater ユーティリティ (dbns10)**　SQL Anywhere Broadcast Repeater ユーティリティを使用すると、他のサブネット上で実行されている SQL Anywhere データベースサーバーや、ファイアウォールの外側にあって UDP ブロードキャストが通常は届かない SQL Anywhere データベースサーバーを、SQL Anywhere クライアントは HOST パラメーターや LDAP を使用することなく検索できます。「Broadcast Repeater ユーティリティ (dbns12)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいレポート送信ユーティリティ (dbsupport)**　新しいサポートユーティリティ (dbsupport) では、エラーレポートと統計の送信機能、更新 (EBF の可用性) の問い合わせ機能、以前に送信した問題が修正されたかどうかのチェック機能が提供されます。「サポートユーティリティ (dbsupport)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **アンロードユーティリティ (dbunload) の強化**　dbunload が次のように強化されています。

  ○dbunload がエラーを検出したときに、未処理の文のログが取られるようになりました。「アンロードの失敗」『SQL Anywhere サーバー データベース管理』を参照してください。

  ○アンロードしたテーブルでバイナリデータがサポートされるようになりました。

  ○データベースのアンロード処理のパフォーマンスを向上するために、さまざまな内部処理が強化されました。

  次の新しいオプションが追加されました。

  ○ **-dc**　データベースのすべての計算カラムの値を再計算します。

  ○ **-g**　再ロード時にマテリアライズドビューを初期化します。

  ○ **-k**　トレースサポート用に補助テーブルを作成します。このオプションを指定すると、sa_diagnostic_auxiliary_catalog テーブルに移植されます。このオプションは、トレーシングデータベースの作成時に役立ちます。

  ○ **-nl**　各テーブルに対する LOAD TABLE 文と INPUT 文を含み、データは含まない *reload.sql* ファイルを作成します。

  「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **検証ユーティリティ (dbvalid)**　新しいデータベース検証オプション -d が追加されました。このオプションを指定すると、チェックサム検証、孤立したテーブルページと BLOB の検査、構造検査などのデータベース検証を実行します。データはチェックアウトされません。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「検証ユーティリティ (dbvalid)」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースオプション

次のデータベースオプションが追加または強化されました。

- **ansi_substring データベースオプション** このオプションは、SUBSTRING 関数の動作を制御します。デフォルトでは、SUBSTRING 関数の動作が ANSI/ISO SQL/99 の動作と一致するようになりました。開始オフセットが負または 0 の場合は、文字列の左側が文字以外で埋められているかのように扱われ、このときに負の長さが指定されるとエラーになります。

- **collect_statistics_on_dml_updates データベースオプション** DML 文 (INSERT、DELETE、および UPDATE) を実行中の統計の収集を制御します。「collect_statistics_on_dml_updates オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **default_dbspace オプション** テーブルを作成するデフォルトの dbspace を指定できます。「default_dbspace オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **http_session_timeout オプション** http_session_timeout オプションを使用して、さまざまなセッションタイムアウトを制御できます。このオプションは分単位で指定します。デフォルトのパブリック設定は 30 分です。最小は 1 分で、最大は 525600 分 (365 日) です。「http_session_timeout オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **max_temp_space データベースオプション** max_temp_space オプションを指定したときに接続で使用可能なテンポラリ領域の最大量を指定できます。「max_temp_space オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **materialized_view_optimization データベースオプション** オプティマイザーによるマテリアライズドビューの使用を制御します。「materialized_view_optimization オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **oem_string データベースオプション** oem_string データベースオプションを使用して、データベースファイルのヘッダーページに情報を格納できます。この文字列は、アプリケーションからアクセスでき、バージョン情報を格納したり、そのデータベースファイルがアプリケーション用であることを検証したりするために使用できます。「oem_string オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **request_timeout データベースオプション** このオプションでは、接続が長時間にわたってサーバーリソースを大量に消費しないように、単一要求を実行できる最大時間を指定します。「request_timeout オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **synchronize_mirror_on_commit オプション** 非同期モードまたは非同期フルページモードでデータベースミラーリングを実行しているときに、データベースの変更がミラーサーバーに送信されたことを確定するタイミングを制御します。「synchronize_mirror_on_commit オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **uuid_has_hyphens データベースオプション**　uniqueidentifier 値が文字列に変換されるときのフォーマットを制御します。

- **verify_password_function データベースオプション**　verify_password_function データベースオプションは、パスワードルールを実装するために使用できる関数を指定します。この関数は GRANT CONNECT 文で呼び出します。「verify_password_function オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Java サポート用の新しいデータベースオプション**　次のデータベースオプションが追加されました。

  ○ java_location オプション :「java_location オプション」『SQL Anywhere サーバー データベース管理』を参照してください。
  ○ java_main_userid オプション [データベース]
  ○ java_vm_options オプション :「java_vm_options オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースサーバーオプション

「新機能」の項で説明されているサーバーオプションのほかに、次の新しいサーバーオプションが追加されています。

- **-cm サーバーオプション**　このサーバーオプションを使用して、Windows で Address Windowing Extensions (AWE) に割り当てるアドレス空間の大きさを指定できます。「-cm dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-dh サーバーオプション**　このサーバーオプションを使用すると、サーバーに対してサーバー列挙ユーティリティ (dblocate) を実行した場合でも、データベースが検出されないようになります。「-dh dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-dt サーバーオプション**　このサーバーオプションを使用して、テンポラリファイルが格納されるディレクトリを指定できます。UNIX で共有メモリ接続を使用するデータベースサーバーでは、このオプションを指定できません。「-dt dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-gtc サーバーオプション**　このオプションを使用して、CPU で同時実行できるスレッド数を制御できます。「-gtc dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-ot サーバーオプション**　このサーバーオプションを指定すると、メッセージが書き込まれる前にデータベースサーバーメッセージログファイルがトランケートされます。「-ot dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-su サーバーオプション**　このオプションを使用すると、ユーティリティデータベースへの接続時に DBA ユーザーのパスワードを指定できます。*util_db.ini* ファイルの代わりに -su を使

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

用する必要があります。「-su dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-zp サーバーオプション**　このサーバーオプションを使用すると、最後に使用したクエリ最適化プランが接続ごとに格納されます。「-zp dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

プロパティとパフォーマンスモニターの統計値

データベースを管理できるようにするための、接続、サーバー、データベースの新しいプロパティと、パフォーマンスモニターの新しい統計値が追加されました。

- **接続プロパティ**　このリリースには、次の接続プロパティが追加されています。
  - allow_snapshot_isolation
  - ansi_substring
  - ApproximateCPUTime
  - conn_auditing
  - default_dbspace
  - ExprCacheAbandons
  - ExprCacheDropsToReadOnly
  - ExprCacheEvicts
  - ExprCacheHits
  - ExprCacheInserts
  - ExprCacheLookups
  - ExprCache
  - GetData
  - HeapsCarver
  - HeapsLocked
  - HeapsQuery
  - HeapsRelocatable
  - HttpServiceName
  - http_session_timeout
  - java_location
  - java_main_userid
  - LastPlanText
  - LockCount
  - LockedCursorPages
  - LockTableOID
  - materialized_view_optimization
  - max_query_tasks
  - max_temp_space
  - MultiPageAllocs
  - NcharCharSet
  - oem_string
  - post_login_procedure
  - QueryHeapPages
  - ReqCountActive
  - ReqCountBlockContention
  - ReqCountBlockLock
  - ReqCountBlockIO
  - ReqCountUnscheduled
  - ReqTimeActive
  - ReqTimeBlockContention
  - ReqTimeBlockIO
  - ReqTimeBlockLock
  - ReqTimeUnscheduled
  - ReqStatus
  - RequestsReceived
  - RetryConnectionTimeout

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- SessionCreateTime
- SessionID
- SessionLastTime
- SnapshotCount
- synchronize_mirror_on_commit
- tsql_outer_joins
- verify_password_function

接続プロパティの詳細については、「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **サーバープロパティ**　このリリースには、次のサーバープロパティが追加されています。
  - CachePinned
  - CacheReadEng
  - CacheSizingStatistics
  - CarverHeapPages
  - ConsoleLogMaxSize
  - CollectStatistics
  - DebuggingInformation
  - DefaultNcharCollation
  - DiskReadEng
  - ExchangeTasks
  - FirstOption
  - FunctionMaxParms
  - FunctionMinParms
  - HeapsRelocatable
  - HeapsLocked
  - HeapsQuery
  - HeapsCarver
  - IsEccAvailable
  - IsRsaAvailable
  - LastConnectionProperty
  - LastDatabaseProperty
  - LastOption
  - LastServerProperty
  - MapPhysicalMemoryEng
  - MaxConnections
  - MultiProgrammingLevel
  - NumLogicalProcessors
  - NumLogicalProcessorsUsed
  - NumPhysicalProcessors
  - NumPhysicalProcessorsUsed
  - QueryHeapPages
  - RememberLastPlan
  - RemoteputWait
  - RequestFilterConn
  - RequestFilterDB
  - RequestLogMaxSize
  - RequestsReceived
  - ServerName
  - StartDBPermission

  これらのプロパティの詳細については、「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **データベースプロパティ**　このリリースには、次のデータベースプロパティが追加されています。

  - AccentSensitive
  - AlternateServerName
  - ArbiterState
  - AuditingTypes
  - CleanablePagesAdded
  - CleanablePagesCleaned
  - EncryptionScope
  - http_session_timeout
  - IOParallelism
  - JavaVM
  - LockCount
  - MirrorState
  - NcharCollation
  - NcharCharSet
  - NextScheduleTime
  - PartnerState
  - ReceivingTracingFrom
  - SendingTracingTo
  - SnapshotCount
  - SnapshotIsolationState
  - VersionStorePages
  - XPathCompiles

  これらのプロパティの詳細については、「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **パフォーマンスモニターの統計値のプロパティ**　このリリースには、次のパフォーマンスモニターの統計値が追加されています。

  - キャッシュ：マルチページ割り当て
  - キャッシュ：パニック
  - キャッシュ：スカベンジアクセス
  - キャッシュ：スカベンジ
  - キャッシュページ：割り当て構造体
  - キャッシュページ：ファイル
  - キャッシュページ：ファイルダーティ
  - キャッシュページ：空き
  - 通信：受信要求数
  - ヒープ：カーバー
  - ヒープ：クエリ処理
  - ヒープ：再配置可能ロック
  - ヒープ：再配置可能
  - メモリページ：カーバー
  - メモリページ：固定カーソル
  - メモリページ：クエリ処理
  - バージョンストアページ

これらの統計値の詳細については、「パフォーマンスモニターの統計値」『SQL Anywhere サーバー SQL の使用法』を参照してください。

システムプロシージャーとファンクション

次に、新しいシステムプロシージャーとファンクション、および既存のシステムプロシージャーとファンクションへの新しい拡張機能を示します。

- **DEFAULT 句をサポートするためのすべてのプロシージャーと関数の強化** プロシージャーとユーザー定義の関数では、値 DEFAULT に対応するパラメーターがデフォルト値として定義されている場合、DEFAULT を引数として使用できます。プロシージャーに複数のパラメーターがあり、デフォルトに設定されていないパラメーターがある場合は、名前付きのパラメーターを使用するよりも、引数リストで DEFAULT を指定する方が簡単な場合があります。また、名前付きのパラメーターは、関数呼び出しで使用できません。

- **新しいシステムプロシージャー** 次のシステムプロシージャーが追加されました。

  - **sa_clean_database システムプロシージャー** 指定した時間中、データベースクリーナーを実行します。「sa_clean_database システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_column_stats システムプロシージャー** sa_column_stats システムプロシージャーは、指定されたカラムについて文字列に関連する統計値を返します。「sa_column_stats システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_conn_list システムプロシージャー** sa_conn_list システムプロシージャーは、接続 ID を返します。「sa_conn_list システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_conn_options システムプロシージャー** sa_conn_options システムプロシージャーは、データベースオプションに対応する接続プロパティのプロパティ情報を返します。「sa_conn_options システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_db_list システムプロシージャー** sa_db_list システムプロシージャーは、データベース ID を返します。「sa_db_list システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_describe_query システムプロシージャー** sa_describe_query システムプロシージャーは、カラムごとに 1 つのローを返し、結果の式とその NULL 入力属性のドメインを記述します。このプロシージャーは、各カラムで EXPRTYPE 関数を実行することと同じです。「sa_describe_query システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_get_bits システムプロシージャー** sa_get_bits システムプロシージャーは、ビット文字列を復号化し、ビットの値を示すビット文字列の各ビットについて 1 つのローを返します。「sa_get_bits システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **sa_make_object システムプロシージャー**　sa_make_object システムプロシージャーで、イベントをオブジェクトタイプとして指定できるようになりました。「sa_make_object システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_materialized_view_info システムプロシージャー**　sa_materialized_view_info システムプロシージャーは、指定したマテリアライズドビューの情報 (ステータス、ビューの所有者など) を返します。「sa_materialized_view_info システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_refresh_materialized_views システムプロシージャー**　sa_refresh_materialized_views システムプロシージャーは、データベース内で現在初期化されていないすべてのマテリアライズドビューをリフレッシュします。「sa_refresh_materialized_views システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_remove_tracing_data システムプロシージャー**　このプロシージャーは、診断トレーシングテーブルから指定したロギングセッションのすべてのレコードを永続的に削除します。「sa_remove_tracing_data システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_save_trace_data システムプロシージャー**　このプロシージャーは、テンポラリトレーシングテーブルからベーステーブルにデータを保存します。「sa_save_trace_data システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_set_tracing_level システムプロシージャー**　プロファイルされるデータベースに対して生成されるトレーシングデータのレベルを設定します。「sa_set_tracing_level システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_snapshots システムプロシージャー**　データベースで現在アクティブなスナップショットのリストを返します。「sa_snapshots システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_split_list システムプロシージャー**　値のリストを表す文字列を引数に取り、そのリストを含む結果セットを返します。「sa_split_list システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_table_stats システムプロシージャー**　各テーブルから読み込まれたページ数に関する情報を返します。「sa_table_stats システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_transactions**　データベースに対して現在実行しているトランザクションのリストを返します。「sa_transactions システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_unload_cost_model と sa_load_cost_model システムプロシージャー**　新しいシステムプロシージャー sa_unload_cost_model と sa_load_cost_model を使用して、コストモデルをデータベースからアンロードし、別のデータベースにロードできるようになりました。これによって、同じようなハードウェアインストールが大量にある場合に、繰り返し行われる時間のかかる再調整作業がなくなりました。「sa_unload_cost_model システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』と「sa_load_cost_model システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい関数**　次の関数が追加されました。
  - **BIT_LENGTH 関数**　配列内に格納されたビット数を返します。「BIT_LENGTH 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **BIT_SUBSTR 関数**　ビット配列のサブ配列を返します。「BIT_SUBSTR 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **BIT_AND 関数**　2 つのビット配列を引数に取り、引数のビット処理 AND の結果を返します。比較する各ビットで両方のビットが 1 の場合は 1 を、それ以外の場合は 0 を返すというロジックが使用されます。「BIT_AND 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **BIT_OR 関数**　2 つのビット配列を引数に取り、引数のビット処理 OR の結果を返します。比較する各ビットで片方 (または両方) のビットが 1 の場合は 1 を、それ以外の場合は 0 を返すという論理が使用されます。「BIT_OR 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **BIT_XOR 関数**　2 つのビット配列を引数とし、引数のビット処理排他 OR の結果を返します。比較する各ビットで片方のビットだけ (両方のビットではなく) が 1 の場合は 1 を、それ以外の場合は 0 を返すというロジックが使用されます。「BIT_XOR 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **COUNT_SET_BITS 関数**　配列で 1 (TRUE) に設定されたビットの数を返します。「COUNT_SET_BITS 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **GET_BIT 関数**　ビット配列で指定したビットの値 (1 または 0) を返します。「GET_BIT 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **REVERSE 関数**　この新しい関数は、文字式のリバースを返します。「REVERSE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **SET_BIT 関数**　ビット配列の特定ビットの値を設定します。「SET_BIT 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **SET_BITS 関数**　指定したビット (ローセットからの値に対応するビット) が 1 (TRUE) に設定されたビット配列を作成します。「SET_BITS 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - **TRACED_PLAN 関数**　トレーシングデータと、クエリがトレースされたときのオプティマイザーの状態に関する情報を使用して、クエリのグラフィカルなプランを生成します。「TRACED_PLAN 関数 [その他]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
- **さまざまなシステムプロシージャーとファンクションの強化**　次のシステムプロシージャーとファンクションが強化されました。
  - **プロパティ関数の強化**　プロパティ関数が LONG VARCHAR を返すようになりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

次の項を参照してください。

- 「CONNECTION_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』
- 「DB_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』
- 「PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』

○ **DB_EXTENDED_PROPERTY 関数の強化** DB_EXTENDED_PROPERTY 関数で NextScheduleTime データベースプロパティを使用して、イベントの次にスケジュールされている実行時刻を取得できるようになりました。CHAR 文字セットの拡張情報を返す関数も使用できます。「DB_EXTENDED_PROPERTY 関数 [システム]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **新しい CONNECTION_EXTENDED_PROPERTY 関数** CONNECTION_EXTENDED_PROPERTY 関数を使用して、特定の接続パラメーターの拡張情報を検索できます。「CONNECTION_EXTENDED_PROPERTY 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **sa_procedure_profile システムプロシージャー** sa_procedure_profile システムプロシージャーでは、出力をファイルに保存できるようになりました。また、新しい構文が追加され、いくつかのパラメーターが必須になり、新しい使用方法が追加されました。「sa_procedure_profile システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **sa_procedure_profile_summary システムプロシージャー** sa_procedure_profile_summary システムプロシージャーでは、出力をファイルに保存できるようになりました。また、新しい構文が追加され、いくつかのパラメーターが使用できるようになり、新しい使用方法が追加されました。「sa_procedure_profile_summary システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **sa_server_option システムプロシージャー** sa_server_option システムプロシージャーを使用すると、データベースサーバーの実行中に設定を変更できます。次の設定を変更できるようになりました。

- **CacheSizingStatistics プロパティ** キャッシュサイズが変更されるたびに、データベースサーバーメッセージウィンドウにキャッシュ情報を表示します。
- **CollectStatistics プロパティ** データベースサーバーのパフォーマンスモニターの統計値を収集します。
- **ConsoleLogFile プロパティ** データベースサーバーメッセージウィンドウの情報が記録される出力ファイルの名前を指定します。
- **ConsoleLogMaxSize プロパティ** データベースサーバーメッセージウィンドウ情報の記録に使用される出力ファイルの最大サイズを指定します。
- **DebuggingInformation プロパティ** トラブルシューティング目的で診断通信メッセージやその他のメッセージを表示します。
- **IdleTimeout サーバーオプション** 指定された時間の間、要求を送信しなかった TCP/IP 接続または SPX 接続を切断します。

- **ProfileFilterConn プロパティ**　その他の接続によるデータベースの使用を妨げることなく、特定の接続 ID のプロファイリング情報を取得します。

- **RequestFilterDB プロパティ**　sa_server_option システムプロシージャーを使用して、要求ロギング用の単一データベースに対する接続をフィルターできるようになりました。

- **RequestLogging プロパティ**　要求ログでブロックイベント、アンブロックイベント、プラン情報、プロシージャー、トリガーを記録できるようになりました。

- **RequestTiming プロパティ**　要求タイミングをオンにすると、各要求のタイミング情報を管理するように、データベースサーバーに指示されます。

詳細については、「sa_server_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **xp_startsmtp システムプロシージャーの強化**　xp_startsmtp システムプロシージャーでは、smtp_user_name、smtp_auth_username、smtp_auth_password の新しい 3 つのパラメーター サポートされています。「xp_startsmtp システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **xp_sendmail システムプロシージャーの強化**　xp_sendmail システムプロシージャーが、include_file パラメーターを使用することで、SMTP を使用してメールを送信するときに添付ファイルをサポートするようになりました。また xp_sendmail では、content_type パラメーターを使用することで、SMTP メールを使用するときの MIME コンテンツをサポートします。「xp_sendmail システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **sa_conn_info システムプロシージャーが新しいプロパティ値を返す**　sa_conn_info システムプロシージャーが、追加プロパティ ClientPort、ServerPort、LockTable を返すようになりました。また、LastIdel プロパティを返さなくなり、UncmtOps 値の名前は UncommitOps に変更されました。「sa_conn_info システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **sa_performance_diagnostics がより多くの情報を返す**　スナップショットアイソレーションの使用時に、sa_performance_diagnostics システムプロシージャーが LockCount と SnapshotCount を返すようになりました。「sa_performance_diagnostics システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **HASH 関数の強化**　HASH 関数で新しいアルゴリズム SHA256、SHA1_FIPS、SHA256_FIPS を使用できるようになりました。FIPS 認定のアルゴリズムは、FIPS 認定ソフトウェアを使用するシステムのみで使用されます。「HASH 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **COMPRESS 関数と DECOMPRESS 関数で新しいアルゴリズムをサポート**　関数で文字列を圧縮および解凍するために gzip アルゴリズムを使用できるようになりました。「COMPRESS 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』と「DECOMPRESS 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

SQL 文

次に、新しい SQL 文と、既存の SQL 文の構文に対する新しい拡張機能について説明します。これらの新機能は、このマニュアルで前述した機能に関する項に記載されている文の変更に追加される機能です。

- **マテリアライズドビューをサポートする SQL 文** マテリアライズドビューをサポートするために、次の SQL 文が追加または構文や機能が拡張されました。

  - ALTER MATERIALIZED VIEW 文：「ALTER MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - COMMENT 文：「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - CREATE INDEX 文：「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - CREATE MATERIALIZED VIEW 文：「CREATE MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - DROP MATERIALIZED VIEW 文:「DROP MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - REFRESH MATERIALIZED VIEW 文：「REFRESH MATERIALIZED VIEW 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - VALIDATE 文：「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  SELECT 文の新しい OPTION 句は、materialized_view_optimization データベースオプションを上書きするために使用できます。「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **診断トレーシングとアプリケーションプロファイリングをサポートする新しい SQL 文** アプリケーションプロファイリングをサポートするための新しい SQL 文は次のとおりです。

  - ATTACH TRACING 文：「ATTACH TRACING 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - DETACH TRACING 文：「DETACH TRACING 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - REFRESH TRACING LEVEL 文：「REFRESH TRACING LEVEL 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい VALIDATE DATABASE 文** VALIDATE DATABASE 文を使用してデータベースを検証できるようになりました。「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい VALIDATE MATERIALIZED VIEW 文** VALIDATE MATERIALIZED VIEW 文を使用して、マテリアライズドビューを検証できるようになりました。「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい ALTER STATISTICS 文** ALTER STATISTICS 文を使用して、カラムの統計値が自動的に更新されるようにするかを制御できるようになりました。自動更新が無効の場合でも、明示的に CREATE STATISTICS 文または DROP STATISTICS 文を使用することで、統計値を

強制的に更新できます。「ALTER STATISTICS 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER INDEX 文の強化**　ALTER INDEX 文の REBUILD 句を使用して、インデックスを再構築できるようになりました。「ALTER INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER TABLE 文と CREATE TABLE 文の強化**　MATCH 句を使用して、参照元テーブルの外部キーと参照先テーブルのプライマリキーの間で一致させる内容を詳細に制御できるようになりました。また、外部キーをユニークとして宣言できるようになり、一意性を個別に宣言する必要がなくなりました。「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER DATABASE 文の新しい CALIBRATE PARALLEL READ 句**　並列入出力が可能なハードウェアを検出するには、ALTER DATABASE 文の新しい CALIBRATE PARALLEL READ 句を使用します。dbspace の調整結果を取得するには、DB_EXTENDED_PROPERTY 関数を使用して新しい IOParallelism 拡張データベースプロパティを問い合わせます。「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』と「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **COMMENT 文の PRIMARY KEY ON 句**　COMMENT 文の PRIMARY KEY ON 句を使用して、プライマリキーに注釈を作成できるようになりました。「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **暗号化キーを変更するための CREATE ENCRYPTED FILE 文の強化**　CREATE ENCRYPTED FILE 文の拡張機能を使用して、データベースのアンロードと再ロードを行わずにデータベース、トランザクションログ、dbspace の暗号化に使用する暗号化キーを変更できるようになりました。データベースが暗号化されていないがテーブルの暗号化が有効な場合は、CREATE ENCRYPTED FILE 文を使用してテーブルの暗号化に使用するキーを変更できます。「CREATE ENCRYPTED FILE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE DATABASE 文の強化**　文字セットの処理を向上させるために、新しい句 ENCODING、NCHAR COLLATION、ACCENT が追加されました。また、データベースの初期サイズを指定できるように、DATABASE SIZE 句が追加されました。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SELECT 文の強化**　カーソルによるローの更新に使用される FOR UPDATE 句が拡張され、カラムリストでは、後置された UPDATE 文を使用して変更可能なカラムを制限することができるようになりました。「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  SELECT 文の FROM 句が拡張され、READPAST テーブルヒントと UPDLOCK テーブルヒントがサポートされるようになりました。READPAST テーブルヒントは、データベースサーバーに対してロックされたローを無視するように指示します。UPDLOCK テーブルヒントは、XLOCK と同様に動作します。「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  SELECT 文が拡張され、その文のクエリ最適化を制御する OPTION 句がサポートされるようになりました。OPTION 句には、その SELECT 文の MATERIALIZED VIEW OPTIMIZATION

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

句によるマテリアライズドビューの一致を制御するための構文が含まれます。2 つ目の句 FORCE OPTIMIZATION は、コストベースの最適化をバイパスするようにクエリが修飾されている場合であっても、データベースサーバーに対してクエリの最適化を実行するように指示します。「SELECT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **LOAD TABLE 文と UNLOAD TABLE 文の強化**　LOAD TABLE 文の STRIP 句では、引用符で囲まれていない値を挿入する前に、その値から先行ブランクを削除するかを制御できるオプションを使用できるようになりました。追加の STRIP オプションを使用すると、データが削除される方法を微調整できます。

  LOAD TABLE 文が拡張され、COMMENTS INTRODUCED BY オプションがサポートされるようになりました。このオプションを使用して、入力データ内のコメントを識別するための文字列を指定できます。指定した文字列で始まる入力行は、ロード操作時にすべて無視されます。

  LOAD TABLE 文と UNLOAD TABLE 文が拡張され、次のオプションがサポートされるようになりました。

  - **ENCODING オプション**　データのロードやアンロードのときに使用するエンコーディングを指定するのに使用されます。
  - **ROW DELIMITED BY オプション**　データのバルクロードやバルクアンロードのときの入力レコードの末尾を示す文字列を指定するのに使用されます。
  - **QUOTE オプション**　Interactive SQL の OUTPUT 文の QUOTE オプションに似ています。「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「UNLOAD 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **VALIDATE INDEX 文の強化**　VALIDATE INDEX の構文が強化され、インデックス指定がサポートされるようになりました。「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **プライマリキーの名前を変更するための ALTER INDEX 文の強化**　ALTER INDEX 文を使用して、プライマリキーの名前を変更できるようになりました。「ALTER INDEX 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい CONTINUE 文**　この文は、ループを再起動するために使用します。CONTINUE 文に続くループ内の文はスキップされます。「 CONTINUE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **新しい BREAK 文 [T-SQL]**　この文は、複合文またはループから出るために使用します。「BREAK 文 [T-SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **INSERT 中にデフォルト値を更新する INSERT 文制御の強化**　DEFAULTS ON | OFF 句を使用すると、ローがすでに存在するときに INSERT 中にデフォルト値が更新されるかを制御できます。DEFAULT TIMESTAMP、DEFAULT UTC TIMESTAMP、DEFAULT LAST USER の各デフォルトフィールドには、この新機能は適用されません。これらのフィールドは常時更新されます。「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ORDER BY 句をサポートするための DELETE 文の強化**　DELETE 文で ORDER BY 句がサポートされ、データベースからローを削除する順序を指定できるようになりました。「DELETE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **START DATABASE 文の強化**　START DATABASE 文は、データベースが起動できなかった理由を示せなかったときに、幅広いエラーメッセージを返すようになりました。また、START DATABASE 句は任意の順序で指定できるようになりました。「START DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **イベントログやシステムログのみへのロギングをサポートするための MESSAGE 文の強化**　ロギングのオン／オフを制御できるだけでなく、イベントログやシステムログのみにロギングするかを指定できます。MESSAGE 文の構文が拡張され、TO LOG 句内でオプション句 [EVENT | SYSTEM] を使用できるようになりました。たとえば TO EVENT LOG は、イベントログのみにロギングできます。「MESSAGE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **FOR OLAP WORKLOAD オプション**　CREATE INDEX、CREATE TABLE、ALTER TABLE の各文の構文が拡張され、外部キー定義で FOR OLAP WORKLOAD オプションがサポートされるようになりました。このオプションは、OLAP パフォーマンスを向上させるために、特定の最適化を実行してキーの統計値を収集するように、データベースサーバーに対して指示します。「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』と「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **テンポラリストアドプロシージャーのサポート**　CREATE PROCEDURE 文の拡張機能を使用して、テンポラリストアドプロシージャーを作成できるようになりました。テンポラリストアドプロシージャーは、そのストアドプロシージャーを作成した接続のみが参照でき、接続が切断されると自動的に切断されます。「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ローカルテンポラリテーブルのサポート**　CREATE LOCAL TEMPORARY TABLE 文を使用して、ローカルテンポラリテーブルを作成できるようになりました。この方法で作成されたローカルテンポラリテーブルは、接続が閉じると切断されます。「CREATE LOCAL TEMPORARY TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **テンポラリテーブルの強化**　CREATE LOCAL TEMPORARY TABLE 文の SHARE BY ALL 句を使用して、データベースへのすべての接続でデータが共有されるグローバルテンポラリテーブルを作成できるようになりました。「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

データ型

- **CHAR と VARCHAR データ型の文字長セマンティックのサポート**　CHAR または VARCHAR カラムを指定するときに、文字長セマンティックを使用できるようになりました。文字長セマンティックを使用すると、長さをバイト数ではなく文字数で表現できます。「CHAR データ型」『SQL Anywhere サーバー SQL リファレンス』と「VARCHAR データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **ビット配列データ型のサポート**　SQL Anywhere で VARBIT と LONG VARBIT データ型がサポートされるようになりました。これらのデータ型は、ビット配列を格納するために使用されます。「ビット配列データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ビット配列データ型で使用される次の関数が追加されました。

  - BIT_LENGTH 関数 [ビット配列]：「BIT_LENGTH 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - BIT_SUBSTR 関数 [ビット配列]：「BIT_SUBSTR 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - BIT_AND 関数 [集合]：「BIT_AND 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - BIT_OR 関数 [集合]：「BIT_OR 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - BIT_XOR 関数 [集合]：「BIT_XOR 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - COUNT_SET_BITS 関数 [ビット配列]：「COUNT_SET_BITS 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - GET_BIT 関数 [ビット配列]：「GET_BIT 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - SET_BIT 関数 [ビット配列]：「SET_BIT 関数 [ビット配列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - SET_BITS 関数 [集合]：「SET_BITS 関数 [集合]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

プログラミングインターフェイス

- **ADO.NET 2.0 のサポート**　ADO.NET ドライバーが更新され、.NET Framework のバージョン 2.0 がサポートされるようになりました。その一環で、新しいクラスとメソッドが追加されました。「ネームスペース」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL Anywhere Explorer**　SQL Anywhere Explorer を使用すると、Visual Studio .NET 内から SQL Anywhere データベースに接続できます。また、Sybase Central と Interactive SQL を Visual Studio .NET から直接開くことができます。

- **iAnywhere JDBC ドライバーで JDBC 3.0 のサポート**　iAnywhere JDBC ドライバーで JDBC 3.0 呼び出しがサポートされるようになりました。iAnywhere JDBC ドライバーで JDBC 2.0 がサポートされなくなりました。既存のアプリケーションを変更しないで引き続き実行できるように、ianywhere.ml.jdbcodbc.IDriver クラスと ianywhere.ml.jdbcodbc.jdbc3.IDriver クラスのサポートは継続されますが、どちらのドライバーも現在は同じであり、JDBC 3.0 だけを実装します。バージョン 1.4 より前の JRE は iAnywhere JDBC ドライバーとともに使用できなくなりました。「JDBC サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **iAnywhere JDBC ドライバーでの SQL Server Native Client ODBC ドライバーのサポート**　iAnywhere JDBC ドライバーは、ODBC ドライバーが Microsoft SQL Server Native Client ODBC ドライバーであるかを確認し、デフォルトの結果セットタイプやその他の属性を適切に設定します。

- **PreparedStatement.addBatch メソッドのサポート**　iAnywhere JDBC ドライバーで PreparedStatement.addBatch メソッドがサポートされるようになりました。このメソッドはバッチ (またはワイド) 挿入用にサポートされています。

- **ODBC ドライバーに SQL_GUID のサポート追加**　SQL Anywhere ODBC ドライバーに、UNIQUEIDENTIFIER カラムのサポートが追加されました。UNIQUEIDENTIFIER カラムは SQL_GUID として入力できるようになりました。

- **ODBC ドライバーに GUID エスケープシーケンスのサポート追加**　SQL Anywhere ODBC ドライバーに、GUID エスケープシーケンスのサポートが追加されました。GUID エスケープシーケンスは、ODBC 経由で準備および実行された SQL 文で使用できます。GUID エスケープシーケンスの形式は、{guid 'nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn'} です。

- **ODBC メッセージコールバックが接続ごとになった**　ODBC では、SQL Anywhere バージョン 9.0.0 からメッセージコールバックをサポートしていますが、すべての接続のメッセージは単一のコールバック関数に送られていました。バージョン 9.0.2 からは、メッセージコールバック関数を指定すると、単一の接続のみに適用されるようになりました。これは、DBLIB の動作方法と一貫性があります。すべてのメッセージが、ODBC ドライバーの単一関数に集まるようになりました。この関数は、メッセージを接続ごとにフィルターし、コールバック関数を備えた接続に対して、そのコールバック関数だけを呼び出します。

- **SQL Anywhere PHP モジュールに追加された新しい関数**　SQL Anywhere PHP モジュールに追加された新しい関数は次のとおりです。

  ○ sqlanywhere_execute
  ○ sqlanywhere_error
  ○ sqlanywhere_errorcode
  ○ sqlanywhere_insert_id

  また、2 つの新しいオプション verbose_errors と row_counts が sqlanywhere_set_option 関数に追加されました。「SQL Anywhere PHP API リファレンス」『SQL Anywhere サーバー プログラミング』を参照してください。

- **db_locate_servers_ex 関数の強化**　db_locate_servers_ex 関数で、2 つの新しいフラグ DB_LOOKUP_FLAG_ADDRESS_INCLUDES_PORT と DB_LOOKUP_FLAG_DATABASES がサポートされるようになりました。DB_LOOKUP_FLAG_ADDRESS_INCLUDES_PORT は、コールバック関数に渡された a_server_address 構造体内の TCP/IP ポート番号を返します。DB_LOOKUP_FLAG_DATABASES は、見つかったデータベースまたはデータベースサーバーごとにコールバック関数を 1 回呼び出すことを指定します。「db_locate_servers_ex 関数」『SQL Anywhere サーバー プログラミング』を参照してください。

- **Perl DBI モジュールの Perl DBD::ASAny ドライバー名の変更**　Perl ドライバーの名前が DBD::ASAny から DBD::SQLAnywhere に変更されました。SQL Anywhere を使用する Perl スクリプトは、新しいドライバー名を使用するように変更する必要があります。ネイティブ SQL Anywhere 型を返すカーソル属性 ASATYPE は、変更されていません。また、型名がありません (ASA_STRING、ASA_FIXCHAR、ASA_LONGVARCHAR など)。「Perl DBI サポート」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQL プリプロセッサー (sqlpp) の -o オプション値**　sqlpp の -o オプションで、Microsoft Windows 用に、WINNT ではなく WINDOWS を使用できるようになりました。また、サポー

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

トされている 64 ビットの UNIX オペレーティングシステム用に、UNIX64 を指定できるようになりました。「SQL プリプロセッサー」『SQL Anywhere サーバー プログラミング』を参照してください。

新しい ODBC ドライバーマネージャーと ODBC ドライバーの強化

- **ODBC ドライバーマネージャーの強化** ODBC ドライバーマネージャーで、すべての ODBC 3.x 呼び出し、ワイド CHAR エントリポイント、接続トレーシングがサポートされるようになりました。また、ODBC ドライバーマネージャーを使用して、非スレッド化またはスレッド化された SQL Anywhere ドライバー間で切り替えができるようになりました。

- **スレッド化アプリケーションと非スレッド化アプリケーションの両方での ODBC ドライバーマネージャーの使用** スレッド化アプリケーションと非スレッド化アプリケーションの両方で、ODBC ドライバーマネージャーを使用できるようになりました。

配備

- **Deployment ウィザード** Deployment ウィザードが追加され、SQL Anywhere for Windows の配備ファイルを作成できるようになりました。Deployment ウィザードを使用すると、Microsoft Windows Installer パッケージファイルと Microsoft Windows Installer Merge Module ファイルの両方を作成できます。以前のバージョンの SQL Anywhere で提供されていた InstallShield のマージモジュールとテンプレートは、提供されなくなりました。代わりに、Deployment ウィザードを使用して、SQL Anywhere の配備ファイルを作成してください。「Deployment ウィザード」『SQL Anywhere サーバー プログラミング』を参照してください。

Windows CE の強化

- **WindowsCE での動的キャッシュサイズ決定のサポート** WindowsCE で動的キャッシュサイズ決定がサポートされるようになりました。Windows や UNIX のように、Windows CE でも、データベースサーバーの負荷とその他のシステムメモリへの要求に応じて、データベースサーバーのキャッシュサイズが増減します。この機能を使用すると、さまざまな状況で明示的にキャッシュサイズを選択する必要がなくなり、パフォーマンスが向上します。

- **Windows CE 用プロキシポートの作成** 以前のリリースのソフトウェアでは、Windows CE デバイス上のデータベースに接続するためにプロキシポートを使用するように ActiveSync を設定するには、レジストリのエントリを変更する必要がありました。Interactive SQL、Sybase Central、SQL Anywhere コンソールユーティリティの [接続] ウィンドウに、Windows CE プロキシポートの設定ツールが追加されています。これにより、レジストリを変更しなくても、Windows CE デバイス上のデータベースに接続するためのプロキシポートを作成できるようになりました。

- **Windows CE 上のデータベースを再構築可能** dbunload ユーティリティを使用してデータベースをファイルにアンロードし、そのファイルを新しいデータベースに再ロードするという 2 段階の処理で、Windows CE 上のデータベースを再構築できるようになりました。「Windows Mobile でのデータベースの再構築」『SQL Anywhere サーバー データベース管理』を参照してください。

- **dbrunsql ユーティリティ** スクリプト実行ユーティリティ (dbrunsql) を使用すると、Windows CE 上で SQL 文を入力したりスクリプトファイルを実行したりすることができま

す。「スクリプト実行ユーティリティ (dbrunsql)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Windows Mobile 5 用の署名済み .cab ファイル**　SQL Anywhere インストールには、VeriSign によって署名された、構築済みの *.cab* ファイルが含まれます。これらの *.cab* ファイルを使用した場合、発行元が不明であることを知らせる警告は表示されません。

UNIX/Linux の強化

- **UNIX プラットフォームで使用できる新しい ODBC ドライバーマネージャー**　UNIX プラットフォームで ODBC ドライバーマネージャーとして libdbodm10 共有オブジェクトを使用できるようになりました。iAnywhere ODBC ドライバーマネージャーを使用するアプリケーションでは、依存する ODBC をバージョン 3.0 以上に制限する必要があります。「UNIX 用 SQL Anywhere ODBC ドライバーマネージャー」『SQL Anywhere サーバー プログラミング』を参照してください。

- **-uf サーバーオプション**　-uf オプションを使用すると、UNIX で致命的なエラーが発生したときにデータベースサーバーが対処する方法を指定できます。「-uf dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Linux でサービスユーティリティのサポート**　サービスユーティリティを Linux で使用して、SQL Anywhere サービスを作成、削除、リスト、開始、停止できるようになりました。

- **UNIX でサポートされる追加サンプル**　UNIX でサポートされるようになった SQL Anywhere サンプルは次のとおりです。

  - **DBTools**　このサンプルは、SQL Anywhere サンプルデータベースのバックアップを作成することで、データベースツールライブラリを呼び出してコンパイルする方法を説明するデータベースツールアプリケーションです。

  - **DiskFull**　Disk-Full Callback サンプル DLL を説明するためのサンプルです。

  - **HTTP**　HTTP ディレクトリにあるサンプルでは、Web サービスのさまざまな機能の例が示されています。これには、Web サービスを使用して SQL プロシージャーからクライアント側 cookie を設定および取得する方法、HTTP Web サービスプロシージャーからバイナリデータを処理する方法、フォームや HTML テーブルを使用して簡単なカレンダーを表示する方法、xp_read_file を使用してローカルディスクからイメージを取得し、HTTP 要求への応答として返す方法、HTTP セッションを作成、使用、削除する方法などがあります。

  - **oemString**　OEM ソフトウェア用にデータベースファイルが設定されているかどうかを判断する方法を説明するためのサンプルです。

  - **PerformanceFetch**　fetchtest を使用して任意のクエリのフェッチ速度をテストする方法を説明するためのサンプルです。

  - **PerformanceInsert**　instest を使用してテーブルへの挿入速度をテストする方法を説明するためのサンプルです。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **Linux でファイバ (スレッド親和性) のサポートと同時実行処理の向上**　Linux データベースサーバー用の SQL Anywhere に、Windows ファイバの処理モデルに似た、新しいコルーチン処理モデルが導入されました。この処理モデルにより、タスクやルーチン間のコンテキストスイッチ処理をサーバーがより制御できるようになり、データベース操作との親和性が向上します。

- **Linux でのダイレクト I/O のサポート**　Linux 版 SQL Anywhere で、Linux 2.6 O_DIRECT 機能がサポートされるようになりました。この機能を使用すると、ファイルシステムが I/O をキャッシュしなくなるため、I/O パフォーマンスが向上します。

- **Linux での非同期 I/O**　Linux 版 SQL Anywhere で、Linux AIO 機能がサポートされるようになりました。この機能を使用すると、単一のアプリケーションスレッドでも I/O 操作を他の処理と重複できるようになるため、I/O パフォーマンスが向上します。

- **Linux のデスクトップ GUI**　Linux 版 SQL Anywhere で、パーソナルデータベースサーバー、ネットワークデータベースサーバー、Mobile Link サーバー、Mobile Link 同期クライアント、SQL Remote 用の追加デスクトップ GUI が提供されるようになりました。この GUI は、GTK ライブラリがインストールされている場合に、-ui オプションで呼び出すことができます。

- **Linux のデスクトップアイコン**　Linux 版 SQL Anywhere では、Linux デスクトップを使用するユーザーが、データベースサーバー、Sybase Central、Interactive SQL、SQL Anywhere コンソールユーティリティ、Mobile Link モニターの開始と管理をしやすくするために、Linux デスクトップ用にオプションでインストールできるアイコンが提供されるようになりました。

- **IBM AIX、HP-UX、Sun Solaris に対する 64 ビットクライアントのサポート**　64 ビットメモリモデルで SQL Anywhere クライアントライブラリを使用できるようになり、IBM AIX、HP-UX、Sun Solaris のアプリケーションで大量のメモリを使用できるようになりました。

Web サービス

- **Web サーバーの HTTP 1.1 への準拠**　HTTP 1.1 への準拠により、Web サーバーで次の項目を受け入れるようになりました。

  - HTTP 要求のパイプライン処理。GET や HEAD などの複数の HTTP 要求を同時に実行できます。
  - 絶対 URI (以前は相対 URI のみサポートされていました)。
  - 100-continue request-header フィールド。クライアントは要求本文全体を送信する前に、サーバーが要求を受け入れるかを (要求ヘッダーを基に) 判断できます。
  - Accept-Charset request-header フィールドの quality 値 (以前はこれらの値は無視されていました)。

- **HTTP クライアントの HTTP 1.1 への準拠**　実装された HTTP 関連の強化は次のとおりです。

  - **HTTP 文字列メモリプール処理のサポート**　HTTP 文字列は連続したメモリに格納されなくなりました。キャッシュがバックエンド記憶領域として使用されます。

  - **クライアントのチャンクモード**　HTTP クライアントは HTTP チャンクモードを使用して POST 要求を送信できるようになりました。

- **HTTP セッション**　HTTP 接続は、HTTP 要求間のステータスを管理する HTTP セッションを作成できます。

- **HTTP サーバーの keep-alive オプション**　データベースサーバーで、HTTP クライアントからの要求時に keep-alive オプションがサポートされるようになりました。要求を終えるごとに接続を閉じるのではなく、要求と応答を終えても HTTP 接続を開いたままにしておくことができるため、同じ接続で複数の要求を実行できます。

  この機能をサポートするために、KeepaliveTimeout プロトコルオプションも追加されました。「KeepaliveTimeout (KTO) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい HttpServiceName 接続プロパティ**　新しい接続プロパティ HttpServiceName が追加され、Web アプリケーションでサービス名のオリジンを判断できるようになりました。このプロパティは、エラーレポートやフロー制御の場合に便利です。「接続プロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **sa_set_http_option の強化**　sa_set_http_option システムプロシージャーを使用して、要求の Accept-Charset request-header フィールドを基に、HTTP 応答で使用される文字セットを制御できるようになりました。「sa_set_http_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SOAP サービスでのデータ型指定のサポート**　CREATE SERVICE 文と ALTER SERVICE 文が拡張され、新しい DATATYPE 句がサポートされるようになりました。この句は、SOAP サービスでのみ使用するもので、入力パラメーターと出力応答でデータ型指定をサポートするかどうかを制御します。「CREATE SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』と「ALTER SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_set_soap_header システムプロシージャー**　sa_set_soap_header システムプロシージャーを使用して、SOAP サービス用の応答ヘッダーを設定します。「sa_set_soap_header システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SOAP_HEADER 関数と NEXT_SOAP_HEADER 関数**　SOAP_HEADER 関数を使用して、SOAP サービスの要求ヘッダーを取得します。「SOAP_HEADER 関数 [SOAP]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  NEXT_SOAP_HEADER 関数を使用して、SOAP ヘッダー内の次のヘッダーエントリを取得します。「NEXT_SOAP_HEADER 関数 [SOAP]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **CREATE PROCEDURE 文、ALTER PROCEDURE 文、CREATE FUNCTION 文、ALTER FUNCTION 文の HEADER 句**　これらの文には新しく HEADER 句が追加されています。この句は、HTTP Web サービスのクライアントプロシージャーと関数を作成するときに使用されます。この句を使用すると、HTTP 要求のヘッダーエントリを追加または修正できます。

  「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「クライアント提供の HTTP 変数とヘッダーのアクセス」『SQL Anywhere サーバー プログラミング』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **CREATE PROCEDURE 文、ALTER PROCEDURE 文、CREATE FUNCTION 文、ALTER FUNCTION 文の SOAPHEADER 句**　これらの文には新しく SOAPHEADER 句が追加されています。この句は、SOAT Web サービスのクライアントプロシージャーと関数を作成するときに使用されます。この句では、IN (IN/OUT) 代入パラメーターを使用して、送信する SOAP ヘッダーエントリと受信する SOAP ヘッダーデータを指定できます。

  「CREATE PROCEDURE 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「CREATE FUNCTION 文 [Web サービス]」『SQL Anywhere サーバー SQL リファレンス』、「チュートリアル：SQL Anywhere を使用した SOAP/DISH サービスへのアクセス」『SQL Anywhere サーバー プログラミング』を参照してください。

その他

- **インデックス処理の強化**　今回のリリースでは、インデックス処理が次のように強化されています。

  - **インデックス共有処理のサポート**　プライマリキー、セカンダリキー、外部キー、または一意性制約を作成するときに、物理インデックス (ディスク上の実際のインデックス構造) を指す論理インデックスを作成できるようになりました。データベースサーバーは、論理インデックスを満たすために新しい物理インデックスが必要かを自動的に判断します。このモデルを使用すると、物理インデックスの共有処理が可能になり、重複した物理インデックスを作成したり管理したりすることがなくなり、ディスク領域を無駄にすることがなくなります。「高度：論理インデックスと物理インデックス」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  - **インデックス情報の記憶領域の向上**　データベースでインデックス情報が編成される方法が向上されました。たとえば、プライマリキーと外部キーのインデックスを含むすべてのインデックスのリストが、単一のシステムテーブル ISYSIDX に格納されるようになりました。「高度：カタログ内のインデックス情報」『SQL Anywhere サーバー SQL の使用法』を参照してください。

    3 つの新しいシステムテーブル ISYSPHYSIDX、ISYSIDXCOL、ISYSFKEY によって、ISYSIDX にリストされているインデックスの追加情報が提供されます。

  - **インデックスコンサルタントの強化**　インデックスコンサルタントが強化され、クラスタードインデックス、負荷時のデータベースとサーバーのステータス、負荷統計値の完全なレポートに関する推奨事項が改善されました。インデックスコンサルタントはアプリケーションプロファイリングツールに統合されました。

  - **インデックスの作成方法の制御が向上**　アプリケーションが参照整合性制約 (プライマリキー、外部キー、または一意性の制約) を作成すると、データベースサーバーは制約のキーを構成するカラムでインデックスを暗黙的に作成することで、その制約を確保します。データベースサーバーで、インデックスの作成方法を指定できるようになりました。制約キーのカラムの順序を指定したり、インデックスの各カラムに値のシーケンス (昇順または降順) を指定できます。また、外部キーのカラムの順序やシーケンスを、対応するプライマリキーや一意性制約と一致させる必要はありません。

    その他に、次の機能も強化されています。

- プライマリキーの順序を変更するときに、テーブルのカラムを並べ替える必要がなくなりました。
- すべての制約インデックスのカラムのシーケンスを、アプリケーションの要件を満たすように指定できるようになりました。
- プライマリテーブルの設計と関連させなくても、外部キーのインデックスを外部キーテーブルに関するアプリケーションの要件に適合させることができるようになりました。
- 外部キーに一意性制約を設定できるようになりました。

- **新しい外部ジョインの削除によるリライト最適化** 外部ジョインがクエリから削除されてもそのクエリが元のクエリと意味が変わらないときは、クエリの実行前に外部ジョインが削除されます。「クエリ処理中に実行される最適化」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **日付フォーマット文字列での文字長セマンティックの使用** 日付フォーマット文字列で、フォーマット指定子に置き返されるテキストの量を制御するために、文字長セマンティックを使用するようになりました。以前はバイト長セマンティックが使用されていました。たとえば、文字列 MMM を使用して日付をフォーマットする場合、以前は月を格納するために 3 バイトを使用することを示していましたが、3 文字を意味するようになりました。

- **ディレクトリアクセスサーバー** ディレクトリアクセスサーバーを作成することで、データベースサーバーを実行しているコンピューターのディレクトリ構造にアクセスするリモートサーバーを作成できるようになりました。「ディレクトリアクセスサーバー」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ノルウェー語の照合** 1252NOR が追加され、ノルウェー語がサポートされるようになりました。ノルウェー語の Windows システムで照合が指定されていない場合、データベースサーバーは、新規データベースのデフォルトの照合として 1252NOR を選択します。「サポートされている照合と代替照合」『SQL Anywhere サーバー データベース管理』を参照してください。

- **UTF8BIN の照合** UTF8BIN 照合が追加され、バイナリデータのソート処理が向上しました。この新しい照合は、廃止予定の UTF8 照合に代わるものです。「サポートされている照合と代替照合」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースサーバーメッセージウィンドウの強化** データベースサーバーメッセージウィンドウが、次のように強化されました。

  - **新しいウィンドウタイトルバーの右クリックメニュー** サポートされるすべての Windows プラットフォーム (Windows CE を除く) で、データベースサーバーメッセージウィンドウのタイトルバーを右クリックすると、[バージョン情報] または [メッセージ領域のクリア] を選択できるようになりました。[バージョン情報] を選択すると、データベースサーバーの情報が表示されます。[メッセージ領域のクリア] を選択すると、データベースサーバーメッセージウィンドウのすべてのメッセージが消去されます。このウィンドウのレプリカ (データベースサーバーメッセージログファイル、Sybase Central の [サーバーメッセージと実行された SQL] ウィンドウ枠、SQL Anywhere コンソールユーティリティ) は、消去操作の影響を受けません。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **データベースサーバーで使用される環境変数がデータベースサーバーメッセージウィンドウにロギングされる**　-ze サーバーオプションを使用すると、データベースサーバーメッセージウィンドウにデータベースサーバーの環境変数のリストが表示されます。この機能は NetWare または WindowsCE では使用できません。「-ze dbeng12/dbsrv12 オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **開始時のウィンドウ最小化の制御**　データベースサーバーが起動すると、デフォルトでデータベースサーバーメッセージウィンドウは最小化されます。データベースサーバーの起動後に、データベースサーバーメッセージウィンドウが最小化されないにするには、-qn オプションを指定します。「-qn dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **テーブルの前回更新日時の追跡機能**　データベースサーバーが、前回テーブルが更新された日時を追跡できるようになりました。SYSTAB システムビューの新しい last_modified_at カラムを使用します。

- **ミラーリング中に別のサーバーに切り替わったときに SNMP トラップ**　SQL Anywhere SNMP Extension Agent とミラーリング中のサーバーとの接続が切断されて、別のサーバーとの接続が新しく確立された場合、SQL Anywhere SNMP Extension Agent によってトラップが送信されるようになりました。

  このトラップは、元のサーバーがダウンしていて、ミラーとして動作していたサーバーがプライマリになったことを示します。「トラップ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **要求のロギングの変更**　要求のログはカンマ区切りのテキストフォーマットで格納されるようになり、元のサイズの約 3 分の 1 にまで小さくなりました。また、可能な場合は通常の時刻エントリではなく、等号 (=、ログで直前のエントリと同じ時刻という意味) または *+nnn* (*nnn* はログで直前のエントリから経過したミリ秒数) として時刻が記録されるようになりました。また、追加情報も記録されるようになりました。たとえばクエリでは、独立性レベル、フェッチされたローの数、カーソルタイプが記録されます。INSERT、UPDATE、DELETE 文の場合は、影響を受けたローの数と起動されたトリガーの数が記録されます。「要求ロギング」『SQL Anywhere サーバー SQL の使用法』を参照してください。

  sa_get_request_times システムプロシージャーでは新しい要求のログフォーマットだけをサポートします。ただし、tracetime Perl スクリプト *tracetime.pl* は新旧両方の要求のログフォーマットを処理します。tracetime スクリプトでは、新しいフォーマットのログの方が高速に処理できます。この違いは要求のログのサイズが大きいとより顕著になります。

- **ODBC ドライバーの強化**　SQL Anywhere は、Adaptive Server Enterprise と DB2 のデータベースに接続するときに、新しいドライバーを使用してリモートデータアクセスを行います。「Mobile Link、QAnywhere、リモートデータアクセスで使用される ODBC の変更」362 ページを参照してください。

- **SQLANYSAMP10 環境変数**　SQLANYSAMP10 環境変数は、*demo.db* や *custdb.db* サンプルデータベースなどの SQL Anywhere 10 のサンプルがあるディレクトリのロケーションを指定します。「SQLANYSAMP12 環境変数」『SQL Anywhere サーバー データベース管理』を参照してください。

バージョンサポート

この項では、サポート対象または対象外のサードパーティコンポーネントのバージョン番号に関する変更点について説明します。

- **バージョン 8 以前のデータベースのサポート** Sybase Central、Interactive SQL、または SQL Anywhere コンソールユーティリティを使用する場合、SQL Anywhere 8.0.0 以前のバージョンで作成したデータベースはサポートされなくなりました。古いソフトウェアで作成し、新しいソフトウェアでアップグレードしたデータベースも含まれます。このようなデータベースをロードしようとすると、データベースの開始時にエラーになります。Sybase Central からこのようなデータベースに接続して、新しいバージョン 10 データベースにアンロードすることができます。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **jConnect バージョン 5.5 と 6.0.5 のサポート** SQL Anywhere では、データベースサーバーへの接続で jConnect バージョン 5.5 と 6.0.5 がサポートされるようになりました (バージョン 5.5 は 9.0.2 でもサポートされていました)。jConnect は、http://www.sybase.com/products/informationmanagement/softwaredeveloperkit/jconnect から別途ダウンロードできます。サポートされている機能については、jConnect のマニュアルを参照してください。

- **Intel x86 アーキテクチャーのプロセッサーの要件** Intel x86 アーキテクチャーでは、SQL Anywhere は Pentium 以降のプロセッサーだけをサポートし、80386 や 80486 などの古いプロセッサーでは起動できません。

動作の変更

次に、バージョン 10.0.0 で導入された SQL Anywhere データベースに加えられた変更を、カテゴリごとに示します。

その他

- **Adaptive Server Anywhere の名前の変更** バージョン 10.0.0 では、Adaptive Server Anywhere の名前が SQL Anywhere に変更されました。

- **アップグレードの変更** バージョン 9.0.2 以前のデータベースをバージョン 10 にアップグレードするために、データベースアップグレードウィザード、アップグレードユーティリティ (dbupgrad)、ALTER DATABASE UPGRADE 文を使用することはできません。以前のバージョンのデータベースをバージョン 10 にアップグレードするには、アンロードと再ロードを実行し、データベースを再構築する必要があります。「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

- **パスワードの変更** 新しく作成されたデータベースでは、データベースでの設定にかかわらず、すべてのパスワードは大文字と小文字が区別されます。新しいデータベースのデフォルトの DBA パスワードは、**sql** です。

  既存のデータベースを再構築する場合、パスワードの大文字と小文字の区別は、次のように決まります。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○パスワードを最初に入力したのが大文字と小文字を区別しないデータベースだった場合、そのパスワードの大文字と小文字は区別されません。

○パスワードを最初に入力したのが大文字と小文字を区別するデータベースだった場合、大文字のパスワードと、大文字と小文字が混在したパスワードでは、大文字と小文字が区別されます。ただし、パスワードをすべて小文字で入力した場合、パスワードの大文字と小文字は区別されません。

○既存のパスワードと新しいパスワードの両方に加えられた変更は、大文字と小文字が区別されます。

データベースサーバーで、パスワードのハッシュ処理に SHA256 が使用されるようになりました。古いデータベースから再ロードされるパスワードには古い (独自の) ハッシュ処理アルゴリズムが引き続きサポートされていますが、新しいパスワードにはすべて SHA256 が使用されます。

パスワードは UTF-8 で格納されるようになりました。そのため、異なる文字セットを使用するデータベースに再ロードされても、データベースは動作し続けます。

以前のリリースでは、Embedded SQL から接続すると、DBA パーミッションでデータベースに接続した後で、どのユーザーでもパスワードを指定することなく同じデータベースに対して 2 つ目の接続を確立できていました。今回のリリースでは、接続ごとにパスワードの指定が必要になりました。

- **ブランク埋め込みの変更**　SQL Anywhere の以前のリリースでは、ブランクを埋め込まれたデータベースで文字列を比較した場合のセマンティックは、比較される 2 つの文字列に無限のブランクが埋め込まれているかのようになっていました。バージョン 10 では、これらのセマンティックが変更され、各文字列の後続ブランクを無視して比較を行うようになりました。

  等号 (=) 比較と不等号 (<>) 比較の場合は、セマンティックに変更はありません。どちらの方法 (ブランクの埋め込みと後続ブランクの無視) でも同じ結果になります。ただし、不等比較の場合は異なります。たとえば、2 バイトの文字列 'a*' があるとします ('*' は、ブランクの値より小さい、データベースの照合順の文字を表します)。以前のバージョンの SQL Anywhere では、比較述部 'a*' < 'a' は TRUE を返していました。バージョン 10 では、短い文字列には比較前にブランクが埋め込まれないので、この述部は FALSE となります。

  ブランクの埋め込みの詳細については、「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』と「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **プロパティの戻り値の大文字と小文字**　サーバープロパティ (PROPERTY 関数で返される) は以前のバージョンでは YES または NO を返していましたが、Yes または No を返すようになりました。データベースプロパティ (DB_PROPERTY 関数で返される) と接続プロパティ (CONNECTION_PROPERTY 関数で返される) は、以前のバージョンでは ON または OFF を返していましたが、On または Off を返すようになりました。この変更は、大文字と小文字を区別するデータベースや、大文字と小文字を区別する文字列比較を使用するアプリケーションに影響する可能性があります。

- **サーバープロパティの戻り値の変更**　以前のリリースでは、サーバープロパティ ConnsDisabled と RememberLastStatement は値 ON と OFF を返していました。これらは値 Yes

と No を返すようになりました。「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **sasrv.ini ファイルのデフォルトロケーションの変更**　*sasrv.ini* のデフォルトのロケーションは、Windows では *%ALLUSERSPROFILE%¥Application Data¥SQL Anywhere 10*、UNIX では *$HOME/.sqlanywhere10* です。以前のリリースでは、UNIX でのファイル名は *.sasrv.ini* でした。このファイルの名前は、すべてのプラットフォームで、*sasrv.ini* になりました。

- **長い名前のデータベースサーバーへの接続**　Windows と UNIX では、バージョン 9.0.2 以前のクライアントは、40 バイトより長い名前が設定されたバージョン 10.0.0 以降のデータベースサーバーに接続できません。

- **データベースサーバーで文字セットの変換は常に有効**　文字セットの変換を有効／無効にする -ct データベースサーバーオプションはサポートされなくなりました。データベースサーバーで文字セット変換は常に有効になりました。ただしデータベースサーバーで変換が不要であると判断された場合は、使用されません。文字セット変換を無効にするには、クライアントから CharSet=none と指定します。「CharSet (CS) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Windows CE でサポートされていない文字セット変換**　文字セット変換は Windows CE ではサポートされていません。以前のリリースでは、文字セット変換は Windows CE 用のデータベースサーバー上では無効になっており、データベースにはどの文字セットも使用することができました。今回のリリースでは、オペレーティングシステムの文字セットまたは UTF-8 のいずれかを使用して、Windows CE 用にデータベースを作成する必要があります。「インストール時の考慮事項 : Windows Mobile での ICU の使用」『SQL Anywhere サーバー データベース管理』を参照してください。

- **システムプロシージャーとファンクションの変更**　システムプロシージャーとファンクションの変更は次のとおりです。

  - **複数のシステムプロシージャーの内部への組み込み**　外部システムプロシージャーの xp_read_file、xp_write_file、xp_sprintf、xp_scanf、xp_cmdshell が、内部システムプロシージャーになりました。

  - **sa_validate システムプロシージャー**　sa_validate システムプロシージャーで、VALIDATE 権限が必要になりました。「sa_validate システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_reset_identity システムプロシージャー**　table-name パラメーターが必須になりました。また、owner-name パラメーターを指定しない場合は、table-name パラメーターがデータベース内のテーブルをユニークに識別する必要があります。「sa_reset_identity システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **sa_locks システムプロシージャー**　sa_lock システムプロシージャーの出力が変更され、追加情報 (接続 ID、ユーザー ID、テーブル名、ロッククラス、ロック期間) を返すようになりました。「sa_locks システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  - **RAND 関数**　以前のバージョンでは、各接続のシードが同じ値に設定されていたため、RAND 関数が各接続で同一のシーケンスを返すことがありました。今回のリリースでは、

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

各接続がさまざまなランダムシーケンスを認識するため、各接続はユニークなシードが設定されるようになりました。「RAND 関数 [数値]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

○ **コールバック関数 DB_CALLBACK_START と DB_CALLBACK_FINISH**　コールバック関数 DB_CALLBACK_START と DB_CALLBACK_FINISH がすべてのプラットフォームでサポートされるようになりました (以前は Windows プラットフォームのみでサポートされていました)。「db_register_a_callback 関数」『SQL Anywhere サーバー プログラミング』を参照してください。

● **DATE データ型**　DATE データ型に時間や分が格納されなくなりました。

● **UNC 名を使用して指定されたファイルで分散読み込みが使用されなくなった**　リモートコンピューター上のファイルや UNC 名 (たとえば *¥¥mycomputer¥myshare¥mydb.db*) で指定されたファイルに対して、分散読み込みが使用されなくなりました。「適切なページサイズの使用」『SQL Anywhere サーバー SQL の使用法』を参照してください。

● **プライマリキー制約と外部キー制約でのカラムの順序**　プライマリキー制約を作成するときは、任意の順序でカラムを指定できます。テーブルにカラムが出現する順序は関係ありません。また、外部キーのカラムとプライマリキーのカラムのマッピングを指定すれば、参照先のプライマリキーとカラムの順序が異なる外部キーを作成できるようになりました。「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』の PRIMARY KEY 句を参照してください。

● **インデックスで重複したカラム名を使用できなくなった**　以前のバージョンでは、インデックスのカラムを重複参照できていましたが、プライマリキー、外部キー、一意性制約指定は除外されていました。今回のバージョンでは、すべてのタイプのインデックスで動作が一貫されるようになりました。カラム名を重複して指定すると、エラーが返されます。また、旧バージョンのデータベースに重複したカラム参照のあるインデックスが含まれていると、dbunload ユーティリティによって *reload.sql* の生成時に重複したカラムがインデックスから削除されます。「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

● **暗号化データベースプロパティ**　SELECT DB_PROPERTY('Encryption') を実行すると、データベースが暗号化されていない場合でも、None 以外の値を返すようになりました。これは、データベースでテーブル暗号化が有効な場合に発生します。アプリケーションでデータベースが暗号化されているかを確認する方法としてこの文を実行する場合は、代わりに SELECT DB_PROPERTY('EncryptionScope') を使用してください。「データベースプロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

● **FIPS を使用して HTTPS を開始する場合の構文の変更**　この場合、以前のバージョンでは -xs HTTPS_FIPS(...) を指定していました。今回のバージョンでは、-xs HTTPS(FIPS=yes;...) を指定する必要があります。以前の構文はサポートはされますが、廃止される予定です。「-xs dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

● **ユーザー ID の最大長は 128 バイト**　以前のリリースでは、文でユーザー ID が必要な場合、データベースサーバーでは 128 バイトを超えるユーザー ID を使用する前にトランケートしていました。string_rtruncation オプションを設定した場合は、トランケーションエラーが返さ

れていました。これが 128 バイトを超えるユーザー ID を指定した場合は、string_rtruncation オプションの設定に関係なく、データベースサーバーがエラーを返すようになりました。「識別子」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **サーバー名の最大長**　TCP/IP 接続と共有メモリ接続で、データベースサーバー名の最大長が 40 バイトから 250 バイトに増加しました。「-n dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **識別子で使用可能な文字の変更**　識別子で二重引用符と円記号を使用できなくなりました。「識別子」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **LicensesInUse サーバープロパティ名の変更**　サーバープロパティ LicensesInUse の名前は、UniqueClientAddresses に変更されました。「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SQL Anywhere OLE DB プロバイダー名の変更**　SQL Anywhere OLE DB プロバイダーの名前は、以前は ASAProv、ASAProv.90、ASAProv.80 でしたが、SAOLEDB になりました。バージョン 10 のプロバイダーは、SAOLEDB.10 という名前で具体的に参照できます。

- **SQL Anywhere サンプルデータベース ODBC DSN の変更**　ODBC データソースの名前は、以前は ASA 9.0 Sample でしたが、SQL Anywhere 10 Demo になりました。

- **接続文字列の変更**　ODBC 接続と OLE DB 接続で、接続パラメーターを検出する場所の優先度が接続文字列、SQLCONNECT 環境変数、データソースの順になりました。以前のバージョンでは、ODBC 接続と OLE DB 接続でデータソースの優先度は SQLCONNECT よりも高くなっていました。「接続パラメーターの構文ルール」『SQL Anywhere サーバー データベース管理』を参照してください。

- **空の値を使用した接続パラメーターが未指定として扱われるようになった**　すべての API で、空の値で指定された接続パラメーターは、パラメーターが指定されなかったものとして扱われます。以前のリリースでは空の値は、指定された場所と使用されている API に応じて未指定または空の文字列として扱われていました。「接続パラメーターの構文ルール」『SQL Anywhere サーバー データベース管理』を参照してください。

- **監査がオンの場合はトランザクションログをオフにできない**　以前のバージョンのソフトウェアでは、監査がオンになっているデータベースでトランザクションログの使用を停止できました。今後は、データベースで監査がオンになると、トランザクションログを使用する必要があります。トランザクションログの使用を中止する場合は、監査をオフにする必要があります。

- **監査がオンになっているデータベースは読み込み専用モードで起動できない**　以前のバージョンのソフトウェアでは、監査がオンになっているデータベースを読み込み専用モードで開始できました。今回のバージョンでは、監査がオンになっているデータベースは、読み込み専用モードで起動できません。

- **符号付き BIGINT カラムの精度が 20 から 19 になった**　以前は、ODBC アプリケーションで SQL_BIGINT を使用して符号付き BIGINT カラムを記述すると、不適切な精度 20 の値が返されていました。今回のバージョンでは、精度 19 の値が返されるようになりました。以前の (不適切な) 値に依存するアプリケーションは変更する必要があります。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **Java VM の強化**　SQL Anywhere では、Java オプションは別途ライセンスが必要なコンポーネントとして提供されなくなりました。データベース内の Java は、Java コードを実行するために内部 VM を使用するのではなく、外部 VM を使用するようになりました。これによって、任意の Java VM を使用できるようになり、特定の JDK バージョンや Java ターゲットに限定されなくなりました。新しく初期化されるデータベースは、常に Java 対応になります。

  その結果、次の変更が行われました。

  - **サポート対象外のデータベースオプション**　SQL Anywhere による以下のオプションのサポートは終了しました。

    - describe_java_format
    - java_heap_size
    - java_namespace_size
    - java_page_buffer_size
    - java_input_output
    - return_java_as_string

  - **サポート対象外のプロパティ**　以下のプロパティは、サポートされなくなりました。

    ●データベースプロパティ

      - JDKVersion
      - JavaHeapSize
      - JavaNSSize

    ●データベースサーバープロパティ

      - IsJavaAvailable
      - JavaGlobFix

    ●接続プロパティ

      - JavaHeapSize
      - java_input_output

  - **新しい JavaVM プロパティ**　JavaVM データベースプロパティは、データベース内の Java を実行するためにデータベースサーバーが使用する Java VM のパスを返します。

  - **サポート対象外の互換ビューカラム**　システム互換ビューで、次のカラムを使用できなくなりました。

    - SYSINFO.classes_version
    - SYSJAVACLASS.replaced_by
    - SYSJAVACLASS.type_id

- **データベースユーティリティで廃止された Java オプション**　次のデータベースユーティリティプロパティは廃止される予定です。
  - 初期化ユーティリティ (dbinit)：-ja、-jdk
  - アンロードユーティリティ (dbunload)：-jr
  - アップグレードユーティリティ (dbupgrad)：-ja、-jdk、-jr、-j
- **CREATE DATABASE 文と ALTER DATABASE 文の一部の Java 関連句で Java サポートが廃止された**　CREATE DATABASE 文では、JAVA ON | OFF 句と、JDK バージョン句がサポートされなくなりました。ALTER DATABASE 文では、REMOVE JAVA 句がサポートされなくなりました。
- **新しい Java ファイル**　前述の変更のほかに、*java¥sajvm.jar* ファイルが追加されました。

- **Ping ユーティリティ (dbping)**　以前は、データベースサーバーがプロパティ値に対して NULL を返すと、Ping ユーティリティ (dbping) はエラーを返していました。今回のバージョンでは、プロパティ値が不明の場合に、dbping は NULL を出力し、成功のリターンコードで終了するようになりました。プロパティ値が不明の場合に dbping が失敗のエラーコードで終了するようにする場合は、-en オプションを指定できます。「Ping ユーティリティ (dbping)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **環境変数の名前の変更**　このリリースでは、次の環境変数の名前が変更されています。

以前の名前	新しい名前
ASTMP	SATMP
ASDIR	SADIR
ASLOGDIR	SALOGDIR
ASLANG	SALANG
ASCHARSET	SACHARSET

- **PHP モジュールのファイル名の変更**　PHP ファイルの命名規則が変更されました。以前のバージョンでは、ファイル名は *phpX_sqlanywhereY.dll* という形式でした (*X* は PHP のメジャーバージョン番号、*Y* は SQL Anywhere のメジャーバージョン番号)。今回のバージョンでは、PHP モジュールファイルの名前は *php-a.b.c_sqlanywhereY.dll* という形式になります (*a.b.c* はファイルのビルド元 PHP ソースの完全なバージョン番号、*Y* は SQL Anywhere のメジャーバージョン番号)。たとえば *php-5.0.2_sqlanywhere10.dll* のようになります。

- **PrefetchBuffer 接続パラメーターの値の指定**　下位互換のため、PrefetchBuffer 接続パラメーターで 16384 未満の値がキロバイトとして解釈されるようになりました。k サフィックスなしでキロバイトを使用する方法は廃止されます。PrefetchBuffer の値が有効な範囲外にあったり k サフィックスなしでキロバイト単位を指定したりしたためにこの値が調整される場合は、実際に使用された PrefetchBuffer 値がクライアントログファイルに示されます。「PrefetchBuffer (PBUF) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **システム定義のドメインを削除できない**　システム定義のドメイン (MONEY、UNIQUEIDENTIFIERSTR など) はデータベースから削除できなくなりました。「DROP DOMAIN 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **データベースユーティリティの変更**　前述したとおり、データベースユーティリティは次のように変更されています。

  - **サービスユーティリティ (dbsvc) でサービスとしてログインする権限を付与できる**　サービスユーティリティ (dbsvc) で -a オプションを使用し、サービスとしてログインする権限が有効でないアカウントで実行しようとすると、サービスとしてログインする権限を付与するかどうかを指定するプロンプトが表示されます。-y オプションを使用すると、dbsvc はプロンプトせずに、サービスとしてログインする権限を付与しようとします。「Windows 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **アンロードユーティリティ (dbunload) の -an オプションをリモートサーバーに使用できる**　この変更以前には、dbunload -an は同じコンピューター上のサーバーのみに対して実行できていました。今回のリリースでは、dbunload -an を別のコンピューターで実行しているサーバーに対して実行できるようになりました。「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **サーバー列挙ユーティリティ (dblocate) ホスト名または IP アドレスのフォーマット**　ホスト名や IP アドレスは、-n を指定したかどうかに関係なく、任意のフォーマットで使用できます。たとえばサーバーが myhost.mycompany.com で実行中で、この IP アドレスが 1.2.3.4 の場合、このコンピューターで実行しているサーバーだけを mycompany.com ドメインの任意のコンピューターからリストするには、dblocate myhost、dblocate myhost.mycompany.com、dblocate 1.2.3.4 のいずれでも使用できます。以前のバージョンでは、指定されたホスト名や IP アドレスは dblocate によって表示されるアドレス文字列 (ポート番号を除く) に一致する必要があったため、dblocate myhost.mycompany.com または dblocate -n 1.2.3.4 だけが機能していました。「サーバー列挙ユーティリティ (dblocate)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **デフォルト関連の変更**　デフォルトは、次のように変更されています。

  - **パーソナルデータベースサーバーのデフォルト TCP/IP 受信アドレスの変更**　Windows では、パーソナルデータベースサーバーは 0.0.0.0 ではなく、127.0.0.1 で接続を受信するようになりました。この変更により、Windows ファイアウォールを有効にして SQL Anywhere を実行している場合は、dbeng10 の使用前に例外リストに追加する必要がなくなりました。

    これにより、*hostname* がコンピューターの実際のホスト名や IP アドレスの場合、LINKS=tcpip(HOST=hostname;DOBROADCAST=none) で接続しようとしても機能しません。ただし、ホスト名 **localhost** または **127.0.0.1** を使用すると機能します。

  - **デフォルトデータベースページサイズが 4096 に変更**　SQL Anywhere データベースのデフォルトデータベースページサイズが、4096 バイトから 2048 バイトに変更されました。このページサイズにすると、多くの環境でパフォーマンスが向上することがわかっています。「CREATE DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

-gp オプションを指定しないで、データベースがロードされていない状態でデータベースサーバーを起動した場合、そのデータベースサーバーのデフォルトページサイズは 4096 になります。

○ **デフォルト最大キャッシュサイズの変更**　Windows (非 AWE) のデフォルトの最大キャッシュサイズが増加しました。デフォルトの最大キャッシュサイズは次のどちらか小さい方になりました。

●90% of (total_physical_memory - 4 MB)、2 MB 以上

●(available address space - 512 MB)

○ **UNIX キャッシュサイズ**　UNIX での最大キャッシュサイズの計算方法が変更されました。デフォルトの最大キャッシュサイズは次のように計算されます。

●32 ビットの UNIX プラットフォームでは、物理メモリ量の合計の 90% または 1,834,880 KB のいずれか小さい方です。

64 ビットの UNIX プラットフォームでは、物理メモリ量の合計の 90% または 8,589,672,320 KB のいずれか小さい方です。

「-ch dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

○ **UNIX ストアドプロシージャー**　既存の UNIX アプリケーションをアップグレードするときに、64 ビットデータベースサーバーを使用している場合は、既存の外部ストアドプロシージャーを 64 ビットに変更する必要があります。

○ **NULL 定数を NUMERIC または文字列データ型に変換する場合のデフォルトサイズ**　NULL 定数を NUMERIC データ型または文字列データ型 (CHAR、VARCHAR など) に変換する場合、長さが 32767 ではなく 0 に設定されるようになりました。

○ **openxml システムプロシージャーのデフォルト URI の変更**　openxml システムプロシージャーを使用する場合、名前空間宣言が指定されなかったときは、デフォルトでプレフィクス mp が Uniform Resource Identifier (URI) にバインドされます。以前のリリースでは、この URI は urn:ianywhere-com:asa-xpath-metaprop でした。この値が urn:ianywhere-com:sa-xpath-metaprop に変更されました。「openxml システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

● **-c、-ch-、-cl サーバーオプションのキャッシュサイズの割合を計算する方法の変更**　-c、-ch、または -cl を指定して P (パーセント) を使用すると、物理システムメモリ量または使用可能なアドレス空間量のどちらか小さい方に対して割合が計算されるようになりました。アドレス割り当てに使用できるよりも多くのメモリをキャッシュに割り当てようとするリスクがなくなります。「-c dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』、「-ch dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』、「-cl dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

● **Procedure_profiling サーバーオプション名の変更**　プロシージャープロファイリングを制御するサーバーオプションの正しい名前が ProcedureProfiling になりました。以前の

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Procedure_profiling も使用できますが、将来のリリースでサポートされなくなる予定です。「sa_server_option システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **デフォルトポートを使用していない HP-UX 上のデータベースサーバーに接続しているクライアントで、TCP/IP ポート番号の指定不要**　以前のリリースで、HP-UX でデータベースサーバーを起動する場合、デフォルトポート (2638) が使用中またはデフォルトポートを使用しないのであれば、ServerPort [PORT] プロトコルオプションを使用してポート番号を指定する必要がありました。

  HP-UX では、1 つのコンピューターで複数のデータベースサーバーが起動している場合に、TCP/IP の ServerPort プロトコルオプションは不要になりました。Mac OS X では、サーバーが同じコンピューターですでに実行している場合にネットワークサーバーを起動するには、依然として TCP/IP の ServerPort オプションを指定する必要があります。「ServerPort (PORT) プロトコルオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **SOAP CONCRETE 応答の名前が ASADataSet から SimpleDataset に変更**　CONCRETE 応答の名前が ASADataSet から SimpleDataset に変更されました。「CREATE SERVICE 文 [SOAP Web サービス]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **データベースアンロードウィザードの動作の変更**　データベースをバージョン 10 以前のデータベースバージョンにアンロードできなくなりました。バージョン 9.0.2 以前のデータベースをバージョン 10 のデータベースにアンロードすると、再構築が完了してもデータベースに自動的に接続できません。

- **データベース抽出ウィザードの動作の変更**　バージョン 9.0.2 以前のデータベースを抽出できません。バージョン 10 のデータベースから抽出する必要があります。

- **Solaris でサポート対象外の -ui と -ux サーバーオプション**　Solaris で -ui と -ux サーバーオプションはサポートされなくなりました。Linux では引き続き使用できます。

- **数値データ型の変換**　DOUBLE 型から NUMERIC に変換する場合に、元の DOUBLE 値の近似精度が最も高いアルゴリズムが使用されるようになりました。これらの変更により、有効桁数が 15 桁以下の DOUBLE 値は、正確に NUMERIC に変換されます。場合によっては、SQL Anywhere の以前のバージョンとは異なる結果になることがあります。「数値セットの変換」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **sa_validate システムプロシージャーの変更**　sa_validate システムプロシージャーのデータ、インデックス、フルの各オプションは必要なくなり、使用できなくなる予定です。式やチェックサムの検証を要求しないかぎり、以前のデータ、インデックス、フルのオプションを使用して実行されていた検査は、デフォルトで実行されます。

- **a_validate_type 列挙体の変更**　a_validate_type 列挙体の VALIDATE_DATA、VALIDATE_INDEX、VALIDATE_FULL の各パラメーターは必要なくなり、使用できなくなる予定です。VALIDATE_NORMAL が指定されると、これらのオプションで実行された検証は、デフォルトで実行されます。検証列挙 [データベースツール]『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQLPATH 環境変数の構文の変更**　UNIX での SQLPATH 環境変数の構文が変更されました。以前のバージョンでは、すべてのオペレーティングシステムで、各パス要素をセミコロン (;)

で区切っていました。SQL Anywhere 10 では、UNIX プラットフォームではパス要素をコロン (:) で区切り、その他のプラットフォームではセミコロンで区切ります。

- **CharSet 接続パラメーターの変更**　以前は、CharSet=NONE と指定すると、接続の文字セット変換が無効になりました。現在では、CharSet=NONE を指定すると、接続でデータベース CHAR 文字セットの使用が要求されます。「CharSet (CS) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

データベースオプションの変更

- **大文字と小文字の区別とデータベースオプション**　SET OPTION 文と CONNECTION_PROPERTY 関数では、使用するオプション名の大文字と小文字を区別しません。ただし、トルコ語の照合を使用するデータベースや大文字と小文字を区別するデータベースでは、マニュアルで指定された大文字と小文字を使用して、クエリで参照されるオプション名を記述する必要があります。「アルファベット順のオプションリスト」『SQL Anywhere サーバー データベース管理』を参照してください。

  このような場合、オプション名の大文字と小文字が正しくないと、SYSOPTION のクエリや次のようなクエリでは、どのローとも一致しない可能性があります。

  ```
  SELECT *
    FROM sa_conn_properties()
    WHERE propname = 'BLOCKING';
  ```

- **ansi_blanks が On に設定された状態での Embedded SQL の使用**　ansi_blanks が On でブランクが埋め込まれたデータベースを使用する Embedded SQL では、データ型 DT_STRING の値を指定する場合は、sqllen フィールドを値が格納されたバッファーの長さ (少なくとも値の長さと末尾の NULL 文字用のスペースの合計) に設定する必要があります。

  ブランクが埋め込みが有効になっているデータベースでは、ansi_blanks オプションの設定によって、フェッチする式が CHAR または NCHAR であり (VARCHAR または NVARCHAR ではない)、ホスト変数 char または nchar (VARCHAR または NVARCHAR ではない) に格納される場合に、トランケーション警告をクライアントに送信するかどうかが制御されます。「ansi_blanks オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **ansi_integer_overflow オプションのデフォルト設定の変更**　新しいデータベースが作成されるときの、ansi_integer_overflow データベースオプションのデフォルト値は On です。以前のバージョンでは、このオプションのデフォルト値は Off でした。

- **date_format オプションの変更**　date_format では、フォーマット文字列を指定するときに次の値をサポートしなくなりました。

  - **hh**　2 桁の時間
  - **nn**　2 桁の分
  - **ss[.ss..]**　秒数とコンマ以下の秒数
  - **aa**　午前／午後の識別子 (A.M. または P.M.、12 時間表記)
  - **aaa[a...]**　午前／午後の識別子 (A.M. または P.M.、12 時間表記)

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○ **pp**　(必要に応じて) 午後の識別子 (P.M.、12 時間表記)

○ **ppp[p...]**　(必要に応じて) 午後の識別子 (P.M.、12 時間表記)

また、文字データがマルチバイトの場合、各記号の長さが文字数を反映するようになりました。たとえば 'mmm' は 3 文字の月名を示しています。以前のバージョンでは、この記号の長さはバイト数を反映していました。「date_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **login_mode データベースオプション**　login_mode データベースオプションの値 Mixed は廃止される予定です。標準ログインと統合化ログインの両方を可能にする場合は、Standard,Integrated を指定します。「login_mode オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **string_rtruncation オプションのデフォルト設定の変更**　新しいデータベースを作成するときの string_rtruncation データベースオプションのデフォルト値は On です。以前のバージョンでは、このオプションのデフォルト値は Off でした。「string_rtruncation オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

  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。それ以外の場合はエラーになります。

  文字列のトランケートには LEFT 関数を使用することをおすすめします。「LEFT 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **temp_space_limit_check オプションのデフォルト設定の変更**　temp_space_limit_check オプションのデフォルトの設定は On に変更されました。デフォルトでは、接続で指定値以上のテンポラリファイル領域が要求されると、要求は失敗し、エラー SQLSTATE_TEMP_SPACE_LIMIT が返されるようになりました。「temp_space_limit_check オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **timestamp_format オプションの変更**　timestamp_format オプションでは、フランス語の月日の使用がサポートされなくなりました。また、文字データがマルチバイトの場合、各記号の長さが文字数を反映するようになりました。たとえば 'mmm' は 3 文字の月名を示しています。以前のバージョンでは、この記号の長さはバイト数を反映していました。「timestamp_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **truncate_date_values オプションの削除**　truncate_date_values オプションは削除されました。以前のリリースでは、このオプションを使用すると DATE データ型を使用して定義されたカラムに時刻を含めることができました。今回のリリースでは、DATE で定義されたカラムには日付だけを含めることができます。日付と時刻を格納する場合は、TIMESTAMP データ型を使用してください。「TIMESTAMP データ型」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

サーバーオプションの変更

- **-ec サーバーオプションと -xs サーバーオプションで強力な暗号化タイプの TLS 構文が変更**　-ec サーバーオプション、-xs サーバーオプション、暗号化接続パラメーターで、強力な暗号

化タイプの構文が変更されました。none、simple、tls の 3 種類だけになりました。タイプとして使用するキー交換アルゴリズムを指定するのではなく、暗号化タイプとして tls を指定し、アルゴリズムの指定に新しいプロトコルオプション tls_type を使用できるようになりました。「-ec dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』、「-xs dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』、「Encryption (ENC) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-os サーバーオプション**　以前のリリースでは、-os データベースサーバーオプションにより、ログファイルの名前が *current-filename.old* に変更されました。これが -on オプションの動作となりました。-os データベースサーバーオプションでは、出力ログの最大サイズ (このサイズに達するとログの名前が変更される) を指定するようになりました。以前は、-os を使用すると 2 つのログファイルが作成されていましたが、今回のバージョンでログファイルの数が無制限になりました。「-os dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』と「-on dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

カタログの変更

カタログは、バージョン 10.0.0 で大きく変更されました。最も重要な変更は、システムテーブルの名前が変更され、名前の先頭に I が含まれるようになったことです。システムテーブルにアクセスしようとすると、パーミッション拒否エラーを受け取ります。システムテーブル内の情報は、システムビューから使用できます。システムテーブルあたりシステムビューは 1 つあります。下位互換のためにこのシステムビューの名前は以前のバージョンの SQL Anywhere からのテーブル名と一致します。たとえば 9.0.2 では、SYS.SYSARTICLE と呼ばれるシステムテーブルがありました。バージョン 10.0.0 では、このシステムテーブルは SYS.ISYSARTICLE と呼ばれ、対応するシステムビューは SYS.SYSARTICLE になります。

カタログには、統合ビューも含まれるようになりました。これらのビューは、一般に必要な 2 つ以上のテーブルやビューからのジョインを提供します。統合ビューのほとんどは、以前のリリースでシステムビューとして存在していました。

一部のシステムテーブルやシステムビューは、廃止予定であるか、カタログから削除されました。ただし、ほとんどの互換ビューは提供されていません。

SQL Anywhere 9.0.2 から SQL Anywhere 10.0.0 へのカタログの完全なマッピングを次の表に示します。1 番目のカラム「9.0.2 システムテーブル/ビュー」には、9.0.2 システムテーブルの名前と、スラッシュ (/) を挟んで 9.0.2 関連ビューを示します。中央のカラム「10.0.0 システムテーブル」には、10.0.0 テーブル名が含まれます。最後のカラム「10.0.0 システムビュー」には、関連する 10.0.0 ビュー名と、互換性に関する注意が含まれます。

注意
カラム内のダッシュ (-) は、等価オブジェクトがないことを示します。たとえば 10.0.0 リリースのカタログで新しいテーブルは、9.0.2 カラムのテーブルの位置はダッシュになります。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
DUMMY / -	DUMMY	-
RowGenerator / -	RowGenerator	-
SYSARTICLE / SYSARTICLES	ISYSARTICLE	「SYSARTICLE システムビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合：「SYSARTICLES 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSARTICLECOL / SYSARTICLECOL	ISYSARTICLECOL	「SYSARTICLECOL システムビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合：「SYSARTICLECOLS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSATTRIBUTE / -	ISYSATTRIBUTE	-
SYSATTRIBUTENAME / -	ISYSATTRIBUTENAME	-
SYSCAPABILITY / SYSCAPABILITIES	ISYSCAPABILITY	「SYSCAPABILITY システムビュー」『SQL Anywhere サーバー SQL リファレンス』 「SYSCAPABILITIES 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSCAPABILITYNAME / -	ISYSCAPABILITYNAME	「SYSCAPABILITYNAME システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSCATALOG		「SYSCATALOG 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSCHECK / -	ISYSCHECK	「SYSCHECK システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSCOLAUTH	-	「SYSCOLAUTH 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSCOLLATION / -	-	「SYSCOLLATION 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
SYSCOLLATIONMAPPINGS / -	-	「SYSCOLLATIONMAPPINGS 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSCOLPERM / -	ISYSCOLPERM	「SYSCOLPERM システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSCOLSTAT / SYSCOLSTATS	ISYSCOLSTAT	「SYSCOLSTAT システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSCOLSTATS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSCOLUMN / SYSCOLUMNS	ISYSTABCOL	「SYSTABCOL システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSCOLUMNS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合： 「SYSCOLUMN 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSCONSTRAINT / -	ISYSCONSTRAINT	「SYSCONSTRAINT システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / -	ISYSDEPENDENCY	「SYSDEPENDENCY システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSDOMAIN / -	ISYSDOMAIN	「SYSDOMAIN システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSEVENT / -	ISYSEVENT	「SYSEVENT システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSEVENTTYPE / -	ISYSEVENTTYPE	「SYSEVENTTYPE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSEXTENT / -	-	-
SYSEXTERNLOGINS / -	ISYSEXTERNLOGIN	「SYSEXTERNLOGIN システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSFILE / -	ISYSFILE	「SYSFILE 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
SYSFKCOL / -	ISYSIDXCOL	「SYSIDXCOL システムビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合：「SYSFKCOL 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSFOREIGNKEY / SYSFOREIGNKEYS	ISYSFKEY	「SYSFKEY システムビュー」『SQL Anywhere サーバー SQL リファレンス』と「SYSFOREIGNKEYS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合：「SYSFOREIGNKEY 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
- / SYSGROUPS	ISYSGROUP	「SYSGROUP システムビュー」『SQL Anywhere サーバー SQL リファレンス』と「SYSGROUPS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSHISTORY / -	ISYSHISTORY	「SYSHISTORY システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSINDEX / SYSINDEXES	ISYSIDX	「SYSIDX システムビュー」『SQL Anywhere サーバー SQL リファレンス』と「SYSINDEXES 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合：「SYSINDEX 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSINFO / -	-	「SYSINFO 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSIXCOL / -	ISYSIDXCOL	「SYSIDXCOL システムビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合：「SYSIXCOL 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
SYSJAR / -	ISYSJAR	「SYSJAR システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSJARCOMPONENT / -	ISYSJARCOMPONENT	「SYSJARCOMPONENT システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSJAVACLASS / -	ISYSJAVACLASS	「SYSJAVACLASS システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSLOGIN / -	ISYSLOGINMAP	「SYSLOGINMAP システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSOPTBLOCK / -	-	システムでのみ使用
- / -	ISYSMVOPTION	「SYSMVOPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / -	ISYSMVOPTIONNAME	「SYSMVOPTIONNAME システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / -	ISYSOBJECT	「SYSOBJECT システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSOPTION / SYSOPTIONS	ISYSOPTION	「SYSOPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSOPTIONS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSOPTJOINSTRATEGY / SYSOPTJOINSTRATEGIES	-	システムでのみ使用
SYSOPTORDER / SYSOPTORDERS	-	システムでのみ使用
SYSOPTQUANTIFIER / -	-	システムでのみ使用
SYSOPTREQUEST / -	-	システムでのみ使用
SYSOPTREWRITE / -	-	システムでのみ使用

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
SYSOPTSTAT / -	ISYSOPTSTAT	「SYSOPTSTAT システムビュー」『SQL Anywhere サーバー SQL リファレンス』
-	ISYSPHYSIDX	「SYSPHYSIDX システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSPROCAUTH	-	「SYSPROCAUTH 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSPROCEDURE / SYSPROCEDURES	ISYSPROCEDURE	「SYSPROCEDURE システムビュー」『SQL Anywhere サーバー SQL リファレンス』 SYSPROCEDURES ビューは、SYSPROCS に名前が変更されました。「SYSPROCS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
SYSPROCPARM / SYSPROCPARMS	ISYSPROCPARM	「SYSPROCPARM システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSPROCPARMS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSPROCPERM / -	ISYSPROCPERM	「SYSPROCPERM システムビュー」『SQL Anywhere サーバー SQL リファレンス』
-	ISYSPROXYTAB	「SYSPROXYTAB システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSPUBLICATION / SYSPUBLICATIONS	ISYSPUBLICATION	「SYSPUBLICATION システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSPUBLICATIONS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
- / -	ISYSREMARK	「SYSREMARK システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSREMOTEOPTION / SYSREMOTEOPTIONS , SYSREMOTEOPTION2	ISYSREMOTEOPTION	「SYSREMOTEOPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』、「SYSREMOTEOPTION2 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』、「SYSREMOTETYPES 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
SYSREMOTEOPTIONTYPE / -	ISYSREMOTEOPTIONTYPE	「SYSREMOTEOPTIONTYPE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSREMOTETYPE / SYSREMOTETYPES	ISYSREMOTETYPE	「SYSREMOTETYPE システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と「SYSREMOTETYPES 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSREMOTEUSER / SYSREMOTEUSERS	ISYSREMOTEUSER	「SYSREMOTEUSER システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と「SYSREMOTEUSERS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSSCHEDULE / -	ISYSSCHEDULE	「SYSSCHEDULE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSSERVERS / -	ISYSSERVER	「SYSSERVER システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / -	ISYSSOURCE	「SYSSOURCE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSSQLSERVERTYPE / -	ISYSSQLSERVERTYPE	「SYSSQLSERVERTYPE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSSUBSCRIPTION / SYSSUBSCRIPTIONS	ISYSSUBSCRIPTION	「SYSSUBSCRIPTION システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と「SYSSUBSCRIPTIONS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSSYNC / SYSSYNCS, SYSSYNC2	ISYSSYNC	「SYSSYNC システムビュー」『SQL Anywhere サーバー SQL リファレンス』、「SYSSYNCS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』、「SYSSYNC2 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
-	ISYSSYNCSCRIPT	「SYSSYNCSCRIPT システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSSYNCSCRIPTS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSSYNCSUBSCRIPTIONS	-	「SYSSYNCSUBSCRIPTIONS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSSYNCUSERS	-	「SYSSYNCUSERS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSTABAUTH	-	「SYSTABAUTH 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSTABLE / -	ISYSTAB	「SYSTAB システムビュー」『SQL Anywhere サーバー SQL リファレンス』 10.0.0 以前と互換性を持たせる場合： 「SYSTABLE 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
-	ISYSTABCOL	「SYSTABCOL システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSTABLEPERM / -	ISYSTABLEPERM	「SYSTABLEPERM システムビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSTRIGGER / SYSTRIGGERS	ISYSTRIGGER	「SYSTRIGGER システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSTRIGGERS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSTYPEMAP / -	ISYSTYPEMAP	「SYSTYPEMAP システムビュー」『SQL Anywhere サーバー SQL リファレンス』
-	ISYSUSER	「SYSUSER システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSUSERAUTH	ISYSUSERAUTHORITY	「SYSUSERAUTHORITY システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSUSERAUTH 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』

9.0.2 システムテーブル/ビュー	10.0.0 システムテーブル	10.0.0 システムビュー
- / SYSUSERLIST		「SYSUSERAUTHORITY システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSUSERLIST 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSUSERMESSAGES / -	ISYSUSERMESSAGE	「SYSUSERMESSAGE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSUSEROPTIONS	-	「SYSUSEROPTIONS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSUSERPERM / SYSUSERPERMS	-	データは ISYSUSER と ISYSUSERAUTHORITY システムテーブルに置かれるようになりました。「SYSUSER システムビュー」『SQL Anywhere サーバー SQL リファレンス』と 「SYSUSERAUTHORITY システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。 10.0.0 以前と互換性を持たせる場合：「SYSUSERPERM 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』と 「SYSUSERPERMS 互換ビュー (旧式)」『SQL Anywhere サーバー SQL リファレンス』
SYSUSERTYPE / -	ISYSUSERTYPE	「SYSUSERTYPE システムビュー」『SQL Anywhere サーバー SQL リファレンス』
- / SYSVIEWS	ISYSVIEW	「SYSVIEW システムビュー」『SQL Anywhere サーバー SQL リファレンス』 と 「SYSVIEWS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
SYSWEBSERVICE / -	ISYSWEBSERVICE	「SYSWEBSERVICE システムビュー」『SQL Anywhere サーバー SQL リファレンス』

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

新しいビューの一覧

システムビュー名	詳細な情報
SYSDEPENDENCY	SYSDEPENDENCY システムビューの各ローは、2 つのデータベースオブジェクト間の依存性を示します。
SYSFKEY	SYSFKEY システムビューの各ローは、システム内の外部キー制約を示します。
SYSIDX	SYSIDX システムテーブルの各ローは、データベースの論理インデックスを定義します。
SYSIDXCOL	SYSIDXCOL システムビューの各ローは、SYSIDX システムビューで記述されているインデックスのカラム 1 つを示します。
SYSLOGINMAP	SYSLOGINMAP システムビューには、統合化ログインまたは Kerberos ログインを使用したデータベースへの接続に使用できるすべてのユーザー名が含まれます。
SYSMVOPTION	SYSMVOPTION システムビューの各ローは、マテリアライズドビューのオプション値 1 つの設定を示します。
SYSMVOPTIONNAME	SYSMVOPTIONNAME システムビューの各ローには、SYSMVOPTION システムビューで定義されているオプションの名前が含まれます。
SYSOBJECT	SYSOBJECT システムビューの各ローは、オブジェクトを示します。データベースオブジェクトの例として、テーブル、ビュー、カラム、インデックス、プロシージャーなどがあります。
SYSPHYSIDX	SYSPHYSIDX システムビューの各ローは、データベースの物理インデックスを定義します。
SYSPROCS	SYSPROCS システムビューは、以前の SYSPROCEDURES ビューを置き換えます。
SYSPROXYTAB	SYSPROXYTAB システムビューの各ローは、プロキシテーブル 1 つのリモートパラメーターを示します。
SYSREMARK	SYSREMARK システムビューの各ローは、オブジェクトの注釈 (またはコメント) を示します。
SYSSOURCE	SYSSOURCE システムビューの各ローには、ISYSOBJECT システムテーブルにリストされているオブジェクトのソースが含まれます。
SYSSYNCSCRIPT	SYSSYNCSCRIPT システムビューの各ローは、Mobile Link のスクリプト化されたアップロードに関するストアドプロシージャーを識別します。

システムビュー名	詳細な情報
SYSTABCOL	SYSTABCOL システムビューには、データベースの各テーブルとビューの各カラムのローが含まれます。
SYSUSER	SYSUSER システムビューの各ローは、データベース内のユーザーを示します。
SYSUSERAUTHORITY	SYSUSERAUTHORITY システムビューの各ローは、ユーザー ID に付与された権限を示します。

廃止予定のテーブルやビューの一覧

廃止予定のカタログオブジェクトを次に示します。通常、オブジェクトは以前のバージョンのテーブルでしたが、互換ビューになりました。これらのオブジェクトを参照してもエラーにはなりませんが、今後の互換性のために、推奨されるオブジェクトを指すようにアプリケーションを変更することをおすすめします。

廃止予定のテーブルやビュー	変換情報
SYSCOLLATION システムテーブル	照合マッピング情報はデータベースプロパティとして格納されるようになりました。
SYSCOLLATIONMAPPINGS システムテーブル	照合マッピング情報はデータベースプロパティとして格納されるようになりました。
SYSCOLUMN システムテーブル	代わりに SYSTABCOL システムビューを使用してください。
SYSFKCOL システムテーブル	代わりに SYSFKEY システムビューを使用してください。
SYSFOREIGNKEY システムテーブル	代わりに SYSFKEY システムビューを使用してください。
SYSINDEX システムテーブル	代わりに SYSIDX システムビューを使用してください。
SYSIXCOL システムテーブル	代わりに SYSIDXCOL システムビューを使用してください。
SYSTABLE システムテーブル	代わりに SYSTAB システムビューを使用してください。
SYSUSERAUTH システムビュー	代わりに SYSUSERAUTHORITY システムビューを使用してください。
SYSUSERPERM システムテーブル	代わりに SYSUSERAUTHORITY システムビューを使用してください。
SYSUSERLIST システムビュー	代わりに SYSUSERAUTHORITY システムビューを使用してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

廃止予定のテーブルやビュー	変換情報
SYSUSERPERMS システムビュー	代わりに SYSUSERAUTHORITY システムビューを使用してください。

削除または名前が変更されたテーブルやビューの一覧

カタログに存在しなくなったカタログオブジェクトを次に示します。これらのオブジェクトを参照すると、エラーになります。

削除されたテーブルやビュー	変換情報
SYSATTRIBUTE システムテーブル	代わりに SYSTAB と SYSPHYSIDX システムビューを使用してください。空いている割合とクラスタードインデックスの情報は、ISYSTAB システムテーブルで保管されるようになりました。キーの値、キーの距離、リーフページ、深さの情報は、ISYSPHYSIDX システムテーブルで保管されるようになりました。
SYSATTRIBUTENAME システムテーブル	代わりに SYSIDX と SYSPHYSIDX システムビューを使用してください。
SYSEXTENT システムテーブル	SQL Anywhere バージョン 10.0.0 以降のカタログでは、SYSEXTENT テーブルを使用できなくなりました。このテーブルは、以前のバージョンでは未使用でした。
SYSEXTERNLOGINS	SYSEXTERNLOGIN に名前が変更されました。
SYSLOGIN システムテーブル	SYSLOGIN テーブルは、一部変更され、SYSLOGINMAP システムビューに置き換えられました。
SYSOPTBLOCK	このテーブルは内部でのみ使用されていました。
SYSOPTJOINSTRATEGY	このテーブルは内部でのみ使用されていました。
SYSOPTJOINSTRATEGIES	このビューは内部でのみ使用されていました。
SYSOPTORDER	このテーブルは内部でのみ使用されていました。
SYSOPTORDERS	このビューは内部でのみ使用されていました。
SYSOPTQUANTIFIER	このテーブルは内部でのみ使用されていました。
SYSOPTREQUEST	このテーブルは内部でのみ使用されていました。
SYSOPTREWRITE	このテーブルは内部でのみ使用されていました。
SYSPROCEDURES ビュー	代わりに SYSPROCS 統合化ビューを使用してください。

削除されたテーブルやビュー	変換情報
SYSSERVERS	SYSSERVER に名前が変更されました。
SYSUSERMESSAGES	SYSUSERMESSAGE に名前が変更されました。

システムテーブルとシステムビューのカラムの変更

システムテーブルとシステムビューのカラムにはさまざまな変更が加えられました。後述する変更以外は、新しいカラムの追加か未使用のカラムの削除で、どちらもアプリケーションへの影響はありません。

- **SYSCOLUMN と SYSCOLUMNS ビュー**　これらの両方のビューの width カラムが、SMALLINT から UNSIGNED INT に変更されました。
- **SYSCONSTRAINT ビュー**　以前の SYSCONSTRAINT システムテーブルは、新しいシステムテーブル ISYSCONSTRAINT と、対応する SYSCONSTRAINT システムビューで置き換えられました。SYSCONSTRAINT の参照で、新しいシステムビューが使用されるようになりました。これが今回のリリースで大きく異なる点です。
- **SYSREMOTEOPTION ビュー**　SYSREMOTEOPTION から選択できなくなりました。代わりに SYSREMOTEOPTIONS または SYSREMOTEOPTION2 を使用してください。
- **SYSJAR、SYSJARCOMPONENT、SYSJAVACLASS ビュー**　create_time カラムは削除されました。ただし作成時刻の情報は、SYSOBJECT.create_time で使用できます。
- **SYSFILE システムビュー**　store_type カラムは INTEGER になりました。
- **SYSPROCPARM と SYSLOGINMAP ビュー**　これらのビューから remarks カラムが削除されました。また、SYSPROCPARM の width カラムが、SMALLINT から UNSIGNED INT に変更されました。
- **SYSPROCPARMS ビュー**　SYSPROCPARM.width が、SMALLINT から UNSIGNED INT に変更されました。
- **SYSREMOTEUSER ビュー**　log_send、log_sent、confirm_sent、log_received、confirm_received の各カラムは UNSIGNED BIGINT になりました。
- **SYSSUBSCRIPTION ビュー**　created と started カラムは UNSIGNED BIGINT になりました。
- **SYSSYNC ビュー**　progress、created、log_sent カラムは UNSIGNED BIGINT になりました。

SQL 文

- **REVOKE CONNECT 文**　ユーザーを削除するために REVOKE CONNECT 文を実行すると、指定したユーザーが所有するオブジェクトがすべてユーザーとともに削除されます。データベースに、削除対象のユーザーが所有するオブジェクトに依存する、別のユーザーが所有するアクティブなビューが含まれる場合、REVOKE CONNECT 文はエラーを返します。「REVOKE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **派生テーブルのキージョインの制限**　TOP N、START AT、FIRST、ORDER BY、Window 関数、FOR XML、または再帰テーブルを含む派生テーブルでキージョインを行えなくなりました。「ビューと派生テーブルのキージョイン」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ALTER SERVER 文と CREATE SERVER 文**　サーバークラス ASAJDBC と ASAODBC は、それぞれ SAJDBC と SAODBC に名前が変更されました。「ALTER SERVER 文」『SQL Anywhere サーバー SQL リファレンス』と「CREATE SERVER 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **ALTER 文**　すべての ALTER 文で、サブ句として MODIFY ではなく、ALTER を使用するようになりました。アプリケーションで MODIFY サブ句を使用している場合は、代わりに ALTER サブ句を使用するように変更する必要があります。MODIFY 構文は、サポートはされますが、廃止される予定です。この影響は次の文に及びます。

  - ALTER DATABASE 文：「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER EVENT 文：「ALTER EVENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER PUBLICATION 文 [Mobile Link] [SQL Remote]：「ALTER PUBLICATION 文 [Mobile Link] [SQL Remote]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]：「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER SYNCHRONIZATION USER 文 [Mobile Link]：「ALTER SYNCHRONIZATION USER 文 [Mobile Link]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。
  - ALTER TABLE 文：「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **BACKUP 文**　以前のリリースでは、TRANSACTION LOG RENAME 句または TRANSACTION LOG TRUNCATE 句で、DBFILE ONLY 句を指定できました。今回のバージョンでは、バックアップには相互に排他的な 2 つの種類があるため、DBFILE ONLY をどちらかの TRANSACTION LOG 句で指定すると、エラーが発生するようになりました。「BACKUP 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **COMMENT 文**　構文 COMMENT ON LOGIN はサポートされなくなりました。代わりに構文 COMMENT ON INTEGRATED LOGIN を使用してください。「COMMENT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **INSERT 文**　SQL Anywhere 10 で ON EXISTING SKIP 句と ON EXISTING ERROR 句を使用するときに、テーブルにデフォルトのカラムが含まれる場合、サーバーでは、すでに存在するローに対してもデフォルト値が計算されます。結果として、AUTOINCREMENT のようなデフォルト値が、スキップされたローに対しても影響を及ぼします。AUTOINCREMENT の場合は、AUTOINCREMENT のシーケンスで値がスキップされます。以前のバージョンでは、スキップされたローのデフォルトのカラムに対してこれらの計算は行われませんでした。「INSERT 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **VALIDATE 文**　すべての検証アクティビティ (VALIDATE 文の実行、検証ユーティリティ (dbvalid) の実行など) で、VALIDATE 権限が必要になりました。また、REMOTE DBA パーミッションで検証アクティビティを実行できなくなりました。

  VALIDATE TABLE 文 (および VALIDATE MATERIALIZED VIEW) は、孤立した BLOB を検査します。

  VALIDATE INDEX 文の構文が変更され、ALTER INDEX 文の構文と一致するようになりました。以前の構文は、サポートはされますが、廃止される予定です。アプリケーションで VALIDATE INDEX 文を使用している場合は、新しい構文に変更する必要があります。

  これらの変更の詳細については、「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

廃止予定機能とサポート終了機能

- **検証ユーティリティ (dbvalid) の -f、-fi、-fd、-fn オプションは廃止予定**　dbvalid ユーティリティの構文が簡略化されました。以前は、オプションを指定しないと、テーブルの検証時にエクスプレス検証が実行されていました。今回のバージョンでは、-f、-fi-、-fd オプションを指定したかのように、デフォルトでフル検証が実行されるようになりました。そのため、これらのオプションの使用は廃止される予定です。テーブルでエクスプレス検証を実行する場合は -fx オプションを指定する必要があります。

  また、-fn オプション (バージョン 9.0.0 以前のリリースからのアルゴリズムで検証を実行する) はサポートされなくなりました。

  検証ユーティリティの詳細については、「検証ユーティリティ (dbvalid)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **VALIDATE TABLE 文のオプションは廃止予定**　VALIDATE TABLE 文の構文が簡略化されました。以前は、オプションを指定しないと、通常の検証が実行されていました。今回のバージョンでは、WITH FULL CHECK オプションを指定したかのように、デフォルトでフル検証が実行されるようになりました。そのため、WITH FULL CHECK、WITH INDEX、WITH data オプションは廃止される予定です。「VALIDATE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **Transact-SQL 外部ジョインは廃止予定**　今回のリリースで Transact-SQL 外部ジョインは廃止され、SQL Anywhere の将来のバージョンではサポートされなくなる予定です。新しい tsql_outer_joins データベースオプションを使用すると、現在の接続の DML 文とビューで Transact-SQL 外部ジョイン演算子 *= と =* の使用を有効／無効にできます。このオプションのデフォルト設定は Off です。「tsql_outer_joins オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **WITH HASH SIZE 句のサポート終了**　B ツリーインデックステクノロジが削除されたことで、CREATE INDEX 文の WITH HASH SIZE 句はサポートされなくなりました。

- **サポート対象外のプロパティ**　NumProcessorsAvail と NumProcessorsMax サーバープロパティはサポートされなくなりました。代わりに、NumLogicalProcessors、

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

NumLogicalProcessorsUsed、NumPhysicalProcessors、NumPhysicalProcessorsUsed の各サーバープロパティを使用できます。「データベースサーバープロパティ値のアクセス」『SQL Anywhere サーバー データベース管理』を参照してください。

- **LOAD TABLE の STRIP ON 句は廃止予定**　前後のブランクを削除する機能は SQL Anywhere 10.0.0 で強化され、削除機能を微調整できるようになり、STRIP ON は廃止される予定です。今後も後続ブランクだけを削除する場合は、代わりに STRIP RTRIM を使用してください。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **UTF8 照合は廃止予定**　UTF8 照合は廃止される予定です。代りに UTF8BIN 照合を使用してください。「サポートされている照合と代替照合」『SQL Anywhere サーバー データベース管理』を参照してください。

- **jConnect 4.5 のサポート終了**　jConnect 4.5 を使用して接続していたアプリケーションは動作しますが、jConnect 5.5 または 6.0.5 を使用することをおすすめします。「jConnect JDBC ドライバー」『SQL Anywhere サーバー プログラミング』を参照してください。

- **SQLLOCALE 環境変数のサポート終了**　SQLLOCALE 環境変数はサポートされなくなりました。SALANG と SACHARSET 環境変数に置き換わっています。「SALANG 環境変数」『SQL Anywhere サーバー データベース管理』と「SACHARSET 環境変数」『SQL Anywhere サーバー データベース管理』を参照してください。

- **名前付きパイプのサポート終了**　名前付きパイプはサポートされなくなりました。名前付きパイプを使用していたアプリケーションは、共有メモリを使用するように変更する必要があります。「通信プロトコルオプションの考慮事項」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データソース ユーティリティ (dbdsn) の -o オプションは廃止予定**　データソースユーティリティの -o オプションは廃止される予定です。出力メッセージをファイルに書き込む場合は、接続文字列で LogFile 接続パラメーターを指定できます。「LogFile (LOG) 接続パラメーター」『SQL Anywhere サーバー データベース管理』を参照してください。

- **カスタム照合の作成はサポート対象外**　カスタム照合の作成はサポートされなくなりました。カスタム照合作成ウィザード、照合ユーティリティ (dbcollat)、DBCollate 関数、a_db_collation 構造体はサポートされなくなりました。「照合の考慮事項」『SQL Anywhere サーバー データベース管理』を参照してください。

  カスタム照合を使用してデータベースを再構築する場合に 1 ステップで再構築すると、その照合は保存されます。データベースをアンロードしてから、作成したデータベースにスキーマとデータをロードする場合は、提供されるいずれかの照合を使用する必要があります。「バージョン 9 以前のデータベースのバージョン 12 用への再構築」 373 ページを参照してください。

- **サーバーライセンス取得ユーティリティの -p オプションはサポート対象外**　以前のリリースでは、サーバーライセンス取得ユーティリティで -p オプション (データベースサーバーのライセンスが取得されているオペレーティングシステムを指定する) がサポートされていました。このオプションはサポートされなくなりました。

- **データベースサーバーの -d オプションはサポート対象外**　データベースサーバーオプション -d (NetWare で DFS (Direct File System) I/O ではなく POSIX I/O を使用する) はサポートされなくなりました。

- **データベースサーバーの -y オプションはサポート対象外**　Windows 95/98/Me がサポートされなくなったため、データベースサーバーオプション -y (Windows 95/98/Me で Windows サービスとしてデータベースサーバーを実行する) は、サポートされなくなりました。サポートされるプラットフォームでデータベースサーバーをサービスとして実行するには、dbsvc ユーティリティを使用してください。「Windows 用サービスユーティリティ (dbsvc)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **-sc オプションはサポート対象外**　SQL Anywhere 7.0 は、米国政府から TCSEC (Trusted Computer System Evaluation Criteria) の C2 セキュリティ評価を授与されています。-sc サーバーオプションを使用すると、SQL Anywhere の現在のバージョンを、C2 基準を満たした環境と同等の方法で実行できます。-sc オプションと C2 サーバープロパティのサポートは、バージョン 10.0.0 で削除されました。

- **max_work_table_hash_size データベースオプションはサポート対象外**　max_work_table_hash_size オプションは、サポートされなくなりました。クエリオプティマイザーは、テーブル内のデータ分散を基に、内部テンポラリテーブルのハッシュサイズを割り当てます。

- **max_hash_size database オプションはサポート対象外**　max_hash_size オプションは、サポートされなくなりました。

- **圧縮データベースとライトファイルはサポート対象外**　その結果、次の機能は使用できなくなりました。

  - **ファイル拡張子**　次のファイル拡張子は、サポートされなくなりました。

    - ライトファイルを識別するための *.wrt* 拡張子

    - 圧縮データベースファイルを識別するための *.cdb* 拡張子

  - **NetWare 上のデータベースサーバーの動作**　データベースサーバーは、拡張子なしでデータベースファイルが指定されたときに、*.wrt* 拡張子を持つデータベースファイルを検索しなくなりました。「SQL Anywhere データベースサーバーの構文」『SQL Anywhere サーバー データベース管理』を参照してください。

  - **読み込み専用メディアでのデータベースの展開**　読み込み専用メディア (CD-ROM など) で提供されるデータベースへの変更を記録するために、ライトファイルを指定できなくなりました。ただし、読み込み専用モードでデータベースが実行されている場合に、読み込み専用メディアのデータベースを展開することはできます。「読み込み専用メディアでのデータベースの配備」『SQL Anywhere サーバー プログラミング』と「-r dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○ **データベースユーティリティ**　次のユーティリティやウィザードは、サポートされなくなりました。

- データベース圧縮ウィザード
- ライトファイル作成ウィザード
- データベース展開ウィザード
- 展開ユーティリティ (dbexpand)
- 圧縮ユーティリティ (dbshrink)
- ライトファイルユーティリティ (dbwrite)

○ **SQL 文**　次の SQL 文は、サポートされなくなりました。

- ALTER WRITEFILE
- CREATE WRITEFILE
- CREATE COMPRESSED DATABASE
- CREATE EXPANDED DATABASE

○ **DBTools 構造体**　次の構造体や構造体のメンバーはサポートされなくなりました。

- **a_backup_db 構造体**　この構造体は、DBTools ライブラリを使用してバックアップタスクを実行するために必要な情報を格納します。

  backup_writefile メンバーは _unused と示されるようになりました。

- **a_compress_db 構造体**　この構造体は削除されました。

- **a_compress_stats 構造体**　この構造体は、DBTools ライブラリを使用してデータベース圧縮タスクを実行するために必要な情報を格納します。

- **a_db_info 構造体**　この構造体は、DBTools ライブラリを使用して dbinfo 情報を戻すために必要な情報を格納します。

  wrtbufsize メンバーは _unused1 として、wrtnamebuffer メンバーは _unused2 として、compressed メンバーは _unused3 として示されるようになりました。

- **an_expand_db 構造体**　この構造体は、DBTools ライブラリを使用してデータベースを拡張するために必要な情報を格納します。

- **a_stats_line 構造体**　この構造体は、DBTools ライブラリを使用してデータベースを圧縮および拡張するために必要な情報を格納します。

- **a_writefile 構造体**　この構造体は、DBTools ライブラリを使用してデータベースライトファイルを管理するために必要な情報を格納します。

○ **DBTools 関数**　次の関数は、サポートされなくなりました。

- DBChangeWriteFile
- DBCompress
- DBCreateWriteFile
- DBExpand
- DBStatusWriteFile

○ **データベースプロパティ**　次のデータベースプロパティは、サポートされなくなりました。

●Compression

●FileSize *writefile*

●FreePages *writefile*

○ **DB_BACKUP_WRITEFILE**　この Embedded SQL 関数はサポートされなくなりました。

● **未使用の Adaptive Server Enterprise 互換ビューとプロシージャーのサポート終了**　SQL Anywhere データベースで次の未使用の Adaptive Server Enterprise ビューサポートは削除されました。

ビュー名	ビュー名
SYSALTERNATES	SYSLOGINROLES
SYSAUDITOPTIONS	SYSLOGS
SYSAUDITS	SYSMESSAGES
SYSCHARSETS	SYSPROCEDURES
SYSCONFIGURES	SYSPROCESSES
SYSCONSTRAINTS	SYSPROTECTS
SYSCURCONFIGS	SYSREFERENCES
SYSDATABASES	SYSREMOTELOGINS
SYSDEPENDS	SYSROLES
SYSDEVICES	SYSSEGMENTS
SYSENGINES	SYSSERVERS
SYSKEYS	SYSSRVROLES
SYSLANGUAGES	SYSTHRESHOLDS
SYSLOCKS	SYSUSAGES

SQL Anywhere データベースで次の未使用の Adaptive Server Enterprise プロシージャーサポートは削除されました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

プロシージャー名	プロシージャー名
sp_addalias	sp_helpindex
sp_addauditrecord	sp_helpjoins
sp_addlanguage	sp_helpkey
sp_addremotelogin	sp_helplanguage
sp_addsegment	sp_helplog
sp_addserver	sp_helpremotelogin
sp_addthreshold	sp_helpprotect
sp_adddumpdevice	sp_helpsegment
sp_auditdatabase	sp_helpserver
sp_auditlogin	sp_helpsort
sp_auditobject	sp_helpthreshold
sp_auditoption	sp_helpuser
sp_auditsproc	sp_indsuspect
sp_bindefault	sp_lock
sp_bindmsg	sp_locklogin
sp_bindrule	sp_logdevice
sp_changedbowner	sp_modifylogin
sp_checknames	sp_modifythreshold
sp_checkreswords	sp_monitor
sp_clearstats	sp_placeobject
sp_commonkey	sp_primarykey
sp_configure	sp_procxmode
sp_cursorinfo	sp_recompile
sp_dboption	sp_remap

プロシージャー名	プロシージャー名
sp_dbremap	sp_remoteoption
sp_depends	sp_rename
sp_diskdefault	sp_renamedb
sp_displaylogin	sp_reportstats
sp_dropalias	sp_role
sp_dropdevice	sp_serveroption
sp_dropkey	sp_setlangalias
sp_droplanguage	sp_spaceused
sp_dropremotelogin	sp_syntax
sp_dropsegment	sp_unbindefault
sp_dropserver	sp_unbindmsg
sp_dropthreshold	sp_unbindrule
sp_estspace	sp_volchanged
sp_extendsegment	sp_who
sp_foreignkey	sp_column_privileges
sp_help	sp_databases
sp_helpconstraint	sp_datatype_info
sp_helpdb	sp_server_info
sp_helpdevice	sp_table_privileges
sp_helpgroup	

- **SYSINDEX システムビューからの index_type と index_owner カラムの削除**　index_type と index_owner カラムは SYSINDEX ビューから削除されました。これらのカラムは、それぞれデフォルト値 USER と SA が格納されていました。インデックス情報は、ISYSIDX と ISYSIDXCOL システムビューに格納されるようになりました。「SYSIDX システムビュー」『SQL Anywhere サーバー SQL リファレンス』と「SYSIDXCOL システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **サーバーでの DLL プロトコルオプションの削除**　DLL プロトコルオプションは、Windows 32 ビットプラットフォームで実行しているクライアントだけに適用されるようになりました。データベースサーバーでは、DLL プロトコルオプションが削除され、Winsock 2.2 だけを使用します。同様に、Windows CE クライアントでは、DLL プロトコルオプションが削除され、Winsock 1.1 だけを使用します。

  Winsock 2.2 は、すべての Windows プラットフォーム上のデータベースサーバーで必要です。

- **ASANY と ASANYSH 環境変数名の変更**　ASANY と ASANYSH 環境変数は、それぞれ SQLANY10 と SQLANYSH10 に名前が変更されました。「SQLANY12 環境変数」『SQL Anywhere サーバー データベース管理』と「SQLANYSAMP12 環境変数」『SQL Anywhere サーバー データベース管理』を参照してください。

- **PreserveSource プロパティは廃止予定**　今回のリリースで PreserveSource データベースプロパティが廃止され、この設定が問い合わされると常に値 On を返すようになりました。

- **サポート対象外のデータベースプロパティ**　今回のリリースでは、次のデータベースプロパティが削除されました。

  - BlobArenas
  - ClusteredIndexes
  - CompressedBTrees
  - FileVersion
  - FreePageBitMaps
  - Histograms
  - HistogramHashFix
  - IndexStatistics
  - LargeProcedureIDs
  - NamedConstraints
  - SeparateCheckpointLog
  - SeparateForeignKeys
  - StringHistogramsFix
  - TableBitMaps
  - TablesQualTriggers
  - TransactionsSpanLogs
  - UniqueIdentifier
  - VariableHashSize

- **サポート対象外のシステムプロシージャー**　sa_conn_properties_by_name と sa_conn_properties_by_conn システムプロシージャーはサポートされなくなりました。新しい sa_conn_options システムプロシージャーを使用すると、同じ情報を取得できます。「sa_conn_options システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **クエリ最適化プランから削除されたアルゴリズム**　ロック、ネストブロックジョイン、ソートブロック、JNBO の各アルゴリズムは、クエリ最適化プランから削除されました。また、ロックノードはこのプランに表示されなくなりました。ロック情報は、プラン内のスキャンノードで確認できます。

- **util_db.ini ファイルは廃止予定**　ユーティリティデータベースへの接続時に、*util_db.ini* ファイルを使用して DBA ユーザーのパスワードを指定する機能は廃止される予定です。代わりに -su サーバーオプションを使用できます。「ネットワークデータベースサーバーでの util_db.ini の使用 (旧式)」『SQL Anywhere サーバー データベース管理』と「-su dbeng12/dbsrv12 サーバーオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **廃止予定の Windows CE プラットフォーム**　Windows CE MIPS プロセッサーはサポートされなくなりました。サポートされるプラットフォームのリストについては、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

Mobile Link

次の項では、Mobile Link バージョン 10.0.0 の新機能、動作の変更、廃止予定機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された Mobile Link の追加機能を示します。

Sybase Central の Mobile Link プラグインの強化

「同期モデルファイル」の作成にウィザードを使用することにより、Mobile Link アプリケーションの設定が大幅に簡単になりました。このファイルには、リモートテーブルと統合化テーブルについて入力する情報と、それらを同期する方法についての情報が含まれます。モデルが用意できたら、別のウィザードを使用して展開します。展開されたモデルによってアプリケーションに必要なスクリプトとテーブルが生成されます。

- **同期モデル作成ウィザード**　新しい同期モデル作成ウィザードを使用すると、Mobile Link アプリケーションをすばやく作成して展開できます。このウィザードではリモートデータベースを作成することも、既存のリモートデータベースを使用することもできます。同期スクリプトの作成が自動化されるほか、ダウンロードの削除、競合の解決、その他の難しい同期の問題が自動的に処理されます。

  「[同期モデル作成ウィザード] を使用した Mobile Link アプリケーションの設定」『Mobile Link クイックスタート』を参照してください。

- **モデルモード**　同期モデル作成ウィザードを使用した後は、モデルモードを使用して、同期プロジェクトをカスタマイズしてから展開できます。モデルモードでは、オフラインで作業します。モデルモードでは、同期モデルを XML ファイルとして格納します。

  「同期モデルタスク」『Mobile Link クイックスタート』を参照してください。

- **展開ウィザード**　モデルをカスタマイズしたら、新しい展開ウィザードを使用してモデルを展開できます。展開ウィザードでは、スクリプト、ユーザー、スクリプトのバージョンなどを統合データベース上の Mobile Link システムテーブルに追加します。統合データベースに

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

加えた変更は、モデルモードにリエンジニアリングすることはできませんが、同じモデルを複数回展開できます。

「同期モデルの展開」『Mobile Link クイックスタート』を参照してください。

- **管理モード**　バージョン 10.0.0 以前に存在していた Mobile Link プラグインは、管理モードと呼ばれるようになりました。使い勝手を改善するため、管理モードにはさまざまな強化が行われています。管理モードでは、統合データベースに接続されており、変更はすぐに有効になります。管理モードを使用すると、すべての Mobile Link 統合データベースを変更できます。

任意のデータソースとの同期

「ダイレクトローハンドリング」と呼ばれる新機能を使用すると、ほぼすべてのデータソースと同期できます。たとえば、SQL ベースのローハンドリングを使用して、アプリケーションサーバー、Web サーバー、Web サービス、テキストファイル、Excel スプレッドシート、J2ME デバイス、または統合データベースとして使用できない RDBMS と同期できます。ただし、Mobile Link システムテーブルや、Mobile Link で管理するデータを保持するために、統合データベースは必要なままです。新しいデータソースは、同期処理に完全に統合させることができます。

Mobile Link サーバーの API が拡張され、ダイレクトローハンドリングがサポートされるようになりました。さらに、2 つの新しいイベントが追加されました。次の項を参照してください。

- 「ダイレクトローハンドリング」『Mobile Link サーバー管理』
- 「handle_DownloadData 接続イベント」『Mobile Link サーバー管理』
- 「handle_UploadData 接続イベント」『Mobile Link サーバー管理』

また、「モバイル Web サービス」と呼ばれる新しい機能では、モバイルに最適化された、リモートアプリケーションと統合可能な非同期 Web サービスを提供します。

「モバイル Web サービス」『QAnywhere』を参照してください。

展開の簡略化

Mobile Link サーバー、SQL Anywhere クライアント、Mobile Link モニター、暗号化コンポーネントを展開するための展開ウィザードが使用できるようになりました。以前のバージョンで提供されていた InstallShield マージモジュールとテンプレートは、提供されなくなりました。次の項を参照してください。

- 「Deployment ウィザード」『SQL Anywhere サーバー プログラミング』
- 「Mobile Link アプリケーションの配備」『Mobile Link サーバー管理』

統合データベース

新しい設定プロシージャー

- **SQL Anywhere 統合データベースで必要な設定スクリプト**　Mobile Link 統合データベースとして SQL Anywhere データベースを使用する前に、設定スクリプトを実行する必要があります。また、設定スクリプトで作成された Mobile Link システムテーブルは、設定スクリプトを実行したユーザーが所有者になりました。この動作は、他の種類の統合データベースと同じ

です。以前のバージョンの Mobile Link では、Mobile Link システムテーブルの所有者は SQL Anywhere 統合データベースの DBO でした。

「SQL Anywhere 統合データベース」『Mobile Link サーバー管理』を参照してください。

- **単純化された設定プロシージャー**　統合データベースは、Sybase Central で設定したり、設定スクリプトを使用して設定したりできるようになりました。以前は、設定スクリプトを実行する必要がありました。また、統合データベースの種類ごとに、設定スクリプトは 1 つだけになりました。バージョン固有の設定スクリプト (*syncase125.sql* など) は不要です。

  「Mobile Link 統合データベース」『Mobile Link サーバー管理』を参照してください。

新しいシステムオブジェクト

- **統合データベース上の Mobile Link システムテーブルをクリーンアップする新しい方法**　次の処理を実行できる新しいシステムプロシージャーが追加されました。

  ○Mobile Link システムテーブルからの旧式のリモートデータベースの情報の消去。

  「ml_delete_sync_state_before システムプロシージャー」『Mobile Link サーバー管理』を参照してください。

  ○未使用または不要の同期ステータス情報の削除。

  「ml_delete_sync_state システムプロシージャー」『Mobile Link サーバー管理』を参照してください。

  ○同期ステータス情報のリセット。

  「ml_reset_sync_state システムプロシージャー」『Mobile Link サーバー管理』を参照してください。

- **新しい Mobile Link サーバーシステムテーブルとスキーマ**　Mobile Link システムテーブルは、次のように変更されています。

  ○複数の新しい Mobile Link システムテーブルが追加されました。

  - ml_database
  - ml_column
  - ml_qa_clients

  ○ml_subscription の内容が大幅に変わりました。Ultra Light 同期シーケンス番号は、以前は ml_user.commit_state に格納されていましたが、ml_subscription.progress に格納されるようになりました。progress カラムには、SQL Anywhere リモート同期の進行状況も格納されます。

  ○ml_user の内容が大幅に変わりました。

  ○checksum カラムが ml_script テーブルに追加されました。

  ○ml_listening の ml_user カラムが name カラムに変更されました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○サーバー起動同期のために Sybase Central で内部的に使用される新しいシステムテーブルが追加されました。

○複数の QAnywhere システムテーブルに変更が加えられました。次の項を参照してください。

- ml_qa_delivery
- ml_qa_delivery_client
- ml_qa_global_props
- ml_qa_global_props_client
- ml_qa_repository
- ml_qa_repository_client
- ml_qa_repository_props
- ml_qa_repository_props_client
- ml_qa_repository_staging

- **新しいシステムプロシージャー ml_add_column** 名前付きローパラメーターを使用するときに、場合によっては、この新しいシステムプロシージャーを使用して、ml_column Mobile Link システムテーブルにリモートデータベースのカラムの情報を移植する必要があります。

  「ml_add_column system procedure」『Mobile Link サーバー管理』を参照してください。

Mobile Link サーバー

mlsrv10 の新機能

- **サーバー名を mlsrv10 に変更** Mobile Link サーバーは mlsrv10 という名前になりました。以前は、dbmlsrv9 と呼ばれていました。

- **mlsrv10 -x の新しい構文** 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を設定する mlsrv10 -x オプションが変更されました。

  「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **古いクライアント用の新しい -xo オプション** Mobile Link サーバーをバージョン 8 または 9 クライアントに接続するには、mlsrv10 -xo オプションを使用する必要があります。このオプションは、dbmlsrv9 -x オプションと等価です。mlsrv10 のインスタンス 1 つで、バージョン 10 のクライアントに加えて、バージョン 8 と 9 のクライアントをサポートできます。ただし、それには 2 つの異なるポートを使用する必要があります。

  HTTP と HTTPS の -xo オプションに新しいオプション session_key が含まれます。このオプションは、接続の追跡に JSESSIONID を使用できない場合に便利です。

- **キャッシュ処理の向上** Mobile Link サーバーは、タスクごとに別のメモリプールを管理しなくなりました。すべてのキャッシュメモリはすべての同期で共有されます。キャッシュサイズを設定するには、新しい mlsrv10 -cm オプションを使用します。キャッシュサイズを設定するその他のオプション (-bc、-d、-dd、-u) は削除されました。

  「-cm mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **すべてのストリームに影響するようになった ignore オプション**　無視するように指定したすべてのホスト名または IP アドレスが、すべての -x ストリームで無視されるようになりました。以前は (-xo を使用した場合は今回のバージョンでも)、ホストは指定されたストリームでのみ無視されていました。

  「-x mlsrv12 オプション」『Mobile Link サーバー管理』の "ignore" を参照してください。

- **スクリプトの強制アップロード**　mlsrv10 -zus オプションを使用すると、アップロードするデータがテーブルにない場合でも、Mobile Link サーバーでそのテーブルのアップロードスクリプトを強制的に呼び出すことができます。

  「-zus mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **新しい冗長性オプション**　新しい冗長性オプション **e** を使用すると、システムイベントスクリプトを取得できます。-ve が指定されると、Mobile Link サーバーでは、Mobile Link システムテーブルを管理するためのすべてのシステムイベントスクリプトと、アップロードストリームを定義する SQL 文を表示します。

  「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **ファイル転送ディレクトリ**　ファイル転送用のディレクトリを使用できる新しいオプションが追加されました。

  「-ftr mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **同時同期処理の最大数の設定**　アクティブに作業できる同期処理の最大数を設定することで、パフォーマンスを向上できるようになりました。

  「-sm mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **同時ネットワーク接続の制限**　新しい -nc オプションを使用すると、同時ネットワーク接続の数の上限を指定できます。

  「-nc mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **mlsrv10 がメッセージのタイムスタンプ設定に ISO 8601 日時フォーマットを使用**　情報、警告、エラーの各メッセージのタイムスタンプに、明確に定義された ISO 8601 日時フォーマット ({I|W|E}.yyyy-mm-dd hh:mm:ss message) を使用するようになりました。

Mobile Link の新しいスクリプト機能

- **名前付きスクリプトパラメーター**　Mobile Link イベントパラメーターに名前が付けられました。以前は、疑問符でスクリプトパラメーターを指定する必要がありました。今回のバージョンでは、疑問符はオプションになりました。定義済みの名前付きパラメーターのセットから選択したり、独自の名前付きパラメーターを作成したりすることができます。RDBMS で変数をサポートしない場合は、ユーザー定義の名前付きパラメーターが便利です。疑問符の場合とは異なり、名前付きパラメーターは任意の順序で指定でき、使用できるパラメーターの任意のサブセットを使用できます。また、通常は同じ名前付きパラメーターを同じスクリプト内で複数回使用できます。

  「スクリプトのパラメーター」『Mobile Link サーバー管理』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **新しい競合検出イベント**　カラムレベルで競合を検出するためにスクリプト化できる新しいイベントがあります。このイベントは、ローレベルで競合を検出する upload_fetch イベントの代わりになるものです。

  「upload_fetch_column_conflict テーブルイベント」『Mobile Link サーバー管理』を参照してください。

- **グローバルスクリプトバージョン**　グローバルスクリプトバージョンを作成できるようになりました。グローバルスクリプトバージョンに関連するスクリプトを定義すると、同期に使用しているスクリプトバージョンで同じイベントのスクリプトを指定しない場合は、すべての同期処理でそのスクリプトが自動的に使用されます。つまり、複数のスクリプトバージョンを使用している場合は、接続レベルスクリプトの複製を避けることができるということです。

  「ml_global スクリプトバージョン」『Mobile Link サーバー管理』を参照してください。

パフォーマンスの強化

- **Mobile Link アーキテクチャーの向上**　スループット、柔軟性、保守性を向上させるため、Mobile Link サーバーのアーキテクチャーが再構成されました。同様の理由で、内部の Mobile Link クライアント／サーバープロトコルも強化されました。

  「Mobile Link のパフォーマンス」『Mobile Link サーバー管理』を参照してください。

サーバーのその他の強化

- **スナップショットアイソレーション**　SQL Anywhere バージョン 10 と Microsoft SQL Server 2005 以降の統合データベースでは、スナップショットアイソレーションがダウンロードの場合はデフォルトで、アップロードの場合はオプションになりました。この動作を制御できるように、Mobile Link サーバーのオプションが追加されました。

  次の項を参照してください。

  - 「Mobile Link 独立性レベル」『Mobile Link サーバー管理』
  - 「-dsd mlsrv12 オプション」『Mobile Link サーバー管理』
  - 「-dt mlsrv12 オプション」『Mobile Link サーバー管理』
  - 「-esu mlsrv12 オプション」『Mobile Link サーバー管理』

- **同期 ID**　各同期が 1 ～ 4294967295 の間の整数で識別されるようになりました。Mobile Link サーバーの各インスタンスは、独自の同期 ID を管理します。Mobile Link サーバーが起動すると、ID は 1 にリセットされます。この ID は出力ファイルにロギングされます。

- **Mobile Link ネットワークレイヤーの向上**　ネットワークレイヤーには、圧縮、永続的な接続(同じ接続で複数回の同期が可能)、IPv6 サポートなどが含まれるようになり、エラー検出、活性検出、デバッグが向上しました。

Mobile Link モニターの強化

- **ユーティリティ名を mlmon に変更**　Mobile Link モニターは mlmon という名前になりました。以前は、dbmlmon と呼ばれていました。

「Mobile Link モニターの起動」『Mobile Link サーバー管理』を参照してください。

- **複数の Mobile Link モニター**　複数の Mobile Link モニターを同じ Mobile Link サーバーに同時に接続できるようになりました。これによって、複数のユーザーが同じサーバーでの同期処理を追跡できます。

  「Mobile Link モニターの起動」『Mobile Link サーバー管理』を参照してください。

- **ネットワークオプション**　Mobile Link モニターで、Mobile Link クライアントと同じネットワークオプションを使用できるようになりました。

  「Mobile Link モニターの起動」『Mobile Link サーバー管理』を参照してください。

- **新しい使用率グラフ**　[使用率グラフ] ウィンドウ枠には、Mobile Link サーバー内のキューの長さが表示されます。

  「[使用率グラフ] ウィンドウ枠」『Mobile Link サーバー管理』を参照してください。

- **[チャート] ウィンドウ枠でデータの表示**　[チャート] ウィンドウ枠では、従来どおりユーザーごとのデータを表示することができるほか、コンパクトビューでデータを表示することもできます。コンパクトビューでは、できるだけ少ないローでアクティブな同期処理がすべて表示されます。同期が単一のワーカースレッドに関連付けされなくなったので、ワーカービューは削除されました。

  「[チャート] ウィンドウ枠」『Mobile Link サーバー管理』を参照してください。

- **新しい [サンプルプロパティ] ウィンドウ**　新しい [サンプルプロパティ] には、1 秒間隔のデータ、または選択した期間における 1 秒間隔のすべての平均が表示されます。

  「サンプルプロパティ」『Mobile Link サーバー管理』を参照してください。

- **強化された [セッションプロパティ] ウィンドウ**　セッションプロパティには、詳細な [統計] タブが含まれるようになりました。

  「セッションプロパティ」『Mobile Link サーバー管理』を参照してください。

- **FIPS 認定の暗号化を実行しているサーバーのモニターが可能**　Mobile Link モニターで、FIPS 認定の暗号化を実行している Mobile Link サーバーをモニターできるようになりました。以前は、この機能はありませんでした。

- **統計プロパティの変更**　「Mobile Link サーバーの変更」322 ページの「統計プロパティの変更」を参照してください。

Mobile Link リダイレクターの強化

- **リダイレクターで Mobile Link サーバーのグループをサポート**　一部のリダイレクターで、Mobile Link サーバーグループを作成できるようになりました。サーバーグループは、1 つのリダイレクター経由でバージョン 8 または 9 クライアントと同時にバージョン 10 クライアントをサポートするため、またはその他の目的で使用できます。サーバーグループをサポートするリダイレクターの詳細については、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

サーバーグループをサポートしているリダイレクターは、設定も強化されています。また、*redirector_server_group.config* という名前の新しいサンプル設定ファイルが使用されます。

- **HTTPS のサポート**　以前のバージョンの SQL Anywhere では、リモートデータベースと Web サーバーの間の通信で HTTPS を使用する場合、Web サーバーで HTTPS が復号化されて、HTTP がリダイレクター経由で Mobile Link に送信されます。今回のバージョンでは、一部の Web サーバーで、ストリームがリダイレクターで HTTPS に再暗号化されてから、Mobile Link サーバーに送信されるようになりました。リダイレクターの設定ファイルでは、ML ディレクティブ用の新しい構文があります。HTTPS をサポートする Web サーバーの詳細については、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

UNIX/Linux の強化

- **Mobile Link サーバーメッセージウィンドウ**　Linux のインストール環境には、dbmlsync と mlsrv10 のログ情報を表示するメッセージウィンドウが追加されました。

  「-ux dbmlsync オプション」『Mobile Link クライアント管理』と「-ux mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **より一貫性のある文字変換**　UNIX/Linux と Windows の間の文字変換の一貫性が向上しました。

Mobile Link クライアント

- **新しいリモート ID**　Mobile Link は、リモート ID と呼ばれる新しい識別子を使用して、リモートデータベースをユニークに識別するようになりました。以前のバージョンでは、Mobile Link のユーザー名が使用されていました。リモート ID はリモートデータベースに格納されます。Mobile Link は、リモートデータベースが初めて同期されるとき (または、NULL 値のリモート ID が出現したとき) に、リモート ID を生成します。リモート ID は GUID として自動的に作成されますが、自分にとってわかりやすい文字列に設定することもできます。リモート ID によって、同じ Mobile Link ユーザーが複数のリモートデータベースを同期できるようになります。Ultra Light リモートデータベースでは、リモート ID を使用することで、複数の Mobile Link ユーザーが同じリモートデータベースを同期できるようになります。

  Mobile Link ユーザー名をパラメーターとして受け入れる各スクリプトで、remote_id パラメーターも受け入れるようになりました。remote_id パラメーターは、名前付きパラメーターを使用する場合にかぎり使用できます。

  リモート ID を変更できるように、新しいデータベースオプション ml_remote_id が SQL Anywhere と Ultra Light データベースの両方に追加されました。

  次の項を参照してください。

  - 「リモート ID」『Mobile Link クライアント管理』
  - 「Mobile Link ユーザー名とリモート ID」 329 ページ
  - SQL Anywhere クライアント：「リモート ID の設定」『Mobile Link クライアント管理』
  - Ultra Light クライアント：「Ultra Light ml_remote_id オプション」『Ultra Light データベース管理とリファレンス』

- **新しいファイル転送機能**　データの同期に使用しているのと同じネットワークパスを使用して、リモートデバイスにファイルを転送できる新しい機能が用意されました。SQL Anywhere クライアントでは mlfiletransfer ユーティリティ、Ultra Light クライアントでは新しい MLFileTransfer メソッドを使用できます。この機能は、新しいリモートデータベースにデータを設定する場合や、ソフトウェアをアップグレードする場合に特に便利です。必要に応じてファイル転送を認証するための新しい Mobile Link イベントが追加されました。次の項を参照してください。

  ○ SQL Anywhere クライアント：「Mobile Link ファイル転送ユーティリティ (mlfiletransfer)」『Mobile Link クライアント管理』
  ○ Ultra Light クライアント：「Mobile Link ファイル転送」『Ultra Light データベース管理とリファレンス』
  ○ Mobile Link サーバー：「authenticate_file_transfer 接続イベント」『Mobile Link サーバー管理』

- **SendColumnNames の変更**　以前は、dbmlsync 拡張オプション SendColumnNames と Ultra Light 同期パラメーター Send Column Names は、Mobile Link サーバーがサンプル同期スクリプトを生成できるように、リモートデータベースのカラム情報をアップロードするのに使用されていました。サンプル同期スクリプトの作成機能は削除されました (同期モデル作成ウィザードに置き換わりました)。SendColumnNames は、ダイレクトローハンドリングでのみ使用されるようになりました。 次の項を参照してください。

  ○「ダイレクトローハンドリング」『Mobile Link サーバー管理』
  ○「SendColumnNames (scn) 拡張オプション」『Mobile Link クライアント管理』
  ○「Send Column Names 同期パラメーター」『Ultra Light データベース管理とリファレンス』

- **単純化された活性タイムアウト設定**　活性タイムアウトをクライアントで制御できるようになりました。新しいネットワークプロトコルオプション timeout が導入されました。このオプションは liveness_timeout、contd_timeout、unknown_timeout、network_connect_timeout を置き換えます。

  「timeout」『Mobile Link クライアント管理』を参照してください。

- **Buffer_size の強化**　buffer_size ネットワークプロトコルオプションを使用して、TCP/IP プロトコルの書き込みバッファー処理や HTTP プロトコルの HTTP 本文サイズを制御できるようになりました。デフォルト値も変更されています。

  「buffer_size」『Mobile Link クライアント管理』を参照してください。

Ultra Light クライアント

- **Palm での network_leave_open のサポート**　Palm デバイスで、同期の完了後にネットワーク接続を開いたままにするかどうかを選択できるようになりました。他のプラットフォームには以前のリリースでもこの機能が提供されていました。

  「network_leave_open」『Mobile Link クライアント管理』を参照してください。

- **Ultra Light の強化**　その他の Ultra Light の強化については、「同期」345 ページを参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

SQL Anywhere クライアント

- **スクリプト化されたアップロード**　通常の同期では、dbmlsync はトランザクションログを使用してアップロードを作成し、前回のアップロード後にリモートデータベースで変更されたすべての関連データを同期します。アップロードされるローを正確に定義するストアドプロシージャーを記述し、トランザクションログの使用をバイパスできるようになりました。これらのストアドプロシージャーは DML を実行して結果セットをアップロードできるため、ローは必要に応じて動的に作成されます。

  「スクリプト化されたアップロード」『Mobile Link クライアント管理』を参照してください。

  スクリプト化されたアップロードをサポートするために、SQL Anywhere システムオブジェクトは次のように変更されました。

  ○ISYSPUBLICATION システムテーブルに新しいカラム (sync_type) が追加されました。「SYSPUBLICATION システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○ISYSSYNCSCRIPT システムテーブルに、同期スクリプトを追跡するための新しいカタログオブジェクトが追加されました。

  「SYSSYNCSCRIPT システムビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

  ○新しいシステムプロシージャー変換進行状況値が追加されました。次の項を参照してください。

  - 「sa_convert_ml_progress_to_timestamp システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
  - 「sa_convert_timestamp_to_ml_progress システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』

- **dbmlsync の新しいスケジュールオプション**　EVERY と INFINITE スケジュールオプションを使用するときに、dbmlsync の開始時に同期を実行しないことを指定できるようになりました。

  「NoSyncOnStartup (nss) 拡張オプション」『Mobile Link クライアント管理』を参照してください。

- **ダウンロード専用のパブリケーション**　データをダウンロードするだけのパブリケーションを作成できるようになりました。ダウンロード専用のパブリケーションでは、ログファイルを使用しません。

  「ダウンロード専用のパブリケーション」『Mobile Link クライアント管理』を参照してください。

- **エラー処理の強化**　新しいイベントフックが追加され、クライアントで dbmlsync がレポートするエラーを処理できるようになりました。

次の項を参照してください。

- 「イベントフックプロシージャー内でのエラーと警告の処理」『Mobile Link クライアント管理』
- 「sp_hook_dbmlsync_all_error」『Mobile Link クライアント管理』
- 「sp_hook_dbmlsync_communication_error」『Mobile Link クライアント管理』
- 「sp_hook_dbmlsync_misc_error」『Mobile Link クライアント管理』
- 「sp_hook_dbmlsync_sql_error」『Mobile Link クライアント管理』

- **dbmlsync でテーブル順序の確保の停止**　デフォルトでは、子テーブルが親テーブルより先にアップロードされると、dbmlsync はエラーを発行します。新しい拡張オプションを使用すると、この動作を無効にできます。

  「TableOrderChecking (toc) 拡張オプション」『Mobile Link クライアント管理』を参照してください。

- **永続的な接続**　複数の同期間で Mobile Link サーバーへの接続を開いたままにするように、dbmlsync に指定できるようになりました。

  「-pc+ dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

- **同期を追跡する新しい方法**　SQL Anywhere リモートデータベースの場合のみ、begin_publication または end_publication スクリプトで subscription_id パラメーターを指定できるようになりました。この値は、SYSSYNC システムテーブルで sync_id と呼ばれます。同期の情報を追跡できる高度な機能です。次の項を参照してください。

  - 「begin_publication 接続イベント」『Mobile Link サーバー管理』
  - 「end_publication 接続イベント」『Mobile Link サーバー管理』

- **dbmlsync がメッセージのタイムスタンプ設定に ISO 8601 日時フォーマットを使用**　情報、警告、エラーの各メッセージのタイムスタンプに、明確に定義された ISO 8601 日時フォーマット ({I|W|E}.yyyy-mm-dd hh:mm:ss message) を使用するようになりました。

- **#hook_dict で拡張された値**　dbmlsync ユーティリティは、フックを公開し、テンポラリテーブル #hook_dict に名前／値ペアを値として渡します。以前は、#hook_dict テーブル内の値は VARCHAR (255) として定義されていました。これが VARCHAR (10240) に増加しました。

セキュリティ

- **RSA が SQL Anywhere に付属**　FIPS 認定のアルゴリズムを使用していない場合は、RSA 暗号化を使用するためのライセンスを別途購入する必要がなくなりました。

- **HTTPS で使用可能な ECC 暗号化**　HTTPS の使用時に、ECC 暗号化を使用できるようになりました。

- **新しい mlsrv10 -fips オプション**　Mobile Link サーバーの起動時に -fips を指定して、すべてのセキュア接続で FIPS 認定のアルゴリズムを使用させることができるようになりました。この設定は、セキュアでないストリームには影響しません。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

「-fips mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **新しい mluser -fips オプション**　Mobile Link ユーザー認証ユーティリティで、FIPS 認定のセキュリティの使用を強制するオプションが提供されるようになりました。

  「Mobile Link ユーザー認証ユーティリティ (mluser)」『Mobile Link サーバー管理』を参照してください。

- **複数のプラットフォームで FIPS セキュリティのサポート**　より多くのプラットフォームで FIPS 認定のセキュリティがサポートされるようになりました。サポートされるプラットフォームのリストについては、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

- **単純化されたセキュリティストリームの指定方法**　セキュリティオプションを個別のネットワークプロトコルとして扱うことにより、セキュリティオプションを指定する構文がサーバーとクライアントの両方で単純化されました。現在、TCP/IP、TLS (TLS セキュリティを使用した TCP/IP 上の同期)、HTTP、HTTPS の各プロトコルがサポートされています。Ultra Light セキュリティパラメーターは削除されました。

  次の項を参照してください。

  - Mobile Link サーバー：「-x mlsrv12 オプション」『Mobile Link サーバー管理』
  - Mobile Link クライアント：「Mobile Link クライアントネットワークプロトコルオプション」『Mobile Link クライアント管理』

- **オペレーティングシステムとの統合**　デフォルトで Mobile Link クライアントは、動作するオペレーティングシステムで信頼されている証明書を信頼します。

サーバー起動同期

使いやすさ

- **サーバー起動同期は設定が大幅に簡易化されました。**　サーバー起動同期アプリケーションをよりすばやく設定できるよう強化されました。

  - **Sybase Central のサポート**　Notifier と Listener を Sybase Central モデルモードで設定できるようになり、便利な通知サービスのサブセットを使用できるようになりました。モデルモードでは、サーバー起動同期のテーブルを指定すると、通知目的で使用されるデータを判断するために download_cursor が自動的に使用されます。ダウンロードカーソルの変更でデータが識別されると、通知が送信されます。展開ウィザードは、対応する Listener オプションファイルを生成します。

    「同期モデルでのサーバー起動同期の設定」『Mobile Link クイックスタート』を参照してください。

  - **新しいデフォルトゲートウェイ**　新しいゲートウェイ SYNC ゲートウェイを使用すると、Mobile Link に使用するのと同じ種類の通信パス上で永続的な接続を確立できます。SYNC ゲートウェイは、デフォルトのデバイストラッカゲートウェイになりました。通知

はまず SYNC ゲートウェイを試行し、UDP ゲートウェイ、SMTP ゲートウェイの順にフォールバックします。

「ゲートウェイと Carrier」『Mobile Link サーバー起動同期』を参照してください。

Notifier の強化

- **共有接続**　複数の Notifier で、同じデータベースを共有できるようになりました。これにより、統合データベースでの競合と必須サーバーリソースが低減します。

  「Notifier イベント」『Mobile Link サーバー起動同期』を参照してください。

- **Notifier でのリモートデバイスの文字セットの使用**　リモートデバイスの文字セットを使用して、通知がリモートデバイスに送信されるようになりました。デバイストラッキング情報は、統合データベースへの適用前に変換されます。

- **カスタム確認処理**　SQL で、Push 要求の確認を処理してそのステータスを返す Notifier プロパティを実装できるようになりました。

  「confirmation_handler イベント」『Mobile Link サーバー起動同期』を参照してください。

- **カスタムエラー処理**　SQL で、Push 要求が未配信、未確認、不適切に確認されたときなどにエラーを処理する Notifier プロパティを実装できるようになりました。

  「error_handler イベント」『Mobile Link サーバー起動同期』を参照してください。

Listener の強化

- **永続的な接続**　Windows Listener で、永続的な接続をサポートするようになりました。デフォルトで Listener は、デバイスの追跡、通知、確認のために、Mobile Link サーバーへの永続的な接続を管理するようになりました。この機能により、以前のバージョンと比べて大幅にパフォーマンスが強化されました。dblsn -pc オプションを使用すると、この機能を無効にできます。

  「Windows デバイス用の Mobile Link Listener ユーティリティ (dblsn)」『Mobile Link サーバー起動同期』を参照してください。

- **新しい／変更された Windows Listener オプション**　Listener で次のオプションがサポートされるようになりました。

オプション	説明
-ni	-x を使用する場合に UDP アドレスのトラッキングを停止する。以前は -g と呼ばれていた。
-pc{+|-}	通知用に永続的な接続を有効／無効にする。
-ns	Windows Mobile 2003 以降の Phone Edition で、デフォルトの SMS 受信を無効にする。
-nu	デフォルトの UDP 受信を無効にする。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

オプション	説明
-r	$remote_id 変数で使用するリモート ID ファイルを登録する。
-v	1 以上に設定すると、冗長性オプションによってコマンドラインオプションが表示され、ログに記録される。

- **リモート ID ファイル**　Listener コマンドラインで、リモート ID ファイルを使用して新しい Mobile Link リモート ID (デフォルトは GUID) にアクセスできるようになりました。これを行うには、新しい dblsn オプションの -r と、新しい Listener action 変数 $remote_id を使用します。

  「Windows デバイス用の Mobile Link Listener ユーティリティ (dblsn)」『Mobile Link サーバー起動同期』と「Windows デバイス用の Mobile Link Listener アクションコマンド」『Mobile Link サーバー起動同期』を参照してください。

- **認証用の新しい Listener action 変数**　メッセージハンドラーでの使用に便利な新しい action 変数 $ml_user と $ml_password が追加されました。

  「Windows デバイス用の Mobile Link Listener アクションコマンド」『Mobile Link サーバー起動同期』を参照してください。

- **接続パラメーター用の新しい Listener action 変数**　新しい $ml_connect action 変数が、これまで dblsn -x オプションを使用して指定していた Mobile Link 接続パラメーターに拡張されました。

  「Windows デバイス用の Mobile Link Listener アクションコマンド」『Mobile Link サーバー起動同期』を参照してください。

- **Listener がメッセージのタイムスタンプ設定に ISO 8601 日時フォーマットを使用**　情報、警告、エラーの各メッセージのタイムスタンプに、明確に定義された ISO 8601 日時フォーマット ({I|W|E}.yyyy-mm-dd hh:mm:ss message) を使用するようになりました。

- **Listener での TLS の使用**　Listener は、他の Mobile Link クライアントと同様に、あらゆるネットワークオプションを使用して Mobile Link サーバーに接続できるようになりました。これによって、デバイストラッキングと通知でセキュリティを適用できます。

  「Windows デバイス用の Mobile Link Listener ユーティリティ (dblsn)」『Mobile Link サーバー起動同期』の -x の説明を参照してください。

デバイスサポートの増加

- **Treo 600 と 650 のサポート**　Palm Listener は、Treo 600 と 650 をサポートするようになりました。

- **CE Phone Edition のサポート**　Listener で、SMS で Windows Mobile 2003 Phone Edition がサポートされるようになりました。

動作の変更と廃止予定機能

次に、バージョン 10.0.0 で導入された Mobile Link の変更を示します。

Mobile Link サーバーの変更

Mobile Link スクリプトの変更

- **カーソルベースのアップロードの削除**　upload_cursor、new_row_cursor、old_row_cursor の各スクリプトは、バージョン 9.0.0 で廃止されていましたが、このバージョンで削除されました。代わりに、文ベースのスクリプトを使用してください。

  「ローをアップロードするスクリプト」『Mobile Link サーバー管理』を参照してください。

- **認識されないスクリプトにより同期が失敗する**　Mobile Link サーバーで認識されないテーブルレベルまたは接続レベルのスクリプトが検出されると、同期がアボートされます。以前のバージョンでは、認識されないスクリプトは警告メッセージを発行するだけでした。今後は、カーソルベースのアップロードスクリプトが存在すると、同期がアボートされます。

- **同期の失敗を引き起こすアップロードスクリプトまたはダウンロードスクリプトのエラー**　Mobile Link サーバーでアップロードスクリプトやダウンロードスクリプトのエラーを検出すると、同期が常にアボートされるようになりました。以前は、同期がアボートされない場合がありました。

- **制限が厳しくなった handle_error と handle_odbc_error イベントの動作**　handle_error と handle_odbc_error スクリプトが呼び出されるのは、アップロードトランザクション中に Mobile Link で挿入、更新、または削除スクリプトの処理中に ODBC エラーが発生した場合、またはダウンロードローをフェッチしている場合だけになりました。それ以外の場合に ODBC エラーが発生すると、Mobile Link サーバーは report_error または report_odbc_error スクリプトを呼び出して、同期をアボートします。

- **認証スクリプトのコミット**　エラーがない場合、Mobile Link サーバーは authenticate_user、authenticate_user_hashed、または authenticate_parameters を呼び出した後で常にトランザクションをコミットします。以前は、失敗した認証に関係するトランザクションがロールバックされていたため、認証の試行失敗のレコードはありませんでした。次の項を参照してください。

  ○「authenticate_user 接続イベント」『Mobile Link サーバー管理』
  ○「authenticate_user_hashed 接続イベント」『Mobile Link サーバー管理』
  ○「authenticate_parameters 接続イベント」『Mobile Link サーバー管理』

- **authenticate_user_hashed script の変更**　authenticate_user_hashed スクリプトは、ユーザーの認証シーケンス中に複数回呼び出すことができるようになりました。

  「authenticate_user_hashed 接続イベント」『Mobile Link サーバー管理』を参照してください。

- **開始スクリプトが呼び出されると、同期の成功に関係なく終了スクリプトが呼び出される**　Mobile Link スクリプトによっては、begin_connection と end_connection のように開始と終了の形式があります。以前は、同期に失敗すると、終了スクリプトが実行されないことが

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ありました。今回のバージョンでは、開始スクリプトが呼び出されると、同期でエラーが発生した場合でも、終了スクリプトが定義されていれば常に呼び出されるようになりました。

「同期イベント」『Mobile Link サーバー管理』を参照してください。

- **テーブルにアップロードするデータがない場合はアップロードスクリプトが呼び出されない**　以前のバージョンでは、-us オプションを使用して、アップロードするデータがない場合は Mobile Link サーバーがアップロードスクリプトを呼び出さないようにすることができました。-us オプションは削除され、デフォルトではアップロードストリームにアップロードするデータが含まれる場合にかぎり、Mobile Link サーバーはアップロードスクリプトを呼び出します。-zus オプションを使用すると、以前の動作に戻すことができます。

  「-zus mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **SQL Anywhere 10 と Microsoft SQL Server 2005 の統合データベースは、begin_connection スクリプトで独立性レベルを変更しない**　SQL Anywhere バージョン 10 と Microsoft SQL Server 2005 以降では、ダウンロードのデフォルトの独立性レベルがスナップショットになりました。独立性レベルはダウンロードトランザクションの開始時に変更できますが、その場合は begin_connection スクリプトの設定がすべて上書きされます。そのため、begin_download スクリプトのダウンロードの独立性レベルを変更するか、新しい mlsrv10 -dsd オプションを使用して、スナップショットアイソレーションを無効にする必要があります。以前のマニュアルでは、begin_connection スクリプトで独立性レベルを変更することをおすすめしていました。スナップショットアイソレーションを使用しない統合データベースに対しては、現在もこの方法が有効です。

  「Mobile Link 独立性レベル」『Mobile Link サーバー管理』を参照してください。

- **example_upload_cursor、example_upload_delete、example_upload_insert、example_upload_update テーブルイベントの削除**　-za と -ze の Mobile Link サーバーオプションが削除されたことにより、example_upload_cursor、example_upload_delete、example_upload_insert、example_upload_update テーブルイベントは生成されなくなりました。同期モデル作成ウィザードを使用してスクリプトを生成できるようになりました。

  「[同期モデル作成ウィザード] を使用した Mobile Link アプリケーションの設定」『Mobile Link クイックスタート』を参照してください。

mlsrv10 の変更

- **-w と -wu オプションの変更**　-w と -wu オプションは、データベースワーカースレッド数と、アップロードするデータベースワーカースレッドの最大数をそれぞれ設定します。以前のバージョンでは、これらのワーカースレッドが、ネットワークに対する読み込みと書き込み、プロトコルバイトとローデータのパックとアンパック、スクリプトの実行、統合データベースのローの更新とフェッチなど、同期のすべての処理を実行していました。

  今回のバージョンでは、-w と -wu で指定されたワーカースレッドはデータベースワーカースレッドになります。これらのデータベースワーカースレッドだけが、すべてのデータベースアクティビティの処理のみを行います。他のスレッドは、ネットワークアクティビティ、パックとアンパック、その他の Mobile Link サーバーのアクティビティの処理を行います。

  -w と -wu オプションの新しい動作は、ネットワークのアクティビティとは関係ありません。以前のバージョンでは、遅延時間が長いネットワークによってワーカースレッドがブロック

されることがあり、一部の配備で大量のワーカースレッドを指定する必要がありました。新しい Mobile Link アーキテクチャーでは、この要件はなくなりました。

-w と -wu オプションは、Mobile Link サーバーが統合データベースに与える負荷を制限する最も簡単な方法です。-w と -wu の値をテストすると、お使いの同期システムに最適なスループットを見つけることができます。次の項を参照してください。

- 「-w mlsrv12 オプション」『Mobile Link サーバー管理』
- 「-wu mlsrv12 オプション」『Mobile Link サーバー管理』
- 「Mobile Link のパフォーマンス」『Mobile Link サーバー管理』

- **キャッシュサイズの設定オプションの削除**　次の mlsrv10 オプションが削除されました。

  - -bc
  - -d
  - -dd
  - -u

  これらのオプションは、mlsrv10 -cm で置き換えられました。このオプションは、すべての同期でキャッシュを設定します。

  「-cm mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **タイムアウトの設定オプションの削除**　次の mlsrv10 オプションは不要になり、削除されました。

  - contd_timeout
  - unknown_timeout

  また、mlsrv10 liveness_timeout オプションも削除されました。同期クライアント用のタイムアウトオプションで置き換えられました。

  「timeout」『Mobile Link クライアント管理』を参照してください。

- **不要になったバックログオプション**　mlsrv10 バックログオプションは不要になり、削除されました。

- **プロトコル名とネットワークセキュリティオプションの変更**　ネットワークプロトコルキーワード https_fips、rsa_tls、rsa_tls_fips、ecc_tls と、ネットワークプロトコルのオプションセキュリティが削除されました。プロトコルは削除されていませんが、異なる方法で指定する必要があります。mlsrv10 -x 構文は次のように変更されました。

以前の構文	バージョン 10.0.0 の新しい構文	説明
-x https_fips	-x https(fips=y;...)	HTTPS FIPS
-x rsa_tls	-x tls(tls_type=rsa;...)	RSA 暗号化を使用した TCP/IP TLS

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

以前の構文	バージョン 10.0.0 の新しい構文	説明
-x rsa_tls_fips	-x tls(tls_type=rsa;fips=y;...)	RSA 暗号化と FIPS 認定の暗号化を使用した TCP/IP TLS
-x ecc_tls	-x tls(tls_type=ecc;...)	ECC 暗号化を使用した TCP/IP TLS
-x tcpip(security=...)	-x tcpip	TCP/IP
-x http(security=...)	-x http	HTTP

「-x mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **-bn オプションの変更**　mlsrv10 -bn オプションは、競合検出時に BLOB バイトを比較します。以前は、LONGVARCHAR 型のデータの文字が比較されていました。今回のバージョンでは、バイナリと LONGVARCHAR BLOB で比較される単位は常にバイトになります。

  「-bn mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **冗長出力の変更**　mlsrv10 のオプション -vr、-vt、-vu は、どれもわずかに異なる情報を出力します。

  - **-vr**　-vr は、アップロードとダウンロードのロー値だけを返すようになりました。以前は、アップロードとダウンロードのスクリプト名と内容も返していました。
  - **-vt**　-vt は、変換後のスクリプトの内容だけを返すようになりました。以前は、元のスクリプトの内容も返していました。
  - **-vu**　-vu は、未定義のテーブルスクリプトを呼び出す必要があるときに、それらのスクリプトすべてを返すようになりました。これには統計スクリプトも含まれます。

  「-v mlsrv12 オプション」『Mobile Link サーバー管理』を参照してください。

- **Mobile Link サーバーオプション -za と -ze の削除**　Mobile Link サーバーの -za オプションと -ze オプションによる自動スクリプト生成は削除されました。同期モデル作成ウィザードを使用してスクリプトを生成できるようになりました。

  「[同期モデル作成ウィザード] を使用した Mobile Link アプリケーションの設定」『Mobile Link クイックスタート』を参照してください。

- **-zac と -zec の削除**　Mobile Link サーバーの、カーソルベースのスクリプトを生成する -zac と -zec オプションは廃止されていましたが、このバージョンで削除されました。

- **Mobile Link サーバーオプション -oy の削除**　mlsrv10 -oy オプション (タイムスタンプの年を表す) が削除されました。今後は、情報、警告、エラーの各メッセージのタイムスタンプに年は常に含まれます。

統計プロパティ

- **統計プロパティの変更**　大きく 2 つの変更があります。

○アップロードとダウンロードのバイト数の意味が変更されました。この数は、アップロードとダウンロードを格納するために Mobile Link サーバー内で使用されるメモリ量を反映するようになりました。以前は、Mobile Link サーバーに対して送受信されたアップロードとダウンロードのバイト数を表していました。新しい数は、同期がサーバーメモリに与える影響が反映されるので、より有用になりました。また、以前の数は、HTTP、暗号化、または圧縮が使用されたときの信頼性に欠けていました。

○マニュアルの前のバージョンでは、プロパティを通常のアップロードモードと強制的な競合モードのどちらで使用しているかによってプロパティが異なる値を返すことについて、プロパティの説明に記述されていませんでした。この点が修正されました (以下を参照)。

次の統計プロパティが変更されました。

統計プロパティ	説明
conflicted_deletes	通常のアップロードモードでは、この値は常に 0 です。 強制的な競合モードでは、upload_old_row_insert スクリプトを使用して統合データベースに正常に挿入されたアップロード削除の総数を返します。 以前は、競合が検出されたアップロード済み削除の数を返していました。
conflicted_inserts	通常のアップロードモードでは、この値は常に 0 です。 強制的な競合モードでは、upload_new_row_insert スクリプトを使用して統合データベースに正常に挿入されたアップロード挿入の総数を返します。 以前は、競合が検出されたアップロード済み挿入の数を返していました。
conflicted_updates	通常のアップロードモードでは、競合を引き起こした更新ローの総数を返します。 強制的な競合モードでは、upload_new_row_insert または upload_old_row_insert スクリプトを使用して正常に適用されたアップロード更新ローの総数を返します。 以前は、競合が検出されたアップロード済み更新の数を返していました。
download_bytes	ダウンロードを格納するために Mobile Link サーバー内で使用されるメモリ量を返します。 以前は、ダウンロード済みバイト数を返していました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

統計プロパティ	説明
ignored_deletes	通常のアップロードモードでは、handle_error または handle_odbc_error が定義されており、1000 が返された場合、または指定のテーブルに対して upload_delete スクリプトが定義されていない場合において、upload_delete スクリプトを呼び出したときにエラーを発生させたアップロード削除ローの総数を返します。 強制的な競合モードでは、handle_error または handle_odbc_error が定義されており、1000 が返された場合、または指定のテーブルに対して upload_old_row_insert スクリプトが定義されていない場合において、upload_old_row_insert スクリプトを呼び出したときにエラーを発生させたアップロード削除ローの総数を返します。 以前は、無視されたアップロード済み削除の数を返していました。
ignored_inserts	通常のアップロードモードでは、handle_error または handle_odbc_error が定義されており、1000 が返された場合、または指定のテーブルに対して upload_insert スクリプトが定義されていない場合において、upload_insert スクリプトを呼び出したときにエラーを発生させたアップロード挿入ローの総数を返します。 強制的な競合モードでは、handle_error または handle_odbc_error が定義されており、1000 が返された場合、または指定のテーブルに対して upload_insert スクリプトが定義されていない場合において、upload_new_row_insert スクリプトを呼び出したときにエラーを発生させたアップロード挿入ローの総数を返します。 以前は、無視されたアップロード済み挿入の数を返していました。
ignored_updates	通常のアップロードモードでは、handle_error または handle_odbc_error が定義されており、1000 が返された場合、または指定のテーブルに対して upload_update スクリプトが定義されていない場合において、upload_update スクリプトを呼び出したときにエラーを発生させたアップロード更新ローの総数を返します。 強制的な競合モードでは、handle_error または handle_odbc_error が定義されており、1000 が返された場合、または upload_new_row_insert または upload_old_row_insert スクリプトを呼び出したときにエラーを発生させたアップロード更新ローの総数を返します。 以前は、無視されたアップロード済み更新の数を返していました。
upload_bytes	アップロードを格納するために Mobile Link サーバー内で使用されるメモリ量を返します。 以前は、アップロード済みバイト数を返していました。

統計プロパティ	説明
upload_deleted_rows	通常のアップロードモードでは、統合データベースから正常に削除されたローの総数を返します。 強制的な競合モードでは、この値は常に 0 です。 以前は、同期クライアントからアップロードされたロー削除の数を返していました。
upload_inserted_rows	通常のアップロードモードでは、統合データベースに正常に挿入されたローの総数を返します。 強制的な競合モードでは、この値は常に 0 です。 以前は、同期クライアントからアップロードされたロー挿入の数を返していました。
upload_updated_rows	通常のアップロードモードでは、統合データベースで正常に更新されたローの総数を返します。 強制的な競合モードでは、この値は常に 0 です。 以前は、同期クライアントからアップロードされたロー更新の数を返していました。

「Mobile Link の統計のプロパティ」『Mobile Link サーバー管理』を参照してください。

Mobile Link サーバーのその他の変更

- **リモートデータベースの CHAR または NCHAR データ型のカラムに対して NULL 文字を同期できるようになった**　以前の Mobile Link では、NULL 文字を含む VARCHAR カラムと CHAR カラムの値によって同期が失敗することがありました。今回のバージョンでは、CHAR、VARCHAR、LONG VARCHAR、NCHAR、NVARCHAR、LONG NVARCHAR データ型のリモートデータベースカラムに含まれる NULL 文字を同期できるようになりました。

- **情報、警告、エラーメッセージのログの新しいフォーマット**　以前は、Mobile Link サーバーは次のフォーマットでメッセージをログに記録していました。

  T.mm/dd hh:mm:ss. thread_id User_name: message

  今回のバージョンでは、次のフォーマットでメッセージがログに記録されるようになりました。

  T. yyyy-mm-dd hh:mm:ss. synchronization_id: message

  同期ごとに、ログの最初のメッセージはリモート ID、ユーザー名、スクリプトバージョン、クライアント名 (Ultra Light または SQL Anywhere) を示します。

  新しいフォーマットでは、提供される情報は減ることなく、出力ログのサイズが小さくなりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **Oracle 用システムプロシージャーの新しいデータ型**　スクリプトを登録するための Mobile Link システムプロシージャーでは、Oracle 統合データベース用にスクリプトの内容パラメーターで CLOB データ型が使用されるようになりました。ml_add_property システムプロシージャーでは、Oracle 用に prop_value パラメーターが CLOB 型になりました。以前は、これらのパラメーターは VARCHAR 型でした。

Mobile Link ユーザー名とリモート ID

Mobile Link は、リモート ID と呼ばれるユニークな ID を生成するようになりました。これは、リモートデータベースが初めて同期されるとき、または NULL 値のリモート ID が出現したときに生成されます。Mobile Link ユーザー名は、ユニークである必要がなくなりました。Mobile Link ユーザー名は、認証に使用される実際のユーザー名であると見なされるようになりました。

以前のバージョンでは、Mobile Link ユーザー名に対して同期の進行状況が格納されていました。今回のバージョンでは、SQL Anywhere リモートのリモート ID とサブスクリプション、および Ultra Light リモートのリモート ID とパブリケーションに対して、進行状況が格納されるようになりました。

以前は、Mobile Link ユーザー名を使用して、リモートデータベースをユニークに識別していました。Mobile Link ユーザーに複数のリモートデータベースを同期させる場合に、リモート ID を使用してリモートデータベースを識別すると便利です。Ultra Light リモートデータベースでは、複数の Mobile Link ユーザーが同じリモートデータベースを同期する場合に、リモート ID を使用すると便利です。

「リモート ID」『Mobile Link クライアント管理』を参照してください。

Ultra Light クライアント

「動作の変更と廃止予定機能」 351 ページを参照してください。

SQL Anywhere クライアント

- **ダウンロードエラーフックは廃止予定**　エラーフック sp_hook_dbmlsync_download_com_error、sp_hook_dbmlsync_fatal_sql_error、sp_hook_dbmlsync_sql_error は廃止される予定です。これらは置き換えられました。

  「イベントフックプロシージャー内でのエラーと警告の処理」『Mobile Link クライアント管理』を参照してください。

- **sp_hook_dbmlsync_log_rescan は dbmlsync が別の同期を予期した場合だけ呼び出される**　以前は、各同期の終了時に sp_hook_dbmlsync_log_rescan フックが呼び出されていました。そのため、dbmlsync が Mobile Link サーバーから切断されてから、ログに同期の完了メッセージが表示されるまでに間がありました。今回のバージョンでは、コマンドラインで dbmlsync -n オプションが複数回指定された場合や、スケジュールが有効になっている場合など、dbmlsync が別の同期を予期した場合にかぎりこのフックが呼び出されるようになりました。

  「sp_hook_dbmlsync_log_rescan」『Mobile Link クライアント管理』を参照してください。

- **活性タイムアウトオプションの単純化**　クライアントで、liveness_timeout と network_connect_timeout ネットワーク接続プロトコルオプションが削除されました。代わりに、timeout 接続オプションを使用してください。

  「timeout」『Mobile Link クライアント管理』を参照してください。

- **圧縮をしない場合は難読化されない**　圧縮を none に設定した場合、データはまったく難読化されなくなりました。セキュリティ上問題がある場合は、トランスポートレイヤーセキュリティを使用して、データを暗号化してください。

  「compression」『Mobile Link クライアント管理』を参照してください。

- **バージョン 7 の構文とユーティリティの削除**　次の SQL 文とユーティリティは廃止されていましたが、このバージョンで削除されました。

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

  ○CREATE SYNCHRONIZATION SITE 文

  ○CREATE SYNCHRONIZATION DEFINITION 文

  ○CREATE SYNCHRONIZATION TEMPLATE 文

- **ActiveSync の新しいネットワークプロトコルオプション**　ActiveSync ユーザーは、CommunicationAddress 拡張オプションまたは SQL 文で ADDRESS 句を指定するときに、ActiveSync プロトコルを指定する必要がなくなりました。代わりに、ActiveSync 用 Mobile Link プロバイダーと Mobile Link サーバー間の通信に使用しているプロトコルとプロトコルオプションを指定するだけで済みます。

  「Mobile Link クライアントネットワークプロトコルオプション」『Mobile Link クライアント管理』を参照してください。

- **dbmlsync の新しいシャットダウン方法**　dbmlsync の -k オプションは廃止され、-qc オプションに置き換えられました。

  「-qc dbmlsync オプション」『Mobile Link クライアント管理』を参照してください。

その他の Mobile Link の動作の変更

バージョンサポート

- **バージョン 8.0.0 以前のクライアントのサポートの削除**　Mobile Link サーバーでは、バージョン 8.0.0 以前の SQL Anywhere クライアントがサポートされなくなりました。バージョン 10 の Mobile Link サーバーで旧バージョンのデータベースを使用するには、アップグレード手順に従う必要があります。

  「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

名前の変更

次のユーティリティ名が変更されました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

以前のユーティリティ名	新しいユーティリティ名
dbmlsrv9	mlsrv10
dbmluser	mluser
dbmlmon	mlmon
dbmlstop	mlstop
dbasinst	mlasinst

次のファイル名が変更されました。

以前のファイル名	新しいファイル名
dbmlsv9.dll	*mlodbc10.dll*
dbasdesk.dll	*mlasdesk.dll*
dbasdev.dll	*mlasdev.dll*
dbmlsrv.mle	*mlsrv10.mle*
syncasa.sql	*syncsa.sql*

ODBC ドライバーの強化

Mobile Link では、Adaptive Server Enterprise と DB2 の統合データベース用に新しいドライバーが使用されるようになりました。

「Mobile Link、QAnywhere、リモートデータアクセスで使用される ODBC の変更」362 ページを参照してください。

サーバー起動同期

- **Windows SDK の削除**　多くの Windows デバイスをサポートするための SDK は削除されました。向上した SMS サポートで置き換えられました。Palm SDK は残されます。
- **Listener -g オプションの置き換え**　dblsn -g オプションは dblsn -ni オプションで置き換えられました。

Mobile Link のその他の動作の変更

- **Windows パフォーマンスモニターのサポート終了**　Mobile Link で、Windows パフォーマンスモニターはサポートされなくなりました。代わりに Mobile Link モニターを使用してください。

  「Mobile Link モニター」『Mobile Link サーバー管理』を参照してください。

- **サーバー起動同期における Kyocera デバイスのサポート終了**　Kyocera デバイス用の Palm SDK はなくなりました。Palm Listener では、引き続き Treo デバイスをサポートします。

QAnywhere

次の項では、QAnywhere バージョン 10.0.0 の新機能、動作の変更、廃止予定機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された QAnywhere の追加機能を示します。

モバイル Web サービス

モバイル Web サービスでは、モバイルに最適化された非同期 Web サービスのサポートを提供します。これによって、モバイルアプリケーションでは、オフラインであっても Web サービス要求を行うことができ、あとで転送するためにそれらの要求をキューイングすることができます。要求は QAnywhere を使用してメッセージとして配信されます。サーバー側の Web サービスコネクターは、クライアントの要求を Web サービスに転送します。次に、Web サービスから応答を受け取り、クライアントにメッセージとして返します。WSDL コンパイラーが用意されているため、.NET または Java アプリケーションからモバイル Web サービスを使用できます。

「モバイル Web サービス」『QAnywhere』を参照してください。

Sybase Central 用の新しい QAnywhere プラグイン

Sybase Central に QAnywhere プラグインが含まれるようになりました。このプラグインは、QAnywhere アプリケーションの作成と管理に使用する、使いやすいグラフィカルなインターフェイスを提供します。QAnywhere プラグインでは、次の作業を実行できます。

- クライアントメッセージストアとサーバーメッセージストアの作成
- QAnywhere Agent の設定ファイルの作成と管理
- QAnywhere Agent のログファイルのブラウズ
- 送信先エイリアスの作成または変更
- JMS コネクターと Web サービスコネクターの作成
- 転送ルールファイルの作成と管理
- メッセージストアのリモートでのブラウズ
- メッセージの追跡

QAnywhere は UNIX プラットフォームでサポートされていませんが、UNIX で Sybase Central を使用してメッセージを追跡できるようになりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

新しい QAnywhere クライアント API

- **新しい SQL API** QAnywhere SQL API は、SQL ストアドプロシージャーのセットです。SQL 開発者は、簡単に QAnywhere メッセージング機能を使用できるようになります。この API を使用すると、既存のデータベースアプリケーションを補完する簡単な方法で、ストアドプロシージャーでメッセージを送受信できます。これにより、データベースとメッセージングの操作を 1 つのトランザクションに組み合わせる強力なアプリケーションが可能になります。たとえば、1 つのストアドプロシージャーによって、データベースへのローの挿入と、別のアプリケーションへのメッセージ送信を実行し、さらに両方のアクションを同じトランザクションの一部としてコミットさせることができます。

  「QAnywhere SQL API リファレンス」『QAnywhere』を参照してください。

- **新しい Java クライアント API** Java 用の新しい QAnywhere クライアント API を使用すると、Java でメッセージングクライアントアプリケーションを作成できます。Java 用のクライアント API は、現時点では Windows CE を含む Windows のみでサポートされています。

  「クライアント用 QAnywhere Java API リファレンス」『QAnywhere』を参照してください。

QAnywhere クライアント API の強化

QAnywhere クライアント API にインデックス処理に次のような追加が行われています。

- **メッセージセレクター** SQL に似た式を使用して、キューからメッセージを選択的にブラウズしたり受け取ったりすることができるようになりました。メッセージセレクターを作成する構文は、転送ルールの条件に使用する構文と同じです。

  「QAnywhere のメッセージの参照」『QAnywhere』を参照してください。

- **メッセージをブラウズする新しい方法** 複数のキューからメッセージをブラウズしたり、ID やメッセージセレクターを基にメッセージのサブセットをブラウズしたりすることができるようになりました。

  「セレクターを使用したメッセージの参照」『QAnywhere』を参照してください。

- **メッセージストアプロパティ名の列挙** メッセージストアプロパティ名を列挙できるようになりました。

  「クライアントメッセージストアプロパティの列挙」『QAnywhere』を参照してください。

- **配信不可メッセージ** 新しいメッセージストアプロパティ ias_MaxDeliveryAttempts を使用すると、QAnywhere クライアントが配信不可と判断する前にメッセージを受信しようとする最大試行回数を設定できます。

  「ルール変数」『QAnywhere』を参照してください。

- **メッセージのキャンセル** メッセージを送信する前にキャンセルできるようになりました。

  「QAnywhere メッセージのキャンセル」『QAnywhere』を参照してください。

- **メッセージステータスのクエリ** 事前に定義された新しいメッセージプロパティ ias_Status と ias_StatusTime を使用して、メッセージのステータスを問い合わせられるようになりました。

ias_Originator を使用してメッセージの発信者を問い合わせたり、ias_DeliveryCount を使用してメッセージが受信者に配信された回数を問い合わせたりすることもできます。

「事前に定義されたメッセージプロパティ」『QAnywhere』を参照してください。

- **アップロードのインクリメントを設定する新しいメッセージストアプロパティ** ias_MaxUploadSize を使用すると、アップロードのインクリメントを変更できます。

  「事前に定義されたクライアントメッセージストアプロパティ」『QAnywhere』を参照してください。

QAnywhere Agent の新機能

- **単一デバイス上の複数エージェント** 以前は、1 つのデバイスで 1 つの QAnywhere Agent インスタンスしか実行できませんでした。この制限がなくなりました。

  「QAnywhere Agent の起動」『QAnywhere』を参照してください。

- **フェールオーバーを設定するオプションの追加** 2 つの新しい QAnywhere Agent オプション -fd と -fr を使用すると、フェールオーバーが発生する方法をカスタマイズできます。

  「-fd qaagent オプション」『QAnywhere』と「-fr qaagent オプション」『QAnywhere』を参照してください。

- **永続的な接続** 新しいオプション -pc+ が追加され、メッセージの転送用に永続的な接続を使用できるようになりました。新しい -push オプションは、-push_notifications を置き換えます。またこのオプションで、Push 通知に永続的な接続を使用するかどうかを指定できるようになりました。

  次の項を参照してください。

  ○ 「-pc qaagent オプション」『QAnywhere』
  ○ 「-push qaagent オプション」『QAnywhere』

- **新しいアップグレードプロシージャー** 新しい -sur オプションを使用すると、以前のバージョンの SQL Anywhere からのクライアントメッセージストアをアップグレードできます。

  「-sur qaagent オプション」『QAnywhere』を参照してください。

- **QAnywhere Agent がメッセージのタイムスタンプ設定に ISO 8601 日時フォーマットを使用** 情報、警告、エラーの各メッセージのタイムスタンプに、明確に定義された ISO 8601 日時フォーマット ({I|W|E} yyyy-mm-dd hh:mm:ss message) を使用するようになりました。

その他の QAnywhere の強化

- **送信先エイリアス** QAnywhere 送信先のセットを表す送信先エイリアスを定義できるようになりました。送信先エイリアスに送信されたメッセージは、エイリアスの各メンバーに送信されます。

- **サーバー管理要求** サーバー管理要求を使用して、アクティビティ (送信先エイリアスの作成や、JMS コネクターの監視、開始、停止など) を管理および監視できるようになりました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

クライアントでサーバー管理要求を作成して、それらを処理するためにサーバーメッセージストアに送信できます。

「サーバー管理要求」『QAnywhere』を参照してください。

- **サーバー側の転送ルールのメンテナンスの向上**　デフォルトのサーバー側の転送ルールを変更することができるようになりました。変更は、すべてのクライアントに自動的に適用されます。以前は、デフォルトを変更するには、各クライアントに対して転送ルールを手動で定義する必要がありました。

  「サーバー側の転送ルール」『QAnywhere』を参照してください。

- **メッセージプロパティの追加**　事前に定義されたメッセージプロパティが QAnywhere に追加設定されました。メッセージの処理がより柔軟になり、デバッグ中の情報が詳細になり、メッセージのステータスをトラブルシューティングしやすくなりました。

  「メッセージプロパティ」『QAnywhere』を参照してください。

- **JMS 送信先にバックスラッシュを埋め込み可能**　バックスラッシュのデリミターが必要なサブコンテキストを JMS 送信先に含めることができるようになりました。

  「JMS コネクターへの QAnywhere メッセージの送信」『QAnywhere』を参照してください。

- **新しい転送ルール関数**　次の転送ルール関数が追加され、日付処理が向上しました。

  - DATEADD(datepart, count, datetime)
  - DATEPART(datepart, date)
  - DATETIME(string)
  - LENGTH(string)
  - SUBSTR(string, start, length)

  「ルール関数」『QAnywhere』を参照してください。

- **転送ルールのプロパティの前置**　メッセージプロパティやメッセージストアプロパティを転送ルールで使用するときに、これらの名前をプレフィクスとして使用して、同じ名前の転送ルール変数に指定された優先度をバイパスできるようになりました。

  「ルール変数としてのプロパティの使用」『QAnywhere』を参照してください。

動作の変更と廃止予定機能

次に、バージョン 10.0.0 で導入された QAnywhere の変更を示します。

QAnywhere クライアントの変更

- **クライアントメッセージストア ID の変更**　クライアントメッセージストア ID は Mobile Link のリモート ID になりました。以前は、Mobile Link ユーザー名でした。統合データベースでリモート ID を登録する必要はありません。ただし、Mobile Link ユーザー名はサーバーメッセージストアで登録する必要があります。Mobile Link ユーザー名を指定しない場合は、デフォルトでクライアントメッセージストア ID が指定されます。

Mobile Link ユーザー名を管理するために、新しい qaagent オプションが用意されています。次の項を参照してください。

○「-mn qaagent オプション」『QAnywhere』
○「-mp qaagent オプション」『QAnywhere』
○「-mu qaagent オプション」『QAnywhere』

- **新しい ODBC ドライバー**　Adaptive Server Enterprise と DB2 サーバーメッセージストアに接続するための iAnywhere Solutions ODBC ドライバーは、変更されています。

  「Mobile Link、QAnywhere、リモートデータアクセスで使用される ODBC の変更」362 ページを参照してください。

QAnywhere Agent の変更

- **qaagent -port の削除**　-port オプションは、QAnywhere Agent が Listener からの通信を受信するポート番号を指定するオプションでした。このオプションは不要になり、削除されました。空きポートが自動的に使用されます。

- **qaagent -la_port の置き換え**　-la_port オプションは -lp オプションで置き換えられました。

  「-lp qaagent オプション」『QAnywhere』を参照してください。

- **qaagent -push_notifications の名前の変更**　このオプションは -push という名前になりました。永続的な接続を使用した、または使用しない Push 通知を有効にできるようになりました。

  「-push qaagent オプション」『QAnywhere』を参照してください。

- **ポリシーのデフォルトの変更**　デフォルトのポリシーが自動 (automatic) になりました。以前のデフォルト値は、スケジュール済み (scheduled) です。デフォルトのスケジュール間隔は 900 秒 (15 分) になりました。以前のデフォルト値は 10 秒です。

  「-policy qaagent オプション」『QAnywhere』を参照してください。

- **トランザクションログの使用と管理の終了**　QAnywhere Agent では、トランザクションログが作成されなくなり、そのサイズも管理されなくなりました。このため、ほとんどのアプリケーションでは、トランザクションログなしでデータベースを初期化する dbinit -n オプションを使用して、クライアントメッセージストアを作成する必要があります。

  「クライアントメッセージストアの設定」『QAnywhere』を参照してください。

QAnywhere のその他の変更

- **サーバー側プロパティファイルは廃止予定**　プロパティは、ファイルではなくデータベースに格納されるようになりました。

- **getPropertyNames**　getPropertyNames 関数が C++ クライアント API から削除されました。この関数は、beginEnumPropertyNames、nextPropertyName、endEnumPropertyNames で置き換えられました。

  QAMessage クラス [QAnywhere C++]『QAnywhere』を参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **転送ルールでの日付処理**　次の転送ルールメッセージストア変数が削除されました。

  - ias_CurrentDayOfWeek
  - ias_CurrentDayOfMonth
  - ias_CurrentMonth
  - ias_CurrentYear

  代わりに、ias_CurrentTimestamp または DATEPART を使用できます。

  「ルール変数」『QAnywhere』を参照してください。

- **QAnywhere Central の置き換え**　QAnywhere Central は Sybase Central 用の QAnywhere プラグインで置き換えられました。このプラグインでは、多くの機能強化が加えられています。

SQL Remote

次の項では、SQL Remote バージョン 10.0.0 の新機能、動作の変更、廃止予定機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された SQL Remote の追加機能を示します。

- **invalid_extensions オプション**　新しいメッセージングオプションが追加され、FILE と FTP メッセージングで特定のファイル拡張子を使用することにより、SQL Remote を停止できるようになりました。

  「SET REMOTE OPTION 文 [SQL Remote]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **UNIX と Linux プラットフォームで、Message Agent (dbremote) にグラフィカルなユーザーインターフェイスが備わりました。**　「SQL Remote Message Agent ユーティリティ (dbremote)」『SQL Remote』の -ux オプションを参照してください。

- **dbremote がメッセージのタイムスタンプ設定に ISO 8601 日時フォーマットを使用**　情報、警告、エラーの各メッセージのタイムスタンプに、明確に定義された ISO 8601 日時フォーマット ({I|W|E} yyyy-mm-dd hh:mm:ss message) を使用するようになりました。

- **dbremote の新しいオプション**　完了時にウィンドウを閉じるには、新しい -qc オプションを使用してください。

  「SQL Remote Message Agent ユーティリティ (dbremote)」『SQL Remote』を参照してください。

動作の変更と廃止予定機能

次に、バージョン 10.0.0 で導入された SQL Remote の変更を示します。

- **Adaptive Server Enterprise データベースのサポートの停止**　SQL Remote では、Adaptive Server Enterprise 統合データベースはサポートされなくなりました。このため、ssxtract、ssremote、ssqueue、その他すべての SQL Remote for Adaptive Server Enterprise ユーティリティとファイルがインストールに含まれなくなりました。

  Adaptive Server Enterprise データベースを同期するには、Mobile Link を使用する必要があります。

  SQL Remote から Mobile Link へのアップグレードについては、http://www.ianywhere.com/whitepapers/migrate_to_ml.html を参照してください。

- **データベース抽出ウィザードで旧式のデータベースを抽出できなくなった**　データベース抽出ウィザードは、バージョン 10 のデータベースだけに使用できます。

- **#hook_dict で拡張された値**　ユーティリティ dbxtract と dbremote は、フックを公開し、テンポラリテーブル #hook_dict に名前／値ペアを値として渡します。以前は、#hook_dict テーブル内の値は VARCHAR (255) として定義されていました。これが VARCHAR (10240) に増加しました。

- **抽出ユーティリティの変更**　dbxtract には、いくつか変更が加えられています。

  ○-j、-k、-x オプションは削除されました。

  ○新しいオプション -al と -xh が追加されました。

  「抽出ユーティリティ (dbxtract)」『SQL Remote』を参照してください。

- **Message Agent (dbremote) の廃止予定のオプション**　完了時にウィンドウを閉じるための -k オプションは廃止される予定です。完了時にウィンドウを閉じるには、新しい -qc オプションを使用してください。

Ultra Light

次の項では、Ultra Light バージョン 10.0.0 の新機能、動作の変更、廃止予定機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された Ultra Light の追加機能を示します。

主な機能

Ultra Light は、管理のしやすさと SQL Anywhere の互換性を念頭に置いて設計された、完全な機能を備えた関係データベース管理システムになりました。新機能や便利な機能が追加されましたが、Ultra Light の占有容量は小さいままです。今回のリリースにおける Ultra Light の制限事項

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

の詳細については、「Ultra Light および Ultra Light Java Edition データベースの制限事項」『Ultra Light データベース管理とリファレンス』を参照してください。

今回のリリースに含まれる主な機能は次のとおりです。

- **データベースの上限の増加** Ultra Light データベースの上限が大幅に増加されました。特に、テーブルのローの最大数は 1600 万にまで増加されています。現在のデータベースのその他の制限の詳細については、「Ultra Light および Ultra Light Java Edition データベースの制限事項」『Ultra Light データベース管理とリファレンス』を参照してください。

- **スキーマの統合** Ultra Light はスタンドアロンの RDBMS になり、データベースの論理構造を定義する個別のスキーマファイルは必要なくなりました。今回のリリースでは、Ultra Light スキーマはデータベースに完全に統合されました。内部データベーススキーマの詳細については、「Ultra Light データベースのスキーマ」『Ultra Light データベース管理とリファレンス』を参照してください。

- **統合されたファイルフォーマット** バージョン 10 の Ultra Light では、ファイルフォーマットが統合されました。これにより、ほとんどのプラットフォームでデータベースファイルを共有できるようになりました。必要な照合が定義されていない文字が必要な場合は、データベースのエンコードに UTF-8 を選択しなければなりません。詳細については、「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」『Ultra Light データベース管理とリファレンス』と「Ultra Light utf8_encoding 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

- **データベースパフォーマンスとデータ整合性の向上** インデックス処理とデータベースページ管理が改善された結果、Ultra Light のデータベースパフォーマンスとデータ整合性が全体的に向上しました。

- **インデックスでハッシュ処理を利用可能** ハッシュ処理を利用するためにインデックスを指定できるようになりました。ハッシュサイズは、インデックス単位で指定できます。ハッシュサイズによって、インデックスルックアップのパフォーマンス改善につながったり、データベースファイルサイズに影響を及ぼしたりする可能性があります。「Ultra Light のパフォーマンスに関するヒント」『Ultra Light データベース管理とリファレンス』を参照してください。

- **データベースの直接作成** Ultra Light データベースファイルを直接作成できるようになりました。Ultra Light データベースのソースとして、データベーススキーマファイルやリファレンスデータベースファイルは必要ありません。Sybase Central やコマンドラインユーティリティを使用したり、アプリケーションからプログラムで設定して、Ultra Light データベースを個別に作成できます。

  既存の Ultra Light ユーザーの場合は、以前のバージョンと同じ方法でデータベースを作成できなくなりました。「Ultra Light のアップグレード」 396 ページを参照してください。

- **Windows CE の直接サポート** このリリースでは、Ultra Light アプリケーションはデスクトップから Windows CE デバイスに展開されたデータベースに直接接続できます。プレフィクス **WCE:¥** を付けてパスと名前を指定することにより、Ultra Light データベースを指定できます。これらの直接アクセスは、Sybase Central と Interactive SQL を含むすべてのクライアントアプリケーションと管理ツールでサポートされています。「Windows Mobile」『Ultra Light データベース管理とリファレンス』を参照してください。

- **動的 SQL プログラミングインターフェイスとしての Embedded SQL**　以前のバージョンでは、Embedded SQL は静的なインターフェイスでした。今回のバージョンでは、Ultra Light 動的 SQL のインターフェイスとなり、SQL Anywhere データベースは不要になりました。Embedded SQL では、動的 ESQL 文や、ホスト変数のプレースホルダーの使用もサポートしています。また、ESQL アプリケーションでは uleng10 を使用して実行できるようになりました。これを行うには、*ulrt.lib* ではなく *ulrtc.lib* に対してリンクします。

  この変更の結果、簡単な Embedded SQL アプリケーションでもサイズが大きくなったり、複雑なアプリケーションでもサイズが小さくなったりする場合があります。「Ultra Light のアップグレード」396 ページと「Embedded SQL アプリケーションの開発」『Ultra Light C/C++ プログラミング』を参照してください。

プラットフォームとデバイス

プラットフォームのサポートが変更されました。サポートされるプラットフォームのリストについては、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

特筆すべき重要な強化は次のとおりです。

- **配備プラットフォーム**　プラットフォームは次のように強化されています。
  - **Palm OS**　Palm OS デバイスの Ultra Light サポートが強化され、次のように変更されました。

    ●Palm OS v4.x 以上のランタイムサポートが提供されるようになりました。

    ●CodeWarrior の開発サポートはバージョン 9 まで拡張されました。CodeWarrior 8 のサポートは終了しました。

    ●複数のデータベースと汎用のファイル名がサポートされるようになりました。DBF 接続パラメーターを使用して、複数のデバイスに対して 1 つのデータベースを指定できるようになった。これにより、レコードベースとファイルベースのどちらの記憶領域を使用しているかによって、ファイル名が正しく設定されます。「Ultra Light DBF 接続パラメーター」『Ultra Light データベース管理とリファレンス』と「接続パラメーターでの Ultra Light ファイルパスの指定」『Ultra Light データベース管理とリファレンス』を参照してください。

    ●NVFS デバイスと VFS デバイスがサポートされるようになりました。

  - **Symbian OS**　Symbian OS サポートは、今回のバージョンの Ultra Light で新しく追加されました。Ultra Light では、UIQ Phone (2.0 と 2.1)、Nokia S60 (second edition)、Series 80 の各デバイスにおける Symbian OS バージョン 7.0 と 8.0 をサポートします。

  - **Windows Mobile 2005**　Embedded Visual C++ 3.0 または 4.0 を使用している場合は、引き続き既存のランタイムを使用できます。ただし、Visual Studio 2005 を使用してアプリケーションを構築する場合は、新しいランタイム (*¥ultralite¥ce¥arm.50* にインストールされる) が必要です。

- **開発環境サポートの強化**　開発ツールと開発言語は次のように更新されました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○Ultra Light で、Visual Studio .NET 2003 での ADO.NET 1.0 開発と、Visual Studio 2005 での ADO.NET 2.0 開発がサポートされるようになりました。

○Ultra Light では、Visual Basic と C# 開発用の AppForge Crossfire バージョン 5.6 がサポートされるようになりました。AppForge 用のアプリケーションは、Palm OS、Symbian OS、Windows CE の各プラットフォームに配備できます。

○C++ コンポーネントの開発がサポートされるようになりました。

セキュリティ

- **暗号化タイプ**　TLS で圧縮している場合、Ultra Light では ECC と RSA の両方の暗号化タイプがサポートされるようになりました。RSA 暗号化は個別の製品ではなくなりました。「トランスポートレイヤーセキュリティを使用する Ultra Light クライアントの設定」『SQL Anywhere サーバー データベース管理』を参照してください。

- **FIPS 認定の暗号化**　Mobile Link サーバーの通信を FIPS 認定の暗号で保護できるようになりました。

- **単純化されたセキュリティストリーム**　個別のセキュリティパラメーターを使用するのではなく、暗号化ストリームをネットワークプロトコルやストリームタイプとして定義できるようになりました。サポートされるストリームタイプは、TCP/IP、TLS (RSA、ECC、FIPS 用)、HTTP、HTTPS です。「Stream Type 同期パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

データベース管理

重要な新機能と強化は次のとおりです。

- **パスワードの変更**　データベースで大文字と小文字を区別するかは関係なく、すべてのパスワードで大文字と小文字を区別します。新しいデータベースは、デフォルトのユーザー ID **DBA** (パスワード **sql**) で作成されます。したがってユーザー ID、パスワード、信頼できるルート証明書は、以前のリリースからデータベースをアップグレードすると保存されません。

- **データベースプロパティと接続パラメーターの改善**　データベースプロパティと接続パラメーターが強化かつ単純化され、データベースと接続の動作を簡単に記述できるようになりました。今回のリリースのデータベースに設定できるデータベースプロパティと接続パラメーターの完全なリストについては、「Ultra Light データベースプロパティ」『Ultra Light データベース管理とリファレンス』と「Ultra Light 接続パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

- **インデックスのパフォーマンスの向上**　今回のリリースで、Ultra Light のインデックスのパフォーマンスが強化されました。インデックスのハッシュ処理が導入されたことが大きな向上点の 1 つです。Ultra Light でインデックスハッシュサイズが設定できるようになりました。ハッシュサイズを 1 ～ 32 バイトの値に設定することで、Ultra Light はインデックスページにインデックス付けされた値の一部またはすべてを格納します。これにより、必要なロー

ルックアップの回数が減少します。「Ultra Light max_hash_size 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

- **チェックサム検証**　データベースページがディスクに格納されるときにデータ整合性を検証するために、これらのページのチェックサムを含めることができるようになりました。「Ultra Light checksum_level 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

- **BLOB サポートの拡張**　Ultra Light データベースでの BLOB サポートが拡張されました。BLOB の更新、データ型のキャスト、長さの取得が可能です。

管理ツール

今回のリリースで管理ツールが強化されました。ツールを正しく使用できるように、マニュアルを確認してください。

グラフィカルな管理ツール

- **Sybase Central**　Sybase Central を使用して、Ultra Light データベースの作成、変更、管理をグラフィカルなユーザーインターフェイスから実行できるようになりました。これは ulview スキーマ編集ユーティリティを置き換えるツールです。

  Sybase Central には次のウィザードが含まれています。

  ○データベース作成ウィザードを使用して、新しい Ultra Light データベースを構築します。このウィザードは ulcreate ユーティリティと同じ機能を提供します。

  ○データベース消去ウィザードを使用して、既存の Ultra Light データベースを消去します。このウィザードと同じ機能を持つユーティリティはありません。

  ○データベース抽出ウィザードを使用して、SQL Anywhere リファレンスデータベースから新しい Ultra Light データベースを初期化します。このウィザードは ulinit ユーティリティと同じ機能を提供します。

  ○データベースロードウィザードを使用して、XML ファイルから Ultra Light データベースにデータをロードします。このウィザードは ulload ユーティリティと同じ機能を提供します。

  ○C++ API 移行ウィザードを使用して、ulgen ユーティリティ (削除されたユーティリティ) で作成された C/C++ コードをマイグレートします。このウィザードと同じ機能を持つユーティリティはありません。

  ○データベース同期ウィザードを使用して、Ultra Light データベースの同期をとります。このウィザードは ulsync ユーティリティと同じ機能を提供します。

  ○データベースアップグレードウィザードを使用して、既存の Ultra Light データベースを以前のバージョンからアップグレードします。このウィザードは ulload ユーティリティを併用した場合の ulunloadold ユーティリティと同じ機能を提供します。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○データベースアンロードウィザードを使用して、Ultra Light データベースから XML、SQL、または別のデータベースにデータ／スキーマ情報をアンロードします。このウィザードは、ulcreate ユーティリティと ulload ユーティリティの追加機能を併用した場合の ulunload ユーティリティと同じ機能を提供します。

- **Interactive SQL** Interactive SQL を使用して、Ultra Light データベースで SQL 文の開発とテストを実行できるようになりました。Interactive SQL は、以前のバージョンで使用された ulisql ユーティリティを置き換えます。「Ultra Light 用 Interactive SQL ユーティリティ (dbisql)」『Ultra Light データベース管理とリファレンス』を参照してください。

コマンドライン管理ユーティリティ

次のコマンドラインユーティリティが Ultra Light に追加されました。

- **古いデータベースのアンロードユーティリティ** 新しい ulunloadold コマンドラインユーティリティを使用すると、既存の 8.0.2 または 9.x Ultra Light データベース (スキーマとデータ) またはスキーマファイルを XML ファイルにアンロードできます。ulload コマンドラインユーティリティがあれば、Ultra Light バージョン 10 データベースを再構築するために、ユーティリティの出力を使用できます。

- **情報ユーティリティ** 新しい ulinfo ユーティリティを使用すると、Ultra Light データベースに関する情報が表示されます。また、global_id や ml_remote_id などのデータベースオプション ID を変更したりクリアしたりすることもできます。「Ultra Light 情報ユーティリティ (ulinfo)」『Ultra Light データベース管理とリファレンス』を参照してください。

また、Ultra Light 10 の新しい RDBMS 機能をサポートするように既存のコマンドラインユーティリティが強化されたため、これらのユーティリティのオプションが以前のバージョンから変更されました。新しいユーティリティを正しく使用できるように、開始前にマニュアルを確認してください。ユーティリティの完全なリファレンスノートについては、「Ultra Light ユーティリティ」『Ultra Light データベース管理とリファレンス』を参照してください。

- **エラーレポート機能の強化** Ultra Light のユーティリティは、他の SQL Anywhere ユーティリティと一貫性のあるエラーをレポートするようになりました。

- **データベース作成パラメーターの拡張** すべてのデータベース作成ユーティリティ (ulcreate や ulload など) で、拡張された作成パラメーターを使用できるようになりました。これらの拡張された作成パラメーターは、コマンドラインで -o を指定して設定でき、Sybase Central のウィザードで設定できるのと同じ機能を設定できます。「Ultra Light 作成パラメーターの指定」『Ultra Light データベース管理とリファレンス』を参照してください。

- **アンロード動作の強化** ulunload を使用して、Ultra Light データベーススキーマを動的 SQL 文のシーケンスとして出力できるようになりました。「Ultra Light データベースのアンロードユーティリティ (ulunload)」『Ultra Light データベース管理とリファレンス』を参照してください。

- **ulsync の動作の強化** ulsync を使用すると、ネットワークプロトコルオプションや拡張同期パラメーターをこのユーティリティから直接設定できます。完全なリストについては、「Ultra Light 同期パラメーターとネットワークプロトコルオプション」『Ultra Light データベース管理とリファレンス』を参照してください。

また、ulsync では、パブリケーションマスクだけではなく、パブリケーションに名前を付けることができるようになりました。キーワード **Publications** は、パブリケーション名をカンマで区切ったリストで指定します。詳細については、「Ultra Light 同期ユーティリティ (ulsync)」『Ultra Light データベース管理とリファレンス』を参照してください。

- **コンジットインストールの強化**　HotSync コンジットインストールユーティリティ (ulcond10) で、コンジットの拡張機能、接続文字列、複数のデータベースがサポートされるようになりました。
- **ulmvbreg**　Ultra Light for AppForge を登録するための ulmvbreg ユーティリティは、ulafreg に名前が変更されました。このユーティリティは *%SQLANY10%¥win32* ディレクトリにインストールされるようになりました。

ULSQLCONNECT

以前は、すべての Ultra Light ユーティリティがコマンドラインから接続情報を受け取っていました。今回のバージョンで、デフォルトのユーザー ID とパスワード以外の情報を渡す場合に、ホストコンピューターで ULSQLCONNECT 環境変数を設定できるようになりました。「Ultra Light 接続パラメーターと ULSQLCONNECT 環境変数」『Ultra Light データベース管理とリファレンス』を参照してください。

SQL

- **SQL 文**　Ultra Light で、いくつかの新しい文がサポートされるようになりました。次のような新しい文があります。
  - **ALTER TABLE**　Ultra Light SQL を使用してテーブルを作成するだけでなく、この文を使用して定義を変更できるようになりました。「ALTER TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。
  - **ALTER/CREATE/DROP PUBLICATION**　これら 3 つの文を使用したパブリケーションの追加、作成、削除がサポートされるようになりました。「ALTER PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』、「CREATE PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』、「DROP PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。
  - **START/STOP SYNCHRONIZATION DELETE**　Ultra Light SQL にこれらの文が含まれるようになりました。これらの文を使用して、Mobile Link 同期で削除がロギングされる方法を制御します。「START SYNCHRONIZATION DELETE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』と「STOP SYNCHRONIZATION DELETE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。
- **名前付き制約**　ALTER TABLE 文と CREATE TABLE 文を使用して、テーブル制約に名前を付けることができるようになりました。このため、テーブル全体の制約を変更するのではなく、個々の制約を変更することで、テーブルやカラムの制約を変更できます。「ALTER TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』と「CREATE TABLE 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。
- **その他の SELECT 文の強化**　SELECT 文が拡張されました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○SELECT 文の TOP 句に START AT を指定できるようになりました。START AT を指定すると、結果セットを明示的に制限するクエリの中で、さらに柔軟性を高めることができます。「SELECT 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』を参照してください。

○DISTINCT 句が拡張され、この句で集合関数 (SUM、AVERAGE、MAX など) を使用できるようになりました。この句で集合関数を使用すると、実行時間が大幅に増加します。「SELECT 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』と「SQL 関数」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **UNION 演算子** UNION 演算子を使用すると、2 つ以上のクエリから単一の結果セットを構築できます。デフォルトでは、UNION 演算子は、結果セットから重複しているローを削除します。「UNION 句 : 結果セットの結合」『SQL Anywhere サーバー SQL の使用法』を参照してください。

同期

- **設定可能で増加された HotSync コンジット同期のデフォルトキャッシュサイズ** 以前は、Palm OS ファイルベースのデータストアで同期されたデータが一定量を超えると、同期速度に悪影響を与えていました。これが、Ultra Light コンジット用のデフォルトキャッシュサイズ (デスクトップ上) は 4 MB に増加されました。キャッシュサイズが大幅に増加したため、不要なファイル I/O 操作が減少し、同期時間が向上しました。ただし、選択した場合は異なるデフォルトキャッシュサイズを設定することもできます。

- **パブリケーションの述部** Ultra Light 用の同期パブリケーションで述部が受け入れられるようになりました。条件式を論理演算子 AND や OR とオプションで組み合わせる場合、WHERE 句または HAVING 句に条件のセットを定義できるようになりました。SQL Anywhere と同様に、UNKNOWN と評価される述部が FALSE として解釈されます。「CREATE PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』と「ALTER PUBLICATION 文 [Ultra Light]」『Ultra Light データベース管理とリファレンス』。

- **Mobile Link クライアントネットワークレイヤーの向上** 次のクライアントネットワークレイヤーが改善されました。

  ○すべてのプロトコルで同期圧縮できます。

  ○永続的な接続によって、同じ接続で複数回同期できるようになりました。

  ○IPv6 サポートが導入されました。

  ○エラー検出とデバッグ機能が改善されました。

  Ultra Light を Mobile Link へのクライアントとして使用する場合の詳細については、「Ultra Light クライアント」『Ultra Light データベース管理とリファレンス』を参照してください。

- **同期用のテーブルの順序の設定** Ultra Light クライアントからの同期に、テーブルのアップロード時に参照整合性の問題を避けるためにテーブルの順序を指定する機能が備わりまし

た。同期のためにテーブルの順序を指定する場合は、table_order 同期パラメーターを使用します。詳細については、次の項を参照してください。

- ULSyncParms クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』
- 「Additional Parameters 同期パラメーター」『Ultra Light データベース管理とリファレンス』
- ul_sync_info 構造体 [Ultra Light C および Embedded SQL データタイプ]『Ultra Light C/C++ プログラミング』
- SyncParms クラス [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』
- ULGetSyncResult メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』

プログラミングインターフェイス

一般的な向上点

- **カーソルの更新**　Ultra Light アプリケーションでは、カーソル処理中にデータベース内のデータを変更できる機能がサポートされるようになりました。SQL Anywhere データベースと同様に、すべてのクエリの結果セットで、カーソルの更新と削除ができるわけではありません。カーソルの更新が許可され、実行される場合について理解する必要があります。「データのフェッチ」『Ultra Light C/C++ プログラミング』を参照してください。

- **単純化された接続文字列**　デフォルトのユーザー ID **DBA** とパスワード **sql** は常に Ultra Light によって提供されるため、接続文字列にデータベースを指定するだけで接続できるようになりました。さらに、ほとんどのデータベースを DBF 接続パラメーターで設定できます。「Ultra Light データベースへの接続」『Ultra Light データベース管理とリファレンス』を参照してください。

- **MLFileTransfer 関数の導入**　ファイル転送関数を使用すると、Mobile Link ファイル転送ユーティリティでファイルをダウンロードできます。ダウンロードされるファイルは、Mobile Link ユーザー名ごとに変えることも、デフォルトのファイルにすることもできます。たとえば、(月または処理サイクルの頭に) ローカルデータベースを置き換えるために、事前に設定された空のデータベースファイルをダウンロードするように選択できます。

  注意
  Ultra Light for M-Business Anywhere：なし

  次の項を参照してください。

  - 「Mobile Link ファイル転送ユーティリティ (mlfiletransfer)」『Mobile Link クライアント管理』
  - MLFileDownload メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - MLFileUpload メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULFileTransfer クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』
  - ULFileTransferProgressData クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』

- **データベースの作成**　Ultra Light スキーマは、個別の *.usm* ファイルではなく、データベースの一部になりました。このため、以前のバージョンでサポートされていたのと同じ方法では、アプリケーションが新しいデータベースを作成できなくなりました。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

次の項を参照してください。

- ULCreateDatabase メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
- ULDatabaseManager クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』
- DatabaseManager.createDatabase メソッド [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』

Ultra Light for C/C++

- **Symbian OS のサポート**　Symbian OS プラットフォーム用に、CodeWarrior または Carbide C++ 開発環境を使用した Ultra Light for C/C++ がサポートされるようになりました。

- **新しい関数**　今回のリリースには、さまざまな新しい関数が追加されています。これらの関数は次のとおりです。

  ○GetPublicationMask 関数は、特定のパブリケーション名のパブリケーションマスクを取得します。

  ○特定のネットワークプロトコルで同期する前に、適切な ULEnable*Synchronization 関数を呼び出す必要があります。詳細については、次の項を参照してください。

  - ULEnableAesDBEncryption メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableAesFipsDBEncryption メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableEccE2ee メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableEccSyncEncryption メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableRsaE2ee メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableHttpSynchronization メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableRsaFipsE2ee メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableRsaFipsSyncEncryption メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableRsaSyncEncryption メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableTcpipSynchronization メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』
  - ULEnableZlibSyncCompression メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』

- **ワイド文字とナロー文字 (ASCII) のサポートの向上**　Ultra Light のデータベースファイルフォーマットは 1 種類 (ナロー文字) になりましたが、アプリケーションではワイド定義 TCHAR を使用し続けることができます。必要に応じて、ワイド文字とそれに等価な MBCS 文字の間で変換が行われます。

● **強化された関数の変更**　次のように、既存の関数が強化されました。

○アプリケーションで SQL サポートが必要ない場合、ULInitDatabaseManager の代わりに ULInitDatabaseManagerNoSQL 関数を使用すると、アプリケーションのサイズが大幅に減少します。

○SetReadPosition 関数が強化され、2 つ目のパラメーター offset_in_chars を取るようになりました。このパラメーターは、オフセットがバイト単位か文字単位かを示します。

○SetSynchInfo で自動コミットが実行され、すべての同期情報がすぐに保存されるようになりました。

○ULStoreDefragInit、ULStoreDefragFini、ULStoreDefragStep は不要になりました。Ultra Light がデータベーストアの断片化解除を内部的に管理するようになりました。

○ユーザー認証は常に有効になるなので、ULEnableUserAuthentication 関数は廃止される予定です。Ultra Light では、データベースに接続する可能性のあるユーザー名を最大 4 つまで定義できるようになりました (デフォルトのユーザー名は DBA、パスワードは sql)。

○Mobile Link 同期コードで、InitSynchInfo の呼び出し前に ULEnableTcpipSynchronization を呼び出さなければならなくなりました。ULConnection.InitSyncInfo メソッド [Ultra Light C++]『Ultra Light C/C++ プログラミング』を参照してください。

Ultra Light Embedded SQL

Ultra Light Embedded SQL は静的な API ではなくなり、リファレンスデータベースが必要ではなくなりました。SQL プリプロセッサーでは、ソースファイルだけが必要です。SQL プリプロセッサーは、Ultra Light に SQL 文を送信する関数を生成します。以前のリリースでサポートされていたいくつかの SQL 文は、Ultra Light SQL でサポートされなくなりました。バージョン 10 では、以前のリリースでサポートされていなかった動的 SQL 文をサポートします。

● **新しい関数**　今回のリリースには、さまざまな新しい関数が追加されています。これらの関数は次のとおりです。

○ULEnableZlibSyncCompression 関数は同期中の zlib 圧縮を有効にします。ULEnableZlibSyncCompression メソッド [Ultra Light Embedded SQL]『Ultra Light C/C++ プログラミング』と「Mobile Link クライアントネットワークプロトコルオプション」『Mobile Link クライアント管理』を参照してください。

> **注意**
> zlib 圧縮は、Palm OS または Symbian OS ではサポートされていません。

● **強化された関数**　次のように、既存の関数が強化されました。

○GetSQLColumnName が UltraLite_RowSchema_iface に追加されました。関数が返す値は、スキーマの種類によって異なります。

●UltraLite_TableSchema で使用する場合、この関数は column_id パラメーターで指定されたカラム名を返します。

UltraLite_ResultSetSchema で使用する場合、この関数は次の値を返します。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○当該の結果セットカラムでエイリアス名が指定されている場合は、エイリアス名

○結果セットカラムがテーブルのカラムを表す場合は、カラム名

○それ以外の場合は、すべて空の文字列

Ultra Light.NET

Ultra Light で、Visual Studio .NET 2003 での ADO.NET 1.0 開発と、Visual Studio 2005 での ADO.NET 2.0 開発がサポートされるようになりました。

- **新しいメソッド**　今回のリリースには、さまざまな新しい関数が追加されています。これらの関数は次のとおりです。

  ○ExecuteResultSet メソッドは SQL SELECT 文を実行して、更新可能な結果セットを ULResultSet クラスとして返します。ULCommand.ExecuteResultSet メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○ULResultSet クラスには、Append*、Set*、Delete、Update の各メソッドが含まれます。これらのメソッドの詳細については、ULResultSet クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○Ultra Light.NET で、TCP/IP 同期時の TLS がサポートされるようになりました。

  ○ConnectionString プロパティと ULConnectionParms オブジェクトが強化され、限定的な引用符使用がサポートされるようになりました。ULConnectionParms クラス [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○GetPublicationPredicate メソッドは、指定されたパブリケーションのパブリケーション述部文字列を返します。パブリケーションが存在しない場合は、SQLE_PUBLICATION_NOT_FOUND が設定されます。ULTableSchema.GetPublicationPredicate メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○SignalSyncIsComplete メソッドは、ActiveSync 用の Mobile Link プロバイダーに対して、アプリケーションが同期を完了したことを通知します。ULDatabaseManager.SignalSyncIsComplete メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

  ○SetDatabaseOption メソッドは、指定されたデータベースオプションの値を設定します。ULDatabaseSchema.SetDatabaseOption メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

- **強化されたメソッド**　次のように、既存のメソッドが強化されました。

  ○ULSyncParms クラスに TableOrder 順序プロパティが用意されました。このプロパティは、統合データベースにテーブルがアップロードされる順序を指定します。ULSyncParms.AdditionalParms プロパティ [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

○GetSchemaTable メソッドは、拡張 Table メタデータを返すようになりました。完全なリストについては、ULDataReader.GetSchemaTable メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

○テーブルが UL_TABLE_ACCESS_MODE_NONE または UL_TABLE_ACCESS_MODE_FIND_AGAIN の状態である場合、UpdateBegin メソッドは ResultSet レベルのオプションになりました。Ultra Light.NET API が ADO.NET 2.0 結果セットとの互換性を持たせるために、この変更が必要でした。ULResultSet.UpdateBegin メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

○GetDatabaseProperty メソッドが認識するプロパティが増えました。ULDatabaseSchema.GetDatabaseProperty メソッド [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

○ULSyncProgressData クラスに Flags プロパティが含まるようになりました。ULSyncProgressData.Flags プロパティ [Ultra Light.NET]『Ultra Light .NET プログラミング』を参照してください。

Ultra Light for AppForge Crossfire

Ultra Light for AppForge で Symbian OS プラットフォームがサポートされるようになりました。今回のリリースでは、Ultra Light エンジンのサポートが追加され、複数のアプリケーションが 1 つのデータベースに同時にアクセスできるようになりました。

- **新しいメソッド** OnWaiting メソッドは、ユーザーアプリケーションが GUI イベントを処理し、必要に応じて現在の操作をキャンセルするためのメカニズムを提供します。

Ultra Light for M-Business Anywhere

- **新しいメソッド** 今回のリリースには、さまざまな新しいメソッドが追加されています。これらの関数は次のとおりです。

  ○setMBAServerWithMoreParms メソッドは、ワンタッチ同期を使用するときにプロキシサーバー情報を設定します。この新しいメソッドは、新しい文字列引数 **additional** が追加されたことで、既存の setMBAServer メソッドを強化します。

  ○getPublicationMask メソッドは、特定のパブリケーション名のパブリケーションマスクを取得します。

  ○getPublicationPredicate メソッドは、指定されたパブリケーションのパブリケーション述部文字列を返します。パブリケーションが存在しない場合は、SQLE_PUBLICATION_NOT_FOUND が設定されます。

- **強化されたメソッド** 既存のメソッドが次のように強化されました。

  ○setStream メソッドで、TLS (トランスポートレイヤーセキュリティ) の ECC (楕円曲線暗号) がサポートされるようになりました。SyncParms クラス [Ultra Light for M-Business Anywhere]『Ultra Light M-Business Anywhere プログラミング (旧式)』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

注意
ECC 暗号化は、すべてのプラットフォームで使用できるわけではありません。サポートされるプラットフォームのリストについては、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

動作の変更と廃止予定機能

次に、バージョン 10.0.0 で導入された Ultra Light の変更を示します。

廃止予定のプラットフォーム

●今回のリリースで、PocketPC 2000 OS のサポートは廃止される予定です。

●CodeWarrior 8 はサポートされなくなりました。代わりに CodeWarrior 9 を使用してください。

●Windows CE MIPS プロセッサーはサポートされなくなりました。

削除されたコンポーネント、モジュール、ネームスペース

今回のリリースから、次のプログラミングインターフェイスが削除されました。

- **Ultra Light ActiveX**　サポートされる API を使用して、すべてのアプリケーションを書き直す必要があります。
- **静的型 Java API**　サポートされる API を使用して、すべてのアプリケーションを書き直す必要があります。
- **Native Ultra Light for Java**　サポートされる API を使用して、すべてのアプリケーションを書き直す必要があります。
- **静的型 C++ API と静的型 Embedded SQL**　C++ アプリケーションを作成する開発者は、動的 C++ インターフェイスを使用してプログラミングする必要があります。以前のバージョンから静的型 C++ ライブラリで作成されたアプリケーションがある場合、Ultra Light 10 にはこの新しいライブラリへの移動作業を簡単にする、マイグレーションユーティリティが含まれます。「Ultra Light のアップグレード」 396 ページを参照してください。
- **iAnywhere.UltraLite ネームスペース**　Ultra Light.NET では、このネームスペースがサポートされなくなりました。代わりに iAnywhere.Data.UltraLite ネームスペースを使用して、アプリケーションを書き直す必要があります。

削除されたユーティリティ

- **スキーマペインタ**　Ultra Light データベースの作成でスキーマファイルが不要になったため、スキーマペインタツールは削除されました。
- **データベース変換ツール**　データベース変換ツール (ulconv ユーティリティ) はサポートされなくなりました。ulconv の機能を実行する場合は、ulcreate、ulload、ulsync、ulunload の各ユーティリティを使用してください。

- **ulxml ユーティリティ**　スキーマファイルを XML に変換する ulxml ユーティリティはサポートされなくなりました。ulxml に似た機能を実行する場合は、ulload と ulunload を使用して、データベースを XML に変換してください。

- **ulisql**　ulisql ユーティリティは、サポートされなくなりました。代わりに、Interactive SQL (dbisql) で Ultra Light がサポートされるようになりました。

- **ulgen**　ulgen ユーティリティは、サポートされなくなりました。このユーティリティを使用する Ultra Light 配備では、データベースと C/C++ アプリケーションをアップグレードする必要があります。「Ultra Light のアップグレード」396 ページを参照してください。

削除、廃止予定、変更された関数

- **Ultra Light for C/C++ API**　C/C++ API で変更された関数やマクロは次のとおりです。

  ○*.usm* ファイルがなくなったため、データベーススキーマに接続したり、データベーススキーマを動的にアップグレードすることはできなくなりました。このような従来の Ultra Light の機能に関連するすべてのクラスと関数は削除されました。

  ○今回のバージョンでは、ULEnablePalmRecordDB と ULEnableFileDB が削除されました。

  ○すべての ULEnableXXXX 関数は、初期化された SQLCA で呼び出されなければならなくなりました。

  ○マクロ UL_STORE_PARMS はリリース 10 で非推奨になりました。OpenConnection または CreateDatabase の呼び出し時に、接続または作成のオプションは適切なパラメーターで指定されます。

  ○今回のリリースで、ULSecureCerticomTLSStream と ULSecureRSATLSStream は廃止される予定です。これらに代わり、ULEccTlsStream と ULRsaTlsStream を使用できます。

  ○ul_sync_info の security フィールドと security_parms フィールドが削除されました。代わりに、ストリームフィールドに適切な文字列 tcpip、http、https、または tls を設定してください。また、セキュリティパラメーターを他のストリームパラメーターと組み合わせてください。TCPIP は常に基本となるトランスポートメカニズムです。HTTP 上の TLS はサポートされなくなりました。代わりに、HTTPS 同期ストリームを使用できます。「Ultra Light 同期パラメーターとネットワークプロトコルオプション」『Ultra Light データベース管理とリファレンス』を参照してください。

  ○ULSocketStream、ULHTTPStream、ULHTTPSStream は、必要な文字列を適切に返すように変更されました。

  ○Windows CE デバイスのための ULActiveSyncStream は削除されました。Windows メッセージハンドラーで同期メッセージを受信する場合、ActiveSync プロバイダーに登録されている Ultra Light アプリケーションでは、ULActiveSyncStream の代わりに標準の同期ストリームを使用してください。

- **Embedded SQL**　C/C++ API に対する Embedded SQL インターフェイスの関数は次のように変更されました。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○*.usm* ファイルがなくなったため、データベーススキーマを動的にアップグレードできなくなりました。このような従来の Ultra Light の機能に関連するすべてのクラスと関数は削除されました。

- **Ultra Light.NET API**　Ultra Light.NET API の関数は次のように変更されました。

  ○*.usm* ファイルがなくなったため、データベーススキーマに接続したり、データベーススキーマを動的にアップグレードすることはできなくなりました。このような従来の Ultra Light の機能に関連するすべてのクラスとメソッドは削除されました。

  ○ULConnectionParms クラスで ParmsUsed プロパティの名前が ToString に変更されました。

  ○GetSQLColumnName の名前が GetColumnSQLName に変更されました。

  ○ULStreamType のメンバー UNKNOWN と ACTIVE_SYNC はこの列挙体から削除されました。デフォルトは ULStreamType.TCPIP になりました。

- **Ultra Light for MobileVB API**　MobileVB API のメソッドは次のように変更されました。

  ○*.usm* ファイルがなくなったため、データベーススキーマに接続したり、データベーススキーマを動的にアップグレードすることはできなくなりました。このような従来の Ultra Light の機能に関連するすべてのクラスとメソッドは削除されました。

- **Ultra Light for M-Business Anywhere API**　M-Business Anywhere API の関数は次のように変更されました。

  ○*.usm* ファイルがなくなったため、データベーススキーマに接続したり、データベーススキーマを動的にアップグレードすることはできなくなりました。このような従来の Ultra Light の機能に関連するすべてのクラスとメソッドは削除されました。

名前の変更

- **ULUtil**　Palm OS 用の ULUtil ユーティリティの名前は ULDBUtil に変更されました。

- **ulmvbreg**　ulmvbreg の名前は ulafreg に変更されました。このユーティリティは *%SQLANY10%¥win32* ディレクトリにインストールされるようになりました。

その他

- **ulcond.log**　バージョン 10 の Ultra Light HotSync コンジットインストーラー (ulcond10) は、このログファイルにメッセージを書き込まなくなりました。

Sybase Central と Interactive SQL

次の項では、バージョン 10.0.0 の Sybase Central と Interactive SQL での新機能、動作の変更、廃止予定機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された Sybase Central と Interactive SQL の追加機能を示します。

Sybase Central

この項では、Sybase Central の新機能について説明します。Sybase Central プラグインに対する変更点や追加機能は、次の項で説明します。

- 「SQL Anywhere プラグイン」354 ページ
- 「Sybase Central の新しいプラグイン」355 ページ

- **Sybase Central タスクリスト** Sybase Central の左ウィンドウ枠で、データベースのツリー構造ではなく、タスクを表示できます。タスクリストには、現在選択されているオブジェクトに関連した共通のタスクが表示されます。タスクリストには、共通のタスク、ナビゲーションオプション、マニュアルへのリンクがあります。「Sybase Central のナビゲーション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい接続メニュー** 以前のリリースでは、[F11] キーを押して [新しい接続] ウィンドウを開き、接続するプラグインを選択できました。今回のリリースでは、[接続] メニューが用意され、使用するプラグインを選択できるようになりました。[新しい接続] ウィンドウは削除されましたが、[F11] キーを押すと、設定に使用するプラグインを選択できる [接続] メニューが開きます。

- **接続プロファイルの強化** Sybase Central 接続プロファイルに説明を追加できるようになりました。接続プロファイルはインポートやエクスポートもできます。

- **プラグインの検索** Sybase Central で [表示] » [検索ウィンドウ枠] を選択することで、プラグインとデータベースから指定されたテキストを含むオブジェクトを検索できるようになりました。

- **コンテキストドロップダウンリスト** 新しい [コンテキスト] ドロップダウンリストには、現在オブジェクトツリーで選択されているオブジェクトが表示され、プラグイン間を簡単にナビゲーションできます。これは、左ウィンドウ枠でオブジェクトツリーを開いていない場合に特に便利です。

SQL Anywhere プラグイン

- **[デッドロック] データベースタブ** Sybase Central で SQL Anywhere データベースに接続しているときに、[デッドロック] タブでデッドロックについての情報を表示できます。「Sybase Central からデッドロックの表示」『SQL Anywhere サーバー SQL の使用法』を参照してください。

- **ER (実体関連) 図** Sybase Central では、データベースの ER (実体関連) 図が表示され、データベース内のテーブルとその外部キーの関係が示されます。「SQL Anywhere 12 プラグインからの ER 図の表示」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいスケジュール作成ウィザードとその他の [イベント] フォルダーの強化** スケジュールされたイベントの場合は、イベントがトリガーされる次のスケジュールされた時刻が [イベント] フォルダーに表示されるようになりました。条件イベントの場合は、このフォルダーには

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

システムイベントと、オプションでそのイベントがトリガーされるトリガー条件が表示されます。スケジュールは新しいスケジュール作成ウィザードを使用して作成されます。新しいスケジュールされたイベントを作成するときは、イベントウィザードでイベントを作成できますが、必要に応じて後から追加スケジュールをイベントに追加できます。「スケジュール定義」『SQL Anywhere サーバー データベース管理』を参照してください。

- **メンテナンスプラン**　データベースの検証とバックアップを自動的に行うスケジュールを設定でき、その出力ログを電子メールで自分に送信できます。「メンテナンスプランの作成」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しいデータベースリストアウィザード**　データベースリストアウィザードを使用して、アーカイブバックアップからデータベースをリストアできるようになりました。「アーカイブバックアップからのリストア」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Sybase Central と Interactive SQL のエディターの強化**　Sybase Central と Interactive SQL の [オプション] ウィンドウの [フォーマット] タブで、エディターウィンドウで使用するフォントを選択できます。

  データベースオブジェクト名の入力補完オプションがエディターに追加されました。「テキスト補完の使用」『SQL Anywhere サーバー データベース管理』を参照してください。

Sybase Central の新しいプラグイン

- **QAnywhere プラグイン**　QAnywhere プラグインは、QAnywhere アプリケーションの作成と管理に使用する、使いやすいグラフィカルなインターフェイスを提供します。

  「Sybase Central 用の新しい QAnywhere プラグイン」332 ページを参照してください。

- **Ultra Light プラグイン**　Ultra Light プラグインを使用して、Ultra Light データベースの作成、変更、管理をグラフィカルなユーザーインターフェイスから実行できるようになりました。

  「グラフィカルな管理ツール」342 ページを参照してください。

- **Mobile Link の同期モデル作成ウィザードとモデルモード**　ウィザードを使用して同期モデルを作成したり、新しいモデルモードを使用して Mobile Link プラグインのモデルを編集したりすることができるようになりました。Mobile Link サーバー起動同期も設定できます。Mobile Link プラグインの従来の機能も強化され、管理モードで保存されます。

  「Sybase Central の Mobile Link プラグインの強化」308 ページを参照してください。

Interactive SQL

- **Interactive SQL による Ultra Light データベースへの接続**　Interactive SQL を使用して、Ultra Light データベースで SQL 文の開発とテストを実行できるようになりました。ulisql ユーティリティは廃止される予定です。「グラフィカルな管理ツール」342 ページを参照してください。

- **Interactive SQL とサードパーティのソース制御システムの統合**　Interactive SQL はサードパーティのソース制御システムと統合でき、Interactive SQL 内からファイルに対する一般的な

ソース制御操作 (チェックイン、チェックアウト、古いバージョンとの比較など) を実行できます。「Interactive SQL でのソース制御の統合」『SQL Anywhere サーバー データベース管理』を参照してください。

- **新しい Interactive SQL オプション** isql_maximum_displayed_rows オプションを使用すると、Interactive SQL の結果セットに表示されるローの数を指定できます。また、isql_show_multiple_result_sets オプションは、Interactive SQL の [結果] ウィンドウ枠に複数の結果セットを表示できるかどうかを指定します。「isql_maximum_displayed_rows オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』と「isql_show_multiple_result_sets オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』を参照してください。

- **テキスト補完** Interactive SQL に入力補完オプションが備わり、テーブル、ビュー、カラム、ストアドプロシージャー、システム関数の名前を補完できるようになりました。「テキスト補完の使用」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Interactive SQL での DESCRIBE 文のサポート** DESCRIBE 文を使用すると、指定されたテーブルやプロシージャーについて次の情報を取得できます。

  ○ テーブルで見つかったすべてのカラム
  ○ テーブルで見つかったすべてのインデックス
  ○ ストアドプロシージャーで使用されるすべてのパラメーター

  「DESCRIBE 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **Interactive SQL での @data オプションのサポート** Interactive SQL をコマンドプロンプトから開始するときに *@data* オプションを指定すると、指定の環境変数や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。「Interactive SQL ユーティリティ (dbisql)」『SQL Anywhere サーバー データベース管理』を参照してください。

SQL Anywhere コンソールユーティリティ

- **SQL Anywhere での @data オプションのサポート** SQL Anywhere コンソールをコマンドプロンプトから開始するときに *@data* オプションを指定すると、指定の環境変数や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。「SQL Anywhere コンソールユーティリティ (dbconsole)」『SQL Anywhere サーバー データベース管理』を参照してください。

動作の変更と廃止予定機能

次に、バージョン 10.0.0 で導入された Sybase Central と Interactive SQL の変更を示します。

- **Interactive SQL で quoted_identifier オプションが On に設定しなくなった** 以前のバージョンのソフトウェアでは、Interactive SQL は quoted_identifier オプションを On に設定していました。このオプションについては、データベースの設定が使用されるようになりました (デフォルトではこのオプションは On)。「quoted_identifier オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **isql_plan オプションは NONE パラメーターをサポートしなくなった**　isql_plan オプションの NONE パラメーターは、サポートされなくなりました。

- **Interactive SQL から返される結果セット数に制限はない**　以前のバージョンのソフトウェアでは、複数の結果セットを返すクエリを実行すると、Interactive SQL に表示される結果セットは最大で 10 個でした。今回のバージョンでは、クエリで返されるすべての結果セットが表示されるようになりました。「isql_show_multiple_result_sets オプション [Interactive SQL]」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Interactive SQL の -f オプションの動作の変更**　Interactive SQL を -f オプションを使用して起動しても、データベースへの接続は自動的には確立されません。以前は、接続が自動的に開かれていました。

- **Interactive SQL におけるグラフィカルなプランへのアクセスと保存**

  ○グラフィカルなプランを開いたり保存したりするための 2 つの新しいメニュー項目 [プランを開く] と [プランの保存] が Interactive SQL の [ファイル] メニューから使用できるようになりました (以前は、SQL 文を開いたり保存したりする場合と同じメニュー項目 [開く] と [保存] を使用していました)。

  ○以前は、グラフィカルなプランはファイル拡張子 *.xml* で保存されていました。今回のバージョンでは、拡張子 *.saplan* で保存されるようになりました。ただし、以前の *.xml* ファイル拡張子で保存されたグラフィカルなプランを表示できるように、この拡張子も引き続きサポートされます。

  ○[オプション] ウィンドウで、*.sql* ファイルと *.saplan* (グラフィカルなプラン) ファイルのデフォルトのエディターとして Interactive SQL を指定できるようになりました。

- **SET OPTION 文の PUBLIC キーワードは廃止予定**　SET OPTION 文を使用して Interactive SQL を設定するための PUBLIC キーワードは廃止される予定です。「Interactive SQL オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

- **EXIT 文によって現在の Interactive SQL ウィンドウが閉じられる**　以前のリリースでは、EXIT 文を Interactive SQL から実行すると、すべての Interactive SQL ウィンドウが閉じられました。今回のバージョンで、EXIT 文が実行されたウィンドウだけが閉じるようになりました。「EXIT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **SQLE_ENGINE_NOT_MULTIUSER の新しいエラーコード**　SQLE_ENGINE_NOT_MULTIUSER を処理するようにプログラムされたアプリケーションでは、新しいエラーコードのチェックを行う必要があります。以前は、アプリケーションがデータベースで書き込み操作を行おうしたときに、別のスレッドがアップロードを Mobile Link に送信していた場合、SQLE_ENGINE_NOT_MULTIUSER が返されていました。今回のバージョンでは、より正確な新しいエラーコード SQLE_ULTRALITE_WRITE_ACCESS_DENIED が返されるようになりました。「書き込みアクセスが拒否されました。」『エラーメッセージ』を参照してください。

- **Sybase Central プラグインの [ユーティリティ] タブの削除**　SQL Anywhere、Mobile Link、Ultra Light の各プラグインの [ユーティリティ] タブは [ツール] ボタンで置き換えられました。ユーティリティには Sybase Central の [ツール] メニューからもアクセスできます。

廃止予定機能とサポート終了機能

- **jConnect による Sybase Central、Interactive SQL、SQL Anywhere コンソールへの接続サポート終了**　Sybase Central、Interactive SQL、SQL Anywhere コンソール ユーティリティ (dbconsole) で、jConnect を使用した SQL Anywhere データベースへの接続がサポートされなくなりました。iAnywhere JDBC ドライバーを使用して、これらのアプリケーションからデータベースに接続することはできます。この変更に伴い、次の機能が削除されました。

  ○Interactive SQL ユーティリティ (dbisql) の -jconnect オプションと -odbc オプションが削除されました。

  ○SQL Anywhere コンソールユーティリティ (dbconsole) の -jconnect オプションと -odbc オプションが削除されました。

  ○Interactive SQL、SQL Anywhere コンソール、および Sybase Central 内の SQL Anywhere プラグインと Mobile Link プラグインに接続するための [接続] ウィンドウで、jConnect と iAnywhere JDBC ドライバーのどちらを使用するかを指定できなくなりました。すべての接続で iAnywhere JDBC ドライバーが使用されます。

- **Ultra Light プランが Interactive SQL の [プラン] タブに表示される**　以前のリリースでは、Interactive SQL の [結果] ウィンドウ枠に [Ultra Light プラン] タブがあり、このタブに Ultra Light プランの最適化方式が表示されていました。[Ultra Light プラン] タブは削除され、Interactive SQL から Ultra Light データベースに接続している場合は、[プラン] タブに表示されるようになりました。

- **Sybase Central の SQL Anywhere プラグインでバージョン 7 データベースのサポート終了**　バージョン 7 のデータベースサーバーと、バージョン 7 のソフトウェアで作成されたデータベースのサポートが SQL Anywhere プラグインから削除されました。データベースをアンロードして、再ロードファイル、新しいデータベース、または既存のデータベースに再ロードする場合、バージョン 8 以降のデータベースサーバーで稼働している、バージョン 5、6、または 7 のソフトウェアで作成したデータベースには引き続き接続できます。「バージョン 9 以前のデータベースのバージョン 12 用への再構築」373 ページを参照してください。

- **isql_log オプションは廃止予定**　Interactive SQL セッション中に実行された文をロギングするための isql_log オプションは廃止される予定です。代わりに、START LOGGING 文と STOP LOGGING 文を使用してください。「Interactive SQL での文のロギング」『SQL Anywhere サーバー データベース管理』を参照してください。

- **Sybase Central ファイル名の変更**　Sybase Central の説明で示したファイルの変更点のほかに、次のファイルが追加されました。

新しい名前	以前の名前
asaplugin.jar	*saplugin.jar*

Sybase Central の新しいバージョンを反映するように、レジストリキーも次のように変更されました。

HKLM¥SOFTWARE¥Sybase¥Sybase Central¥5.0 (registry entries)

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

マニュアルの強化

既存の機能の説明が、次に示すさまざまな分野で強化されています。

- **SQL 文のコンテキスト別ヘルプ**　Interactive SQL では、SQL 文の名前を右クリックして、その文の参照トピックを開くことができるようになりました。
- **Mobile Link のクイックスタート**　新しいユーザーが Mobile Link を使用して分散アプリケーションを開発する方法を理解できるように、新しい入門マニュアルが追加されました。

  Mobile Link クイックスタートを参照してください。
- **『データベース管理ガイド』に SQL Anywhere 管理ツールについての新しい章**　Sybase Central、Interactive SQL、SQL Anywhere コンソールユーティリティの使用法に重点を置く新しい章が追加されました。

  「SQL Anywhere グラフィカル管理ツール」『SQL Anywhere サーバー データベース管理』を参照してください。
- **EBF 用に更新されたサポートされているプラットフォームの情報**　製品をインストールすると、該当のインストールビルドに関する説明を含んだ、サポートされているプラットフォームのページもインストールされるようになりました。これらのページは、SQL Anywhere インストール環境の *documentation¥ja* サブフォルダーにインストールされます。これらのページは、EBF で更新されます。マニュアルには、必要に応じてこれらのページへのリンクが含まれています。
- **SQL Anywhere のセキュリティ機能の説明を移動**　『SQL Anywhere Studio セキュリティガイド』というマニュアルは削除されました。SQL Anywhere のセキュリティ機能については、SQL Anywhere サーバー データベース管理に記載されています。
- **ODBC ドライバーに関するマニュアルの削除**　『Mobile Link の ODBC ドライバー』というマニュアルは削除されました。

製品全体の機能

次の項では、SQL Anywhere バージョン 10.0.0 のすべてのコンポーネントに影響する新機能、動作の変更、廃止予定機能について説明します。

新機能

次に、バージョン 10.0.0 で導入された製品全体の追加機能を示します。

- **製品名が SQL Anywhere 10 に変更**　SQL Anywhere Studio は SQL Anywhere 10 に、Adaptive Server Anywhere は SQL Anywhere に、それぞれ名前が変更されました。これに伴って、ファイル、ディレクトリ、サービス、実行可能ファイルの多くも名前が変更されました (ほとんどは ASA から SA への変更を反映したものです)。これらの変更の詳細については、この章の関連する「動作の変更」トピックで説明しています。

- **DataWindow .NET の新しいインストール**　DataWindow. NET は、Visual Studio を使用するデータベース開発者に役立つ、カスタム制御ツールです。SQL Anywhere インストール中にオプションのコンポーネントとして提供されます。ツールの完全なマニュアルはインストール環境に保存されます。

- **RSA が SQL Anywhere に付属**　RSA 暗号化を使用するためのライセンスを別途購入する必要がなくなりました。「トランスポートレイヤーセキュリティ」『SQL Anywhere サーバー データベース管理』を参照してください。

- **機能の統計の収集**　SQL Anywhere 10 は、ソフトウェアを実行するコンピューターの情報 (オペレーティングシステムデータベースサーバーの起動オプション、使用中の SQL Anywhere 10 ソフトウェアビルドなど) や、使用中の SQL Anywhere 10 機能の情報を追跡します。致命的なエラーが発生すると、問題についての情報と、SQL Anywhere 機能の統計値を iAnywhere に送信するかどうかを確認するプロンプトが自動的に表示されます。テクニカルサポートに問題を報告すると、この情報を問題の診断に使用できます。「SQL Anywhere のエラーレポート」『SQL Anywhere サーバー データベース管理』を参照してください。

  サポートユーティリティ (dbsupport) を使用して、好きなときに機能の統計値を送信することもできます。機能の統計値情報は、iAnywhere が製品の使われ方を理解し、ソフトウェアを向上させるために役立ちます。「サポートユーティリティ (dbsupport)」『SQL Anywhere サーバー データベース管理』を参照してください。

  SADIAGDIR 環境変数は、クラッシュレポートと機能の統計を格納するディレクトリのロケーションを指定します。「SADIAGDIR 環境変数」『SQL Anywhere サーバー データベース管理』を参照してください。

- **エラーレポート**　Sybase Central、Interactive SQL、SQL Anywhere コンソールユーティリティで内部エラーが発生すると、エラーのログがハードドライブに書き込まれ、エラーレポートを iAnywhere に送信するかどうかを選択するウィンドウが表示されます。

  また、パーソナルサーバー、ネットワークサーバー、Mobile Link サーバー、dbmlsync、Mobile Link Listener、QAnywhere Agent、SQL Remote、または Replication Agent のいずれかで致命的なエラーが発生すると、エラーのログがハードドライブに書き込まれ、エラーレポートを iAnywhere に送信するかどうかを選択するウィンドウが表示されます。これらのエラーレポートが自動的に送信されるようにサポートユーティリティ (dbsupport) を設定することもできます。「サポートユーティリティ (dbsupport)」『SQL Anywhere サーバー データベース管理』を参照してください。

  エラーレポートを送信しないように選択すると、ファイルはハードディスクの診断ディレクトリに残ります。SADIAGDIR 環境変数は、クラッシュレポートと機能の統計を格納するディレクトリのロケーションを指定します。「SADIAGDIR 環境変数」『SQL Anywhere サーバー データベース管理』を参照してください。

- **管理ツールで使用できる機能の制御**　管理ツールでユーザーが使用可能な機能を制御できるようになりました。これを行うには、ツールの *.jar* ファイルと同じディレクトリにある *OEM.ini* ファイルを使用します。

- **Windows Vista での SQL Anywhere 10 の使用**　Windows Vista では、次のような既知の問題があります。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

○サーバー実行プログラムがあるディレクトリの管理者権限または書き込みパーミッションがない場合は、サーバーライセンス取得ユーティリティ (*dblic.exe*) を使用したライセンス情報の更新は失敗します。

○SQL Anywhere をサービスまたは権限のないアカウントとして実行した場合、共有メモリの使用には既知の問題があります。この問題は、調査中です。TCP/IP を代わりに使用できます。

動作の変更

次に、バージョン 10.0.0 で導入された製品全体の変更点を示します。

- **サンプルディレクトリのロケーションの変更**　SQL Anywhere 10 で用意されているサンプルは、SQL Anywhere 10 のインストール環境にはインストールされなくなりました。この変更に伴い、サンプルデータベースは次のロケーションにインストールされるようになりました。

サンプルデータベース	ロケーション
SQL Anywhere サンプルデータベース	*%SQLANYSAMP10%¥demo.db*
Mobile Link CustDB サンプル統合データベースアプリケーション	*%SQLANYSAMP10%¥MobiLink¥CustDB¥*
Ultra Light CustDB サンプルデータベース	*%SQLANYSAMP10%¥UltraLite¥CustDB¥*

- **サポート対象外のプラットフォーム**　次のプラットフォームは、サポートされなくなりました。

  ○ Windows 95
  ○ Windows 98
  ○ Windows Me
  ○ Windows NT
  ○ Compaq Tru64

  サポートされているプラットフォームのその他の変更点については、「サポートされるプラットフォーム」『SQL Anywhere 12 紹介』を参照してください。

- **サンプルデータベースの改訂と名前の変更**　SQL Anywhere サンプルデータベースの名前が *demo.db* になりました。画面読み上げソフトウェアを使用するユーザーの便宜を図り、テーブル、カラム、ビュー、インデックスなどのオブジェクトは、単語が省略されていない名前になりました。「SQL Anywhere サンプルデータベース」『SQL Anywhere 12 紹介』を参照してください。

- **その他のファイル名の変更**　製品別に示したファイル名の変更のほかに、次のファイル名が変更されました。

以前の名前	新しい名前
asa.cvf	*sqlany.cvf*
asaldap.ini	*saldap.ini*
asasrv.ini	*sasrv.ini*
asa_config.sh	*sa_config.sh*
$SQLANY9/SYBSasa9/lib	*sqlanywhere10/lib32* または *sqlanywhere10/lib64*
$SQLANY9/SYBSasa9/bin	*sqlanywhere10/bin32* または *sqlanywhere10/bin64*

新しい製品名を反映するために、一部のレジストリキーが変更されました。次のようになりました。

HKLM¥SYSTEM¥CurrentControlSet¥Services¥Eventlog¥Application¥SQLANY
HKLM¥SYSTEM¥CurrentControlSet¥Services¥Eventlog¥Application¥SQLANY 10.0
HKLM¥SYSTEM¥CurrentControlSet¥Services¥Eventlog¥Application¥SQLANY 10.0 Admin
HKLM¥SOFTWARE¥Sybase¥Sybase Central¥5.0 (registry entries)
HKLM¥SOFTWARE¥Sybase¥SQL Anywhere¥10.0 (registry entries)

製品別に示したファイルの変更点のほかに、*ulbase.lib* ファイルが追加されました。

次のサービスグループ名が変更されました。

説明	以前の名前	新しい名前
ネットワークサーバー	ASANYServer	SQLANYServer
パーソナルサーバー	ASANYEngine	SQLANYEngine
Mobile Link 同期クライアント	ASANYMLSync	SQLANYMLSync
Replication Agent	ASANYLTM	SQLANYLTM

Mobile Link、QAnywhere、リモートデータアクセスで使用される ODBC の変更

- **Sybase Adaptive Server Enterprise ドライバー** SQL Anywhere には、Adaptive Server Enterprise 用の iAnywhere Solutions ODBC ドライバーが含まれなくなりました。代わりに、Adaptive Server Enterprise ネイティブドライバーが Mobile Link で動作するようにテストされています。iAnywhere Solutions 9 - Adaptive Server Enterprise Wire Protocol ドライバーはサポートされなくなりました。

  http://www.ianywhere.com/developer/technotes/odbc_mobilink.html を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

- **IBM UDB DB2 ドライバー**　SQL Anywhere には、DB2 用の iAnywhere Solutions ODBC ドライバーが含まれなくなりました。代わりに、IBM DB2 8.2 CLI ドライバーが Mobile Link で動作するようにテストされています。このネイティブ DB2 ドライバーでは、DB2 バージョン 8.1 と 8.2 がサポートされます。IBM DB2 7.2 ODBC ドライバーと iAnywhere Solutions 9 - DB2 Wire Protocol ドライバーはサポートされなくなりました。

  http://www.ianywhere.com/developer/technotes/odbc_mobilink.html を参照してください。

- **Oracle ドライバー**　iAnywhere Solutions 10 - Oracle Wire Protocol ドライバーは、別途ダウンロードできます。

  http://www.ianywhere.com/developer/technotes/odbc_mobilink.html を参照してください。

SQL Anywhere 12 へのアップグレード

アンロード／再ロード、データベースのアップグレード、クライアントライブラリの更新を必要とする機能

SQL Anywhere データベースサーバーをアップグレードすると、多くの機能が使用できるようになります。最新バージョンのデータベースサーバーで古いデータベースを実行しても多くの機能を使用できますが、一部の機能にアクセスするには、データベースのアンロードと再ロード、データベースファイルでのアップグレードの実行、クライアントライブラリの更新を行う必要があります。

10 より前のバージョンでは、SQL Anywhere データベースサーバーは、Adaptive Server Anywhere と呼ばれていました。

注意
以下のカテゴリに示されない機能には、SQL Anywhere 12.0.1 データベースサーバーのみが必要であり、データベースのアップグレードやクライアントライブラリの更新は必要ありません。SQL Anywhere 12 の新機能の完全なリストについては、「バージョン 12.0.0 の新機能」43 ページと「バージョン 12.0.1 の新機能」1 ページを参照してください。

データベースのアンロード／再ロードを必要とする機能

- 一般的なパフォーマンスの向上。「パフォーマンスの強化」70 ページを参照してください。
- 「Ultra Light」: Ultra Light バージョン 12 では、以前のバージョンの Ultra Light で作成したデータベースに接続できません。したがって、アンロード／再ロードが必要です。ただし、Ultra Light バージョン 11 は、Mobile Link バージョン 12 と同期できます。

アップグレードのみを必要とする機能 (dbupgrad ユーティリティまたは UPGRADE DATABASE 文)

データベースのアップグレードのみを必要とする機能は、データベースをアンロード／再ロードすれば動作します。

- 監視サーバーとしてのコピーノードの使用。「監視サーバーとしてコピーノードを使用する」『SQL Anywhere サーバー データベース管理』を参照してください。
- インデックスのパフォーマンスの向上。SQL Anywhere 12.0.0 のインデックスのパフォーマンス向上 70 ページを参照してください。
- 空間データのサポート機能。空間データのサポート 45 ページを参照してください。
- TIMESTAMP WITH TIME ZONE。新しい TIMESTAMP WITH TIME ZONE データ型 65 ページを参照してください。

- システムプロシージャー。次の項を参照してください。
  - 「sp_forward_to_remote_server システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
  - 「st_geometry_load_shapefile システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
  - 「sa_user_defined_counter_add システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
  - 「sa_user_defined_counter_set システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』

クライアントライブラリの更新を必要とする機能

- 接続プーリング。「接続プーリング」『SQL Anywhere サーバー データベース管理』を参照してください。
- 進行状況メッセージ。新しい progress_messages オプション 56 ページを参照してください。
- 共有メモリ接続でのアイドルタイムアウト。Idle 接続パラメーターの動作変更 75 ページを参照してください。
- TIMESTAMP WITH TIME ZONE。新しい TIMESTAMP WITH TIME ZONE データ型 65 ページを参照してください。
- クライアント側での空間データのサポート。空間データのサポート 45 ページを参照してください。
- SQL Anywhere JDBC ドライバー。新しい SQL Anywhere TYPE-2 JDBC ドライバー 66 ページを参照してください。

SQL Anywhere サーバーのアップグレード

このバージョンのソフトウェアで既存のアプリケーションを使用する前に、動作の変更のリストを確認して、アプリケーションに影響がないかどうかを確認してください。SQL Anywhere 12 変更点とアップグレード 1 ページを参照してください。

バージョン 10 以降のデータベースのアップグレード

バージョン 10 以降からアップグレードするには、データベースをアップグレードするか再構築します。バージョン 12 のソフトウェアでバージョン 10 以降のデータベースを使用できるので、アップグレードや再構築を実行するかどうかは任意です。ただし、バージョン 12 の新機能をすべて利用したい場合は、データベースを再構築する必要があります。「バージョン 10 以降のデータベースのアップグレード手順」379 ページと「バージョン 10 以降のデータベースの再構築手順」371 ページを参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

注意
データベースサーバーをアップグレードした後、またはアップグレード後のデータベースサーバーで使用できるようにデータベースを再構築またはアップグレードした後は、データベース内のマテリアライズドビューを再表示することをおすすめします。「手動マテリアライズドビューのリフレッシュ」『SQL Anywhere サーバー SQL の使用法』を参照してください。

バージョン 9 以前のデータベースのアップグレード

バージョン 9 以前からバージョン 12 にアップグレードするには、データベースを再構築します。再構築するには、古いデータベースをアンロードしてから、バージョン 12 の新しいデータベースに再ロードします。バージョン 9 以前のデータベースを開始しようとすると、データベースの開始時にエラーが発生します。既存のデータベースを再構築するには、いくつかの方法があります。

- -an (データベースの新規作成) オプションまたは -ar (古いデータベースの置き換え) オプションを指定して、バージョン 12 のアンロードユーティリティ (dbunload) を実行します。アップロードユーティリティを使用したバージョン 9 以前のデータベースの再構築 (コマンドライン)378 ページを参照してください。

  注意
  アンロード ユーティリティ (dbunload) には、SQL Anywhere の全バージョンで同じファイル名が使用されています。正しいバージョンを使用していることを確認してください。使用しているアンロードユーティリティのバージョンを確認するには、dbunload -? コマンドを実行します。「ユーティリティのバージョンとアップグレード手順」369 ページを参照してください。

- バージョン 12 のアンロードユーティリティを使用してデータベースをアンロードしてから、バージョン 12 のデータベースサーバーにある *reload.sql* ファイルを使用してデータベースを再ロードします。

  スキーマを変更する必要がある場合は、この方法を使用してアップグレードすることをおすすめします。スキーマの変更後は、新しいデータベースを作成して、再ロードスクリプトを適用できます。

- Sybase Central の**データベースアンロードウィザード**を使用します。新しいデータベースを作成するか、既存のデータベースを新しいデータベースに置き換えるか、データベースをファイルにアンロードします。バージョン 9 以前のデータベースの再構築 (Sybase Central) 377 ページを参照してください。

- 旧バージョンの dbunload を使用してデータベースをアンロードし、*reload.sql* ファイルとバージョン 12 のデータベースサーバーを使用してデータベースを再ロードします。廃止予定またはサポート対象外のデータベースオプション設定、オブジェクト、または SQL 構文が *reload.sql* ファイルにアンロードされる可能性があるため、この方法は他の方法が失敗する場合にのみ使用してください。この方法を使用して再ロード中に問題が発生した場合は、ファイルを手動で編集する必要があります。バージョン 12 では、内部再ロード機能により、このような問題の多くが解決されています。

注意
Mac OS X 用の SQL Anywhere 9.0.2 は PPC でサポートされ、SQL Anywhere 10 以降は Intel でサポートされていました。Mac OS X に 9.0.2 以前のバージョンのデータベースがある場合は、次の 2 つの方法でデータベースをアンロードできます。

●バージョン 9.0.2 のソフトウェアを使用してデータベースをアンロードします。

●SQL Anywhere 12 がインストールされている別のプラットフォームにデータベースをコピーしてから、バージョン 12 のソフトウェアを使用してデータベースをアンロードします。

データベースをアンロードした後は、Mac OS X でバージョン 12 のソフトウェアを使用してデータベースを再ロードできます。

アンロードと再ロードのときに (大文字と小文字を「区別する」から「区別しない」に変更するなど) データベースの特性を変更する場合、手順はもう少し複雑になります。「データベースの再構築」『SQL Anywhere サーバー SQL の使用法』を参照してください。

既存のソフトウェアとの互換性

SQL Anywhere 12 のデータベースサーバーは、バージョン 6.0.0 以降のソフトウェアを使用するクライアントアプリケーションからの接続をサポートします。バージョン 5 以前のクライアントは、バージョン 12 のデータベースサーバーに接続できません。

既存の Sybase Central ソフトウェアとの互換性

フルサポート　Sybase Central の SQL Anywhere プラグインは、バージョン 10 以降のサーバーで動作するバージョン 10 以降のソフトウェアで作成されたデータベースを完全にサポートしています。たとえば、バージョン 10、11、または 12 サーバーで動作するバージョン 10 データベース、バージョン 11 または 12 データベースサーバーで動作するバージョン 11 データベースおよびバージョン 12 データベースサーバーで動作するバージョン 12 データベースなどです。

アンロード／再ロード専用サポート　バージョン 6、7、8、9 データベースサーバーで動作するバージョン 5、6、7、8、9 ソフトウェアで作成されたデータベースの場合、SQL Anywhere プラグイン 「temporarily」 からデータベースに接続して、次のいずれかのタスクを実行できます。

○データベースを再ロードファイルにアンロードします。

○データベースをアンロードして、新しいバージョン 12 データベースに再ロードします。

○データベースをアンロードして、既存のバージョン 12 データベースに再ロードします。

アンロードされるデータベースのファイルはローカルのコンピューターに置いておく必要があります。

サポートなし　バージョン 5 以前のサーバーで動作するバージョン 4 以前のソフトウェアによって作成されたデータベースの SQL Anywhere プラグインはサポートされません。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ユーティリティのバージョンとアップグレード手順

SQL Anywhere の複数のバージョンが同じ Windows コンピューターにインストールされている場合は、ユーティリティを使用するとき、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ディレクトリ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ている場合があります。

たとえば、パス内で SQL Anywhere バージョン 8 の実行プログラムのディレクトリが SQL Anywhere バージョン 12 の実行プログラムのディレクトリよりも前にある場合に dbinit コマンドを使用すると、バージョン 8 のユーティリティを使用することになり、その結果、バージョン 8 のデータベースが作成されてしまいます。

バージョン 12 のユーティリティが確実に使用されるようにするには、いくつかの方法があります。

- SQL Anywhere 12 の実行プログラムディレクトリが、旧バージョンの実行プログラムディレクトリより前になるようにシステムパスを変更します。
- コマンドを実行する前に、現在のディレクトリを SQL Anywhere 12 の実行プログラムディレクトリに変更します。
- 実行するユーティリティの正確な場所を示すユーティリティ名への完全修飾パスを指定します。
- 正しいバージョンのユーティリティが使用されるように、環境を変更するスクリプトを作成します。
- 旧バージョンのソフトウェアをアンインストールします。

アップグレードの注意事項

SQL Anywhere をアップグレードする前に、次の作業を行ってください。

- **動作の変更のチェック**　記述されている動作変更が既存のアプリケーションに影響を与えないことを確認してください。影響する場合は、既存のアプリケーションを更新してください。「バージョン 12.0.1 の新機能」1 ページを参照してください。
- **アプリケーションのテスト**　事前に SQL Anywhere 12 環境でアプリケーションのテストを徹底的に行ってから、実際の運用に使用するアプリケーションをアップグレードしてください。
- **正しいバージョンのユーティリティの使用**　新しいデータベースには、正しいバージョンのデータベースユーティリティを使用してください。「ユーティリティのバージョンとアップグレード手順」369 ページを参照してください。
- **データベースの検証とバックアップ**　アップグレードを開始する前に、データベースを検証し、ソフトウェアとデータベースをバックアップします。後でリカバリできるように、アップグレードの終了時にデータベースをバックアップします。

- **アップグレードの前に同期**　Ultra Light データベースや、Mobile Link インストール環境の SQL Anywhere リモートデータベースなど、同期に関連するデータベースの場合は、同期を正しく行ってからアップグレードしてください。
- **アップグレード手順のテスト**　アップグレード手順は、十分にテストしてから運用システムで実行してください。

SQL Anywhere はさまざまな設定で使用されるので、すべての場合に対応できるアップグレード手順があるわけではありません。

データベースのアップグレード

♦ データベースの再構築 (コマンドライン)

旧バージョンのユーザーは、次の手順に従って、データベースをバージョン 12 に再構築できます。

1. データベースをバックアップします。次に例を示します。

   ```
   dbbackup -c "DBF=mydb.db;UID=DBA;PWD=sql" old-db-backup-dir
   ```

   詳細については、「データベースのバックアップ」『SQL Anywhere サーバー データベース管理』を参照してください。

2. 断片化されているドライブでは、データベースのパフォーマンスが低下するので、可能な場合は、新しいデータベースを保存するドライブの断片化を解除してください。

3. バージョン 12 の dbunload ユーティリティは、旧バージョンのデータベースサーバーで実行されているデータベースに対して使用できないので、すべての SQL Anywhere データベースサーバーを停止します。次に例を示します。

   ```
   dbstop -c "DBF=mydb.db;UID=DBA;PWD=sql"
   ```

4. 既存のデータベースをアンロードして、新しいバージョン 12 のデータベースに再ロード (再構築) します。次に例を示します。

   ```
   dbunload -c "DBF=mydb.db;UID=DBA;PWD=sql" -an mydb12.db
   ```

5. 新しいデータベースは、バックアップしてから使用します。次に例を示します。

   ```
   dbbackup -c "DBF=mydb12.db;UID=DBA;PWD=sql" new-db-backup-dir
   ```

6. 新しいデータベースは、検証してから使用します。次に例を示します。

   ```
   dbvalid -c "DBF=mydb12.db;UID=DBA;PWD=sql"
   ```

参照

- アップロードユーティリティを使用したバージョン 9 以前のデータベースの再構築 (コマンドライン)378 ページ
- バージョン 9 以前のデータベースの再構築 (Sybase Central)377 ページ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 10 以降のデータベースの再構築手順

データベースを再構築するには、データベースをアンロードしてから再ロードすることでファイルフォーマットをアップグレードします。ファイルフォーマットをアップグレードすると、ディスク上でデータの保管とアクセスに使用されるフォーマットが変更され、ソフトウェアの最新バージョンの新機能とパフォーマンス向上をすべて利用できます。

警告
大規模なデータベースのアンロードや再ロードには時間がかかり、大容量のディスク領域が必要になることがあります。この処理は、アンロードされたデータと新しいデータベースファイルを保持するため、データベースの約 2 倍のディスク領域を要する場合があります。

SQL Anywhere のインデックスが変更された結果、データベースをアンロードしてから再ロードして再構築すると、再構築されたデータベースが元のデータベースよりも小さくなる場合があります。このデータベースのサイズの縮小は、問題や、データが失われたことを示すものではありません。

注意
データベースは、再構築の前にバックアップすることをおすすめします。

AUTOINCREMENT カラムのあるテーブルの再ロード

dbunload の -l オプションを指定すると、再構築されたデータベースの AUTOINCREMENT カラムに次に使用可能な値を保持できます。このオプションを指定すると、AUTOINCREMENT 値が含まれるテーブルごとに、生成される *reload.sql* スクリプトに sa_reset_identity システムプロシージャーへの呼び出しが追加され、SYSTABCOL.max_identity の現在の値が維持されます。

データベースの再構築

♦ データベースの再構築 (Sybase Central)

1. ソフトウェアをアップグレードする場合の一般的な対応策を行います。「アップグレードの注意事項」369 ページを参照してください。
2. データベースをバックアップします。次に例を示します。

   ```
   dbbackup -c "DBF=mydb.db;UID=DBA;PWD=sql" old-db-backup-dir
   ```

   詳細については、「データベースのバックアップ」『SQL Anywhere サーバー データベース管理』を参照してください。
3. **[スタート] » [プログラム] » [SQL Anywhere 12] » [管理ツール] » [Sybase Central]** をクリックします。
4. アップグレードするデータベースを実行するためにバージョン 12 のデータベースサーバーを起動してから、Sybase Central からデータベースに接続します。
5. **[ツール] » [SQL Anywhere 12] » [データベースのアンロード]** をクリックします。
6. **データベースアンロードウィザード**の最初のページの説明を読んで、**[次へ]** をクリックします。

7. **[現在のバージョンのサーバーで実行中のデータベースをアンロードする]** をクリックし、リストからデータベースを選択します。**[次へ]** をクリックします。

8. [新しいデータベースへのアンロードと再ロード] を選択します。**[次へ]** をクリックします。

9. データベースの新しいファイル名を指定します。

10. 新しいデータベースのページサイズを指定することもできますが、指定するページサイズは、データベースサーバーのページサイズ以下である必要があります。デフォルトのページサイズは 4096 バイトです。必要に応じてデータベースファイルを暗号化できます。強力な暗号化を選択した場合は、データベースを起動するたびに暗号化キーが必要です。**[次へ]** をクリックします。

    データベースファイルの暗号化の詳細については、「データベースの暗号化と復号化」『SQL Anywhere サーバー データベース管理』を参照してください。

11. **[構造とデータをアンロード]** をクリックします。必要に応じて、他のオプションも選択します。**[次へ]** をクリックします。

12. **[すべてのデータベースオブジェクトをアンロード]** をクリックします。**[次へ]** をクリックします。

13. アンロード／再ロードが完了したときに新しいデータベースに接続するかどうかを指定します。

14. **[完了]** をクリックすると、処理が開始します。新しいデータベースを調べて、再構築が正常に完了したことを確認します。

    データベースアンロードウィザードの使用の詳細については、「データベースアンロードウィザードを使用したデータのエクスポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

♦ データベースの再構築 (コマンドライン)

バージョン 10 以降のデータベースで dbunload を使用する場合は、使用する dbunload のバージョンが、データベースへのアクセスに使用するデータベースサーバーのバージョンと一致する必要があります。dbunload のバージョンがデータベースサーバーのバージョンより古いまたは新しい場合は、エラーが返されます。

dbunload ユーティリティを使用して、Mobile Link インストール環境のリモートデータベース、または SQL Remote レプリケーションに使用するデータベースを再構築する場合は、-ar オプションまたは -an オプションを必ず使用してください。これらのオプションによって、新しいデータベースのトランザクションログのオフセットが、古いデータベースと同じになるように設定されます。

1. ソフトウェアをアップグレードする場合の一般的な対応策を行います。「アップグレードの注意事項」369 ページを参照してください。

2. アップグレードするデータベースに排他的にアクセスできること、またシステムパスで、バージョン 12 のユーティリティのパスがその他のユーティリティのパスより前に指定されてい

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ることを確認します。「ユーティリティのバージョンとアップグレード手順」369 ページを参照してください。

3. アンロードユーティリティ (dbunload) を実行し、-ar オプションを使用して新しいデータベースを作成します。

   ```
   dbunload -c "connection-string" -an new-db-file
   ```

   connection-string で指定するデータベースユーザーは、再構築しているデータベースの DBA 権限を持っている必要があります。

   このコマンドにより、新しいデータベースが作成されます。既存のデータベースをアップグレード後のデータベースに置き換えるには、-an オプションの代わりに -ar オプションを指定します。-ar オプションを使用するには、パーソナルデータベースサーバーに接続するか、アンロードユーティリティ (dbunload) と同じコンピューター上にあるネットワークデータベースサーバーに接続します。

   アンロードユーティリティ (dbunload) のその他のオプションについては、「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

4. 再ロードしたデータベースを使用する前に、データベースを停止し、トランザクションログを圧縮します。

   アンロードと再ロードのときに (大文字と小文字を「区別する」から「区別しない」に変更するなど) データベースの特性を変更する場合、手順はもう少し複雑になります。「データベースの再構築」『SQL Anywhere サーバー SQL の使用法』を参照してください。

バージョン 9 以前のデータベースのバージョン 12 用への再構築

再構築の前に、データベースをバックアップすることをおすすめします。

> **注意**
> データベースのページサイズには、2048、4096、8192、16384、32768 バイトのいずれかを指定できます。デフォルトは元のデータベースのページサイズです。

Windows Mobile データベースのアップグレードについては、「Windows Mobile でのデータベースの再構築」『SQL Anywhere サーバー データベース管理』を参照してください。

注意
Mac OS X 用の SQL Anywhere 9.0.2 は PPC でサポートされ、SQL Anywhere 10.0.0 以降は Intel でサポートされていました。Mac OS X に 9.0.2 以前のバージョンのデータベースがある場合は、次の 2 つの方法でデータベースをアンロードできます。

●バージョン 9.0.2 のソフトウェアを使用してデータベースをアンロードします。

●SQL Anywhere 12 がインストールされている別のプラットフォームにデータベースをコピーしてから、バージョン 12 のソフトウェアを使用してデータベースをアンロードします。

データベースをアンロードした後は、Mac OS X でバージョン 12 のソフトウェアを使用してデータベースを再ロードできます。

警告
大規模なデータベースのアンロードや再ロードには時間がかかり、大容量のディスク領域が必要になることがあります。この処理では、アンロードされたデータと新しいデータベースファイルを保持するため、データベースの約 2 倍のディスク領域にアクセスする必要があります。

アップグレードの制限

バージョン 12 のツールを使用してバージョン 9 以前のデータベースを再構築するときには、次の制限事項があります。

●旧バージョンの全データベースサーバーのデータベースを切断して、コンピューターで実行している旧バージョンのデータベースサーバーを停止する必要があります。また、コンピューターで実行しているバージョン 12 のデータベースサーバーも停止する必要があります。dbunload によってこれらの状態が検出され、処理を続行できない場合は、エラーが発生し、再構築に失敗します。

●再構築前のデータベースの dbunload 接続文字列に、ENG、START、または LINKS 接続パラメーターを含めないでください (-c オプションで指定)。これらの接続パラメーターを指定すると、無視され、警告が表示されます。Sybase Central の **[接続]** ウィンドウの **[サーバー名]** フィールドまたは **[開始行]** フィールドには、値を入力しないでください。

●既存のデータベースへダイレクトにファイルシステムへのアクセスを使用して、コンピューターで dbunload を実行します (dbunload が、共有メモリを使用してデータベースに接続できる状態である必要があります)。

●再構築を実行するコンピューターで、dbunload_support_engine という名前のデータベースサーバーを実行することはできません。

●バージョン 9 以前のデータベースは、データベースファイルのリカバリが必要となった場合に、アンロードできません。リカバリが必要なデータベースファイルでアンロードユーティリティ (dbunload) を実行すると、データベースを起動できないことを示すメッセージが返されます。dbunload を再試行する前に、バージョン 9 のデータベースサーバーを使用して、データベースを起動してから停止します。バージョン 9 以降のデータベースをアンロードするには、データベースを読み込み専用モードで開始できるようにしておく必要があります。「-r dbeng12/dbsrv12 データベースオプション」『SQL Anywhere サーバー データベース管理』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

特別な注意事項

- **パスワードの大文字と小文字の区別**　新しく作成された SQL Anywhere 12 データベースでは、データベースでの設定にかかわらず、すべてのパスワードは大文字と小文字が区別されます。新しいデータベースのデフォルトの DBA パスワードは、**sql** です。

  既存のデータベースを再構築する場合、SQL Anywhere でのパスワードの大文字と小文字の区別は、次のように決まります。

  ○パスワードを最初に入力したのが大文字と小文字を区別しないデータベースだった場合、そのパスワードの大文字と小文字は区別されません。

  ○パスワードを最初に入力したのが大文字と小文字を区別するデータベースだった場合、大文字のパスワードと、大文字と小文字が混在したパスワードでは、大文字と小文字が区別されます。ただし、パスワードをすべて小文字で入力した場合、パスワードの大文字と小文字は区別されません。

  ○既存のパスワードと新しいパスワードの両方に加えられた変更は、大文字と小文字が区別されます。

- **ページサイズ**　SQL Anywhere 12 データベースのデフォルトのデータベースページサイズは 4096 バイトです。バージョン 12 でサポートされるページサイズは、2048、4096、8192、16384、32768 バイトです。既存のデータベースで、サポートされていないページサイズが使用されている場合は、新しいデータベースのページサイズはデフォルトで 4096 バイトに設定されます。dbinit の -p オプションまたは dbunload の -ap オプションを使用して、別のページサイズを指定できます。「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』と「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

- **照合**　バージョン 9 以前の SQL Anywhere では、CHAR データ型で使用される 1 つの照合しかサポートしていませんでした。この照合は、SQL Anywhere 照合アルゴリズム (SACA) を使用していました。バージョン 10 以降の SQL Anywhere では、照合アルゴリズムとして、SACA と UCA (Unicode 照合アルゴリズム) の 2 つをサポートしています。再構築されたデータベース用に新しい照合や別の照合を指定していない場合は、既存のデータベースの SACA 照合がアンロードされ、再構築されたデータベースで再利用されます。

  カスタム照合を使用してデータベースを再構築する場合に 1 ステップで再構築すると、その照合は保持されます (内部アンロード)。データベースをアンロードしてから、作成したデータベースにスキーマとデータをロードする場合は、提供されるいずれかの照合を使用する必要があります。「サポートされている照合と代替照合」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースのファイルサイズ**　SQL Anywhere のインデックスが変更された結果、データベースをアンロードしてから再ロードして再構築すると、再構築されたデータベースが元のデータベースよりも小さくなる場合があります。このデータベースのサイズの縮小は、問題や、データが失われたことを示すものではありません。

既知の問題

dbunload または**データベースアンロードウィザード**を実行したときに再構築処理に失敗した場合は、次の手順に従って失敗の原因を診断できます。

♦ 再構築の失敗の診断

1. 既存のデータベースで dbunload -n を実行します。-n が指定されているため、データはアンロードされません。

   ```
   dbunload -c "connection-string" -n directory-name
   ```

2. 新しい、バージョン 12 の空のデータベースを作成します。

   ```
   dbinit test.db
   ```

3. *reload.sql* ファイルを空のデータベースに適用します。

   ```
   dbisql -c "DBF=test.db;UID=DBA;pwd=sql" reload.sql
   ```

4. *reload.sql* ファイルを新しいデータベースに適用したときに受信したメッセージに基づいて、元のデータベースまたは *reload.sql* ファイルに変更を加えます。

次の表に、再構築の失敗につながる既知の問題とその解決策を示します。

既知の問題	解決策
テーブル名のプレフィクスに所有者名が付いている場合は、プロシージャーまたはトリガー内の DECLARE LOCAL TEMPORARY TABLE 文により、構文エラーが発生します。	所有者名を削除します。
CREATE TRIGGER 文で、トリガーが定義されたテーブルの所有者名が指定されておらず、ユーザーが *reload.sql* ファイルを実行して参照するときに、テーブルが所有者名で修飾されている必要がある場合、CREATE TRIGGER 文は失敗し、テーブルが見つからなかったことを示すエラーが返されます。	テーブル名のプレフィクスに所有者名を付けます。

オブジェクト名 (テーブル、カラム、変数、パラメーター名など) が SQL Anywhere の以後のバージョンで導入された予約語 (NCHAR など) に対応している場合は、再ロードに失敗します。次に例を示します。
`CREATE PROCEDURE p( )`
`BEGIN`
`DECLARE NCHAR INT;`
`SET NCHAR = 1;`
`END;`
	予約語へのすべての参照を、別の名前を使用するように変更します。変数名の場合は、名前の競合を回避する @ を名前のプレフィクスとして使用するのが、一般的な規則です。
	予約語の完全なリストについては、「予約語」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

既知の問題	解決策
データベースをバージョン 9 以前の dbunload でアンロードする場合、*reload.sql* ファイルに ml_add_property システムプロシージャーの呼び出しが含まれることがありますが、新しいバージョン 12 データベースにはこのプロシージャーは存在しません。	バージョン 12 の dbunload ユーティリティを使用してデータベースをアンロードしてください。 適切なバージョンのデータベースユーティリティを使用していることを確認する方法については、「ユーティリティのバージョンとアップグレード手順」 369 ページを参照してください。
バージョン 9 以前の dbunload を使用してデータベースをアンロードする場合、Transact-SQL 外部ジョイン (*= または =* で指定) を使用するビューを再ロードすると、それらのビューが適切に作成されないことがあります。	次のセクション行を再ロードスクリプトに追加します。 SET TEMPORARY OPTION tsql_outer_joins='on'; Transact-SQL 外部ジョインを使用しているすべてのビューを、後で書き直してください。
CREATE PROCEDURE 文と ALTER PROCEDURE 文で [NOT] DETERMINISTIC 句がサポートされていません。この句を指定すると、再ロードに失敗し構文エラーが返されます。	[NOT] DETERMINISTIC 句が指定されたユーザー定義プロシージャーを含むデータベースをアップグレードする場合、句を削除してからデータベースのアンロードと再ロードを行う必要があります。

バージョン 9 以前のデータベースの再構築

♦ バージョン 9 以前のデータベースの再構築 (Sybase Central)

データベースアンロードウィザードを使用して、既存のデータベースを再構築できます。このウィザードを使用すると、データベースを再ロードファイルとデータファイルにアンロードするか、新しいデータベースにアンロードおよび再ロードするか、既存のデータベースにアンロードおよび再ロードするかのいずれかの操作を実行できます。再構築の前に、データベースをバックアップすることを強くおすすめします。

注意

- データベースファイルは、SQL Anywhere 12 がインストールされているコンピューターと同じコンピューターに保存されている必要があります。
- データベースからはテーブルのサブセットはアンロードできません。dbunload ユーティリティを使用してアンロードしてください。
- **データベースアンロードウィザード**で、データベースファイルがすでに実行されていると認識された場合は、アンロードを続行する前にデータベースが停止します。

1. ソフトウェアをアップグレードする場合の一般的な対応策を行います。「アップグレードの注意事項」 369 ページを参照してください。

2. 断片化されているドライブでは、データベースのパフォーマンスが低下するので、可能な場合は、新しいデータベースを保存するドライブの断片化を解除してください。

3. アンロードと再ロードを行うデータベースに対して、排他アクセスを持っていることを確認してください。他のユーザーは接続できません。

4. **[スタート] » [プログラム] » [SQL Anywhere 12] » [管理ツール] » [Sybase Central]** をクリックします。

5. **[ツール] » [SQL Anywhere 12] » [データベースのアンロード]** をクリックします。

6. **データベースアンロードウィザード**の概要ページを読んで、**[次へ]** をクリックします。

7. **[旧バージョンのサーバーで実行中のデータベース、または実行中ではないデータベースをアンロードする]** をクリックします。データベースの接続情報を指定します。**[次へ]** をクリックします。

8. **[新しいデータベースへのアンロードと再ロード]** をクリックします。**[次へ]** をクリックします。

9. データベースの新しいファイル名を指定します。**[次へ]** をクリックします。

   新しいデータベースのページサイズを指定できます。バージョン 12 では、デフォルトのページサイズ (推奨値) は 4096 バイトです。

   必要に応じてデータベースファイルを暗号化できます。強力な暗号化を選択した場合は、データベースを起動するたびに暗号化キーが必要です。「データベースの暗号化と復号化」『SQL Anywhere サーバー データベース管理』を参照してください。

10. [構造とデータをアンロード] を選択します。**[次へ]** をクリックします。

11. 再構築が完了したときに新しいデータベースに接続するかどうかを指定します。

12. **[完了]** をクリックします。新しいデータベースを調べて、再構築が正常に完了したことを確認します。

♦ アップロードユーティリティを使用したバージョン 9 以前のデータベースの再構築 (コマンドライン)

アンロードユーティリティ (dbunload) の -an または -ar オプションを使用して、既存のデータベースを再構築できます。-an オプションを使用すると、元のデータベースをそのまま残して新規データベースが作成されるので、このオプションを使用することをおすすめします。-ar オプションを使用すると、既存のデータベースが新しいバージョン 12 のデータベースに置き換わります。

1. ソフトウェアをアップグレードする場合の一般的な対応策を行います。「アップグレードの注意事項」369 ページを参照してください。

2. システムパスで、他のユーティリティより前にバージョン 12 のユーティリティが置かれていることを確認してください。「ユーティリティのバージョンとアップグレード手順」369 ページを参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

3. バージョン 12 の dbunload ユーティリティは、旧バージョンのデータベースサーバーで実行されているデータベースに対して使用できないので、すべての SQL Anywhere と Adaptive Server Anywhere のデータベースサーバーを停止します。次に例を示します。

   ```
   dbstop -c "DBF=mydb.db;UID=DBA;PWD=sql"
   ```

4. 断片化されているドライブでは、データベースのパフォーマンスが低下するので、可能な場合は、新しいデータベースを保存するドライブの断片化を解除してください。

5. データベースをバックアップします。次に例を示します。

   ```
   dbbackup -c "DBF=mydb.db;UID=DBA;PWD=sql" old-db-backup-dir
   ```

   「データベースのバックアップ」『SQL Anywhere サーバー データベース管理』を参照してください。

6. -an または -ar オプションを指定してアンロードユーティリティ (dbunload) を実行して、新しいデータベースを作成します。

   ```
   dbunload -c "connection-string" -an database-filename
   ```

   次に例を示します。

   ```
   dbunload -c "DBF=mydb.db;UID=DBA;PWD=sql" -an mydb12.db
   ```

   connection-string で指定するデータベースユーザーは、アンロードするデータベースに DBA 権限で接続する必要があります。このコマンドを使用すると、新規データベースが作成されます (-an を指定した場合)。-ar オプションを指定すると、既存のデータベースは再構築後のデータベースに置き換わります。-ar オプションを使用するには、パーソナルデータベースサーバーに接続するか、アンロードユーティリティ (dbunload) と同じコンピューター上にあるネットワークデータベースサーバーに接続します。

   アンロードユーティリティ (dbunload) のその他のオプションについては、「アンロードユーティリティ (dbunload)」『SQL Anywhere サーバー データベース管理』を参照してください。

バージョン 10 以降のデータベースのアップグレード手順

データベースをアップグレードすると、バージョン 12 の機能を有効にするために、システムテーブル、システムプロシージャー、データベースオプションが追加、変更されます。ディスク上でデータが保管、アクセスされるファイルフォーマットは変更されないので、ソフトウェアの最新バージョンの新機能とパフォーマンス向上をすべて利用できるわけではありません。

データベースのファイルフォーマットのアップグレードの詳細については、「バージョン 10 以降のデータベースの再構築手順」371 ページを参照してください。

データベースアップグレードウィザードでは、バージョン 9.0.2 以前のデータベースをバージョン 12 にアップグレードすることはできません。バージョン 9.0.2 以前の既存のデータベースをバージョン 12 にアップグレードするには、dbunload または**データベースアンロードウィザード**を使用してデータベースをアンロードし、再ロードする必要があります。「バージョン 9 以前のデータベースのアップグレード」367 ページを参照してください。

警告
必ずデータベースファイルをバックアップしてからアップグレードしてください。既存のファイルにアップグレードを適用した場合、アップグレードに失敗すると、これらのファイルは使用できなくなります。データベースのバックアップの詳細については、「バックアップとデータリカバリ」『SQL Anywhere サーバー データベース管理』を参照してください。

♦ データベースのアップグレード (Sybase Central)

1. ソフトウェアをアップグレードする場合の一般的な対応策を行います。「アップグレードの注意事項」369 ページを参照してください。
2. **[スタート]** » **[プログラム]** » **[SQL Anywhere 12]** » **[管理ツール]** » **[Sybase Central]** をクリックします。
3. **[SQL Anywhere 12]** のプラグインから、アップグレードするデータベースに接続します。データベースはバージョン 12 のデータベースサーバーで実行されている必要があります。
4. **[ツール]** » **[SQL Anywhere 12]** » **[データベースのアップグレード]** をクリックします。
5. **データベースアップグレードウィザード**の指示に従います。
6. アップグレードしたデータベースを使用する前に、データベースを停止し、トランザクションログをコピーしてアーカイブすることをウィザードで選択しなかった場合は、その操作を行います。

ヒント
データベースアップグレードウィザードは、次の方法でもアクセスできます。

- データベースを右クリックし、**[データベースのアップグレード]** をクリックします。
- データベースを選択し、**[ファイル]** » **[データベースのアップグレード]** をクリックします。

♦ データベースのアップグレード (コマンドライン)

1. ソフトウェアをアップグレードする場合の一般的な対応策を行います。「アップグレードの注意事項」369 ページを参照してください。
2. アップグレードするデータベースに排他的にアクセスできること、またシステムパスで、バージョン 12 のユーティリティがその他のユーティリティより前に指定されていることを確認します。「ユーティリティのバージョンとアップグレード手順」369 ページを参照してください。
3. データベースに対してアップグレードユーティリティ (dbupgrad) を実行します。

```
dbupgrad -c "connection-string"
```

connection-string で指定するデータベースユーザーは、アップグレードするデータベースの DBA 権限を持っている必要があります。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

詳細については、「アップグレードユーティリティ (dbupgrad)」『SQL Anywhere サーバー データベース管理』を参照してください。

4. アップグレードしたデータベースを使用する前に、データベースを停止し、トランザクションログを圧縮します。

♦ データベースのアップグレード (SQL)

1. Interactive SQL、または SQL 文を実行できる別のアプリケーションからデータベースに接続します。他のどの接続も、データベースを同時に使用できません。

2. ALTER DATABASE 文を実行します。

   たとえば、次の文はデータベースをアップグレードします。

   ```
   ALTER DATABASE UPGRADE;
   ```

   詳細については、「ALTER DATABASE 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

3. アップグレードしたデータベースを使用する前に、データベースを停止し、トランザクションログを圧縮します。

データベースミラーリングシステムでの SQL Anywhere ソフトウェアとデータベースのアップグレード

データベースミラーリングを使用しているときは、SQL Anywhere のメンテナンスリリースまたは EBF を適用するため、またデータベースファイルをアップグレードするために追加の作業が必要です。

- メンテナンスリリースの適用の詳細については、SQL Anywhere のメンテナンスリリースのデータベースミラーリングシステムへの適用381 ページを参照してください。
- EBF の適用の詳細については、データベースミラーリングシステムでの SQL Anywhere の EBF の適用382 ページを参照してください。
- データベースファイルのアップグレードまたは再構築の詳細については、「データベースミラーリングシステムでのデータベースのアップグレード」382 ページを参照してください。

♦ SQL Anywhere のメンテナンスリリースのデータベースミラーリングシステムへの適用

データベースミラーリングシステム内のすべてのデータベースサーバーで、SQL Anywhere の同じメンテナンスリリースを使用する必要があります。次の手順で SQL Anywhere のメンテナンスリリースを適用した場合、データベースを使用できないのは手順 3 と 4 の実行中だけです。

1. dbstop コマンドを実行してミラーサーバーを停止します。

2. ミラーサーバーに SQL Anywhere の新バージョンをインストールします。

3. サーバーごとに dbstop コマンドを実行して、プライマリサーバーと監視サーバーを停止します。
4. プライマリサーバーに SQL Anywhere の新バージョンをインストールします。
5. プライマリサーバーとミラーサーバーを再起動します。
6. 監視サーバーにソフトウェアの新バージョンをインストールします。
7. 監視サーバーを再起動します。

♦ データベースミラーリングシステムでの SQL Anywhere の EBF の適用

EBF をインストールするには、ミラーリングシステム内のデータベースサーバー (プライマリサーバー、ミラーサーバー、監視サーバー) ごとに次の操作を行う必要があります。

1. dbstop コマンドを実行して、データベースサーバーを停止します。
2. EBF をインストールします。
3. データベースサーバーを再起動します。

データベースミラーリングシステムでのデータベースのアップグレード

データベースミラーリングシステム内のデータベースは、2 通りの方法でアップグレードまたは再構築できます。最初の方法の方が簡単ですが、2 つ目の方法よりもデータベースのダウン時間が長くなります。

♦ データベースミラーリングシステム内でのデータベースのアップグレードまたは再構築

1. ミラーサーバーを停止します。
2. プライマリサーバーを停止します。
3. プライマリサーバー上のコピーを使用して、データベースをアップグレードまたは再構築します。「バージョン 10 以降のデータベースのアップグレード手順」379 ページまたは「バージョン 10 以降のデータベースの再構築手順」371 ページを参照してください。
4. アップグレードまたは再構築したデータベースとトランザクションログをミラーサーバーにコピーします。
5. プライマリサーバーを再起動します。
6. ミラーサーバーを再起動します。

> **注意**
> 名前を変更したトランザクションログファイルがある場合は移動してください。これらのファイルは、新しいデータベースと互換性がありません。ミラーリングを開始するには、初期のトランザクションログファイルが両方のサーバーに必要です。トランザクションログファイルは、データベースに対して dbping コマンドを実行することで作成できます。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

♦ データベースミラーリングシステム内のデータベースのアップグレードまたは再構築時のダウン時間の最小化

1. データベースをバックアップし、トランザクションログの名前を変更します。
2. データベースのバックアップコピーを別のコンピューターにアップグレードまたは再構築します。「バージョン 10 以降のデータベースのアップグレード手順」379 ページまたは「バージョン 10 以降のデータベースの再構築手順」371 ページを参照してください。
3. プライマリサーバーとミラーサーバーの両方を停止します。
4. プライマリデータベースに対するトランザクションログの最新のコピーを保存します。
5. dbtran ユーティリティを使用して、手順 4 で保存したトランザクションログを変換します。

   このトランザクションログには、手順 1 でバックアップした以降、データベースに適用されたすべての変更が含まれます。
6. ローカルデータベースサーバーを使用して、再構築したデータベースを開始します。
7. Interactive SQL から READ 文を使用して、変換されたトランザクションログを適用します。
8. 再構築したデータベースを停止します。
9. アップグレードまたは再構築したデータベースとそのトランザクションログをプライマリサーバーとミラーサーバーにコピーします。
10. プライマリサーバーを起動します。
11. ミラーサーバーを起動します。

参照

- 「ミラーリングシステムのデータベースサーバーの停止」『SQL Anywhere サーバー データベース管理』
- 「プライマリサーバーのフェールオーバー」『SQL Anywhere サーバー データベース管理』

読み込み専用のスケールアウトシステムでの SQL Anywhere の EBF の適用

♦ 読み込み専用のスケールアウトを使用するデータベースのアップグレード

この手順では、スケールアウトがデータベースミラーリングで使用される場合も含め、スケールアウトシステムで使用されるデータベースのアップグレード方法を説明します。

読み込み専用のスケールアウトシステムで EBF を適用するには、対象ノードのデータベースを停止し、EBF をインストールします。最初にルートノードで EBF をインストールする必要はありません。

1. データベースをバックアップし、トランザクションログの名前を変更します。
2. データベースのバックアップコピーを別のコンピューターにアップグレードまたは再構築します。「バージョン 10 以降のデータベースのアップグレード手順」379 ページまたは「バージョン 10 以降のデータベースの再構築手順」371 ページを参照してください。
3. システム内のすべてのコピーノードについて、データベースサーバーを停止します。
4. ルートサーバーを停止します。スケールアウトと一緒にデータベースミラーリングを使用している場合は、プライマリサーバーとミラーサーバーの両方を停止する必要があります。
5. プライマリデータベースに対するトランザクションログの最新のコピーを保存します。
6. dbtran ユーティリティを使用して、手順 4 で保存したトランザクションログを変換します。

   このトランザクションログには、手順 1 でバックアップした以降、データベースに適用されたすべての変更が含まれます。
7. ローカルデータベースサーバーを使用して、再構築したデータベースを開始します。
8. Interactive SQL から READ 文を使用して、変換されたトランザクションログを適用します。
9. 再構築したデータベースを停止します。
10. アップグレードまたは再構築したデータベースとそのトランザクションログをルートデータベースサーバーにコピーします。
11. アップグレードしたデータベースと空のトランザクションログファイルを各コピーサーバーにコピーします。
12. ルートデータベースサーバーを起動します。スケールアウトと一緒にデータベースミラーリングを使用している場合は、プライマリサーバーとミラーサーバーの両方を起動する必要があります。

アップグレードのトラブルシューティングの情報

この項では、データベースのアップグレード時によく発生する問題について説明します。

EBF の適用時、JDBC アプリケーションが実行されていないことを確認

EBF の適用後、JDBC アプリケーションが動作を停止し、次のようなメッセージを受信する場合があります。

sajdbc.jar のビルドが共有オブジェクトのビルドと一致しません。

このメッセージは、EBF の適用時、Interactive SQL または Sybase Central の高速ランチャーや独自の JDBC アプリケーションが実行されていると返されることがあります。この場合、Java VM は、ロードしたすべての DLL または共有オブジェクトはロックしていますが JAR ファイルは

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ロックしていません。その結果、EBF を適用することによって、対応する dbjdbc および dbjodbc DLL または共有オブジェクトはアップグレードせずに sajdbc、sajdbc4、jodbc、jaodbc4 の JAR ファイルはアップデートされてしまうことが頻繁に発生します。JDBC アプリケーションが再起動されたとき、JDBC JAR ファイルと対応する DLL または共有オブジェクトが一致せず、上記のメッセージが返されます。

まず、JDBC ベースのアプリケーションをすべてシャットダウンし、EBF を再適用してください。EBF の再適用で効果がない場合、次の方法で問題を解決してください。

- **EBF インストーラーで DLL と共有オブジェクトの正常なアップデートを確認**

  ○Windows に EBF を適用する際に、*dbjdbc.dll* ファイルと *dbjodbc.dll* ファイルが正常にアップデートされたことを確認します。

  ○UNIX に EBF を適用する際に *libdbjdbc.so.1* と *libdbjodbc.so.1* の共有オブジェクトが正常にアップデートされたことを確認します。

  注意
  DLL と共有オブジェクトの確認の際には、SQL Anywhere サーバーのビット数ではなく、JAVA VM のビット数と一致するものを確認する必要があります。

- **システムに DLL と共有オブジェクトのコピーが複数ないことを確認**　DLL と共有オブジェクトが正常にアップデートされている場合、DLL または共有オブジェクトのコピーが複数ないことを確認する必要があります。これは、複数のクラスローダーに DLL や共有オブジェクトがロードされないようにする Java の制限をバイパスするために DLL または共有オブジェクトを Java VM の拡張フォルダーにコピーした場合に発生することがあります。

- **JAR ファイルが正常にアップデートされていることを確認**　DLL と共有オブジェクトが正常にアップデートされ、システムにコピーが重複していなければ、さまざまな JAR ファイルが正常にアップデートされたことを確認します。各 JAR ファイルを確認するには、次のコマンドを実行し、JAR ファイルによってレポートされた SQL Anywhere のバージョンとビルド番号が、インストールした EBF の SQL Anywhere のバージョンとビルド番号と一致することを確認します。

  sadbc.jar を確認するには、次のコマンドを実行します。

  ```
  java -cp sajdbc.jar sybase.jdbc.sqlanywhere.IBuildNum
  ```

  sajdbc4.jar を確認するには、次のコマンドを実行します。

  ```
  java -cp sajdbc4.jar sybase.jdbc4.sqlanywhere.IBuildNum
  ```

  jodbc.jar を確認するには、次のコマンドを実行します。

  ```
  java -cp jodbc.jar ianywhere.ml.jdbcodbc.jdbc3.IBuildNum
  ```

  jodbc4.jar を確認するには、次のコマンドを実行します。

  ```
  java -cp jodbc4.jar ianywhere.ml.jdbcodbc.jdbc4.IBuildNum
  ```

どの JAR ファイル、DLL、または共有オブジェクトが EBF のビルド番号と一致しないか判断できたら、アプリケーションでファイルがロックされていないことを確認し、EBF を再適用します。

Mobile Link のアップグレード

既存のソフトウェアとの互換性

- バージョン 12 の Mobile Link クライアントは、バージョン 12.0.0 より前の Mobile Link サーバーとは互換性がありません。
- 64 ビットのオペレーティングシステムでは、Mobile Link サーバーは 64 ビットのアプリケーションとしてのみインストールされ、64 ビットのアプリケーションとして動作します。旧バージョンの 12 のインストールからシステムに残っている 32 ビットの Mobile Link サーバーは 64 ビットシステムでは動作できません。
- バージョン 12 の Mobile Link サーバーは、バージョン 10 以降のクライアントで使用できます。これより前のクライアントをサポートする必要がある場合は、そのバージョンのクライアントをサポートしている古いバージョンの Mobile Link サーバーを使用する必要があります。
- 記述されている動作変更が既存のアプリケーションに影響を与えないことを確認してください。影響する場合は、既存のアプリケーションを更新してください。SQL Anywhere 12 変更点とアップグレード 1 ページを参照してください。

アップグレードの順序

既存の Mobile Link インストール環境をアップグレードするには、次の順序でコンポーネントをアップグレードします。

1. Mobile Link サーバーを停止します。
2. 統合データベースをアップグレードします。

   「統合データベースのアップグレード」387 ページを参照してください。
3. Mobile Link サーバーをアップグレードします。

   「Mobile Link サーバーのアップグレード」392 ページを参照してください。
4. Mobile Link サーバーを起動します。
5. Mobile Link クライアントをアップグレードします。

   バージョン 12 の Mobile Link サーバーは、バージョン 10 以降のクライアントでのみ使用できます。バージョン 12 の Mobile Link サーバーで操作する場合、バージョン 10 より前の Mobile Link クライアントはアップグレードする必要があります。

   SQL Anywhere リモートデータベースの詳細については、「SQL Anywhere Mobile Link クライアントのアップグレード」392 ページを参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

6. Ultra Light アプリケーションの詳細については、「以前のバージョンの Ultra Light で作成したデータベースのアップグレード」397 ページを参照してください。

アップグレードを行う前に、影響がありそうな動作の変更を確認し、一般的なアップグレード前の対応策を実行してください。

参照

- 「動作の変更と廃止予定機能」322 ページ
- 「アップグレードの注意事項」369 ページ

統合データベースのアップグレード

新しい Mobile Link サーバーと既存の統合データベースを使用できるようにするには、新しいシステムオブジェクトをインストールするアップグレードスクリプトを実行する必要があります。アップグレードスクリプトは、現在インストールされている Mobile Link システムテーブルの所有者が実行する必要があります。Mobile Link システム設定を更新するには、次の方法を使用することもできます。

- Sybase Central の Mobile Link プラグインで、**[Mobile Link 12]** » **[プロジェクト]** » **[統合データベース]** をクリックし、データベース名を右クリックして、**[Mobile Link システム設定のチェック]** をクリックします。データベースの設定が必要な場合は、続行のプロンプトが表示されます。
- **同期モデル展開ウィザード**を使用する場合は、サーバーデータベースに接続するとシステム設定がチェックされます。データベースの設定が必要な場合は、続行のプロンプトが表示されます。「同期モデル」『Mobile Link クイックスタート』を参照してください。

6.0.x の Mobile Link アップグレードスクリプトは削除されました。このアップグレードが必要な場合は、テクニカルサポート (http://www.sybase.com/support) に連絡してください。

説明

- ml_add_missing_dnld_scripts ストアドプロシージャーを使用して、欠落している download_cursor スクリプトか download_delete_cursor スクリプトまたはその両方を修正します。スクリプトバージョン名を使用してこのプロシージャーを呼び出すと、指定したスクリプトバージョンで使用されるすべての同期テーブルに対して、欠落している download_cursor スクリプトと download_delete_cursor スクリプトが無視されるスクリプトとして定義されます。
- 10.0.0 より前のバージョンで作成された authenticate_user_hashed スクリプトを使用する場合は、お使いの RDBMS でバイナリに相当する型を使用して、VARBINARY(20) ではなく VARBINARY(32) を受け入れるようにスクリプトを変更する必要があります。

統合データベースのアップグレード

♦ 統合データベースのアップグレード (SQL Anywhere 10.0.0 以降)

1. SQL Anywhere ソフトウェアをアップグレードします。

   「バージョン 10 以降のデータベースのアップグレード」366 ページを参照してください。

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

2. アップグレード元のバージョン用の適切なアップグレードスクリプトを実行して、Mobile Link システム設定をアップグレードします。

   アップグレードスクリプトは、*upgrade_sa.sql* という名前です。これは、SQL Anywhere インストール環境の *MobiLink¥upgrade¥version* にあります。*version* は、アップグレード元の SQL Anywhere バージョンを示します。

   たとえば、Interactive SQL からデータベースに接続し、次の文を実行します。

   ```
   READ "C:¥Program Files¥SQL Anywhere 12¥MobiLink¥upgrade¥10.0.x¥upgrade_sa.sql"
   ```

♦ 統合データベースのアップグレード (Adaptive Server Enterprise、Oracle、MySQL、または Microsoft SQL Server)

Adaptive Server Enterprise、Oracle、MySQL、または Microsoft SQL Server の統合データベース内の Mobile Link システムオブジェクトをアップグレードする必要があるのは、12.0.1 より前のバージョンの Mobile Link サーバーを使用している場合だけです。

1. Adaptive Server Enterprise データベースの場合は、"select into" パーミッションを設定する必要があります。Sybase Interactive SQL で次の文を実行します。

   ```
   USE MASTER
   go
   sp_dboption your-database-name, "SELECT INTO", true
   go
   USE your-database-name
   go
   checkpoint
   go
   ```

2. アップグレードするデータベースのバージョンに対応したアップグレードスクリプトを実行します。

   アップグレードスクリプトは *upgrade_XXX.sql* という名前です。*XXX* は、統合データベースの RDBMS を示します。これは、SQL Anywhere インストール環境の *MobiLink¥upgrade¥version* にあります。*version* は、アップグレード元の Mobile Link バージョンを示します。

   たとえば、バージョン 9.0.2 の Mobile Link システムテーブルが適用された Microsoft SQL Server データベースをアップグレードするには、次のコマンドを実行します。

   ```
   osql -S server_name -U user_name -P password -I
   "C:¥Program Files¥SQL Anywhere 12¥MobiLink¥upgrade¥9.0.2¥upgrade_mss.sql"
   ```

♦ IBM DB2 LUW 統合データベースのアップグレード

IBM DB2 LUW の統合データベースをアップグレードする必要があるのは、12.0.1 より前のバージョンの Mobile Link サーバーを使用している場合だけです。

IBM DB2 LUW 設定スクリプトを実行する方法については、「IBM DB2 LUW 統合データベース」『Mobile Link サーバー管理』を参照してください。

1. IBM DB2 LUW アップグレードスクリプトのロケーションを検索します。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

アップグレードスクリプトは *upgrade_db2.sql* という名前で、SQL Anywhere インストール環境の *MobiLink/upgrade/version* サブフォルダーにあります。*version* ディレクトリは、アップグレード元の Mobile Link のバージョンを示します。

2. *upgrade_db2.sql* をコピーし、このコピーを変更します。スクリプトの先頭にある CONNECT 文を変更して、接続するインスタンスで動作できるようにします。コピーした SQL スクリプトを統合データベースに適用します。

♦ 統合データベースのアップグレード (10.0.0 より前の SQL Anywhere)

●SQL Anywhere 10.0.0 より前のバージョンでは、Mobile Link システムテーブルの所有者は dbo でした。SQL Anywhere データベースの設定スクリプトを実行するには、Mobile Link システムテーブルの所有者として統合データベースにログインする必要があります。テーブルの変更パーミッションを持つユーザーとしてこのスクリプトを実行しても十分ではありません。アップグレードスクリプトを実行するには、SQL 文 SETUSER を使用して dbo を同一化する方法もあります。次に例を示します。

```
SETUSER "dbo";
```

Sybase Central で統合データベースをアップグレードするには、GRANT CONNECT 文を使用して dbo のパスワードを作成してから、dbo として接続する必要があります。次に例を示します。

```
GRANT CONNECT TO dbo IDENTIFIED BY password;
```

この場合、アップグレード後に、ALTER USER を使用して dbo のパスワードを削除する必要があります。次に例を示します。

```
ALTER USER TO dbo IDENTIFIED BY "";
```

●SQL Anywhere 統合データベースは設定済みで、同期はまだ一度も実行していない場合は、アップグレードスクリプトではなく、設定スクリプトを実行する必要があります。この手順は、SQL Anywhere 統合データベースだけに適用されます。「SQL Anywhere 統合データベース」『Mobile Link サーバー管理』を参照してください。

1. バージョン 10.0.0 より前の SQL Anywhere の統合データベースをアップグレードする場合は、先にデータベースをバージョン 12 にアップグレードする必要があります。
   a. データベースサーバーを停止します。
   b. データベースをバージョン 12 にアップグレードします。

      手順については、次の項を参照してください。

      ●「バージョン 10 以降のデータベースのアップグレード手順」379 ページ
      ●「バージョン 10 以降のデータベースの再構築手順」371 ページ
      ●「バージョン 9 以前のデータベースのバージョン 12 用への再構築」373 ページ
   c. DBA としてログインした状態で、データベースサーバーを起動します。

      アップグレードするには DBA でログインする必要があります。

2. アップグレードするデータベースのバージョンに対応したアップグレードスクリプトを実行します。

アップグレードスクリプトは、*upgrade_asa.sql* という名前です。これは、SQL Anywhere インストール環境の *MobiLink¥upgrade¥version* にあります。*version* は、アップグレード元の SQL Anywhere バージョンを示します。

アップグレードスクリプトを実行するには、dbo ユーザーを同一化する必要があります。この処理には SQL 文 SETUSER を使用します。

たとえば、SQL Anywhere バージョン 9.0.2 の統合データベースをアップグレードする場合は、Interactive SQL でデータベースに接続し、次の文を実行します。

```
SETUSER "dbo";
READ 'C:¥Program Files¥SQL Anywhere 12¥MobiLink¥upgrade¥9.0.2¥upgrade_asa.sql'
```

3. dbo のパスワードを削除します。次に例を示します。

```
GRANT CONNECT TO "dbo";
```

4. DBA 以外のユーザーとして Mobile Link サーバーを実行している場合は、新しい Mobile Link システムオブジェクトの EXECUTE パーミッションをこのユーザーに付与する必要があります。新しいシステムオブジェクトは、アップグレード対象のバージョンによって異なります。次のコードで、すべての Mobile Link システムオブジェクトへの必要なパーミッションが付与されます。このコードを実行する前に、ユーザー名 my_user を、Mobile Link サーバーを実行しているユーザーの名前に変更する必要があります。

```
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_column to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_connection_script to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_database to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_device to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_device_address to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_listening to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_passthrough to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_passthrough_repair to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_passthrough_script to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_passthrough_status to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_primary_server to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_property to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_delivery to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_delivery_archive to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_global_props to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_notifications to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_repository to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_repository_archive to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_repository_props to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_repository_props_archive to
my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_repository_staging to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_status_history to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_status_history_archive to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_qa_status_staging to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_agent to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_agent_property to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_agent_staging to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_deployed_task to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_event to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_event_staging to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_managed_remote to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_notify to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_remote_db_class to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_task to my_user;
```

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

```
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_task_command to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_task_command_property to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_ra_task_property to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_script to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_script_version to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_scripts_modified to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_sis_sync_state to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_subscription to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_table to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_table_script to my_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ml_user to my_user;
GRANT EXECUTE ON dbo.ml_add_column to my_user;
GRANT EXECUTE ON dbo.ml_add_connection_script to my_user;
GRANT EXECUTE ON dbo.ml_add_dnet_connection_script to my_user;
GRANT EXECUTE ON dbo.ml_add_dnet_table_script to my_user;
GRANT EXECUTE ON dbo.ml_add_java_connection_script to my_user;
GRANT EXECUTE ON dbo.ml_add_java_table_script to my_user;
GRANT EXECUTE ON dbo.ml_add_lang_conn_script_chk to my_user;
GRANT EXECUTE ON dbo.ml_add_lang_connection_script to my_user;
GRANT EXECUTE ON dbo.ml_add_lang_table_script to my_user;
GRANT EXECUTE ON dbo.ml_add_lang_table_script_chk to my_user;
GRANT EXECUTE ON dbo.ml_add_missing_dnld_scripts;
GRANT EXECUTE ON dbo.ml_add_passthrough to my_user;
GRANT EXECUTE ON dbo.ml_add_passthrough_repair to my_user;
GRANT EXECUTE ON dbo.ml_add_passthrough_script to my_user;
GRANT EXECUTE ON dbo.ml_add_property to my_user;
GRANT EXECUTE ON dbo.ml_add_table_script to my_user;
GRANT EXECUTE ON dbo.ml_add_user to my_user;
GRANT EXECUTE ON dbo.ml_delete_device to my_user;
GRANT EXECUTE ON dbo.ml_delete_device_address to my_user;
GRANT EXECUTE ON dbo.ml_delete_listening to my_user;
GRANT EXECUTE ON dbo.ml_delete_passthrough to my_user;
GRANT EXECUTE ON dbo.ml_delete_passthrough_repair to my_user;
GRANT EXECUTE ON dbo.ml_delete_passthrough_script to my_user;
GRANT EXECUTE ON dbo.ml_delete_remote_id to my_user;
GRANT EXECUTE ON dbo.ml_delete_sync_state to my_user;
GRANT EXECUTE ON dbo.ml_delete_sync_state_before to my_user;
GRANT EXECUTE ON dbo.ml_delete_user to my_user;
GRANT EXECUTE ON dbo.ml_delete_user_state to my_user;
GRANT EXECUTE ON dbo.ml_lock_rid to my_user;
GRANT EXECUTE ON dbo.ml_qa_add_delivery to my_user;
GRANT EXECUTE ON dbo.ml_qa_add_message to my_user;
GRANT EXECUTE ON dbo.ml_qa_handle_error to my_user;
GRANT EXECUTE ON dbo.ml_qa_stage_status_from_client to my_user;
GRANT EXECUTE ON dbo.ml_qa_staged_status_for_client to my_user;
GRANT EXECUTE ON dbo.ml_qa_upsert_global_prop to my_user;
GRANT EXECUTE ON dbo.ml_ra_add_agent_id to my_user;
GRANT EXECUTE ON dbo.ml_ra_assign_task to my_user;
GRANT EXECUTE ON dbo.ml_ra_cancel_notification to my_user;
GRANT EXECUTE ON dbo.ml_ra_cancel_task_instance to my_user;
GRANT EXECUTE ON dbo.ml_ra_clone_agent_properties to my_user;
GRANT EXECUTE ON dbo.ml_ra_delete_agent_id to my_user;
GRANT EXECUTE ON dbo.ml_ra_delete_events_before to my_user;
GRANT EXECUTE ON dbo.ml_ra_delete_task to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_agent_events to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_agent_ids to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_agent_properties to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_latest_event_id to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_orphan_taskdbs to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_remote_ids to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_task_results to my_user;
GRANT EXECUTE ON dbo.ml_ra_get_task_status to my_user;
GRANT EXECUTE ON dbo.ml_ra_int_cancel_notification to my_user;
GRANT EXECUTE ON dbo.ml_ra_int_move_events to my_user;
```

```
GRANT EXECUTE ON dbo.ml_ra_manage_remote_db;
GRANT EXECUTE ON dbo.ml_ra_notify_agent_sync to my_user;
GRANT EXECUTE ON dbo.ml_ra_notify_task to my_user;
GRANT EXECUTE ON dbo.ml_ra_reassign_taskdb to my_user;
GRANT EXECUTE ON dbo.ml_ra_set_agent_property to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_agent_auth_file_xfer to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_download_ack to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_download_prop to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_download_remote_dbs to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_download_task to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_download_task2;
GRANT EXECUTE ON dbo.ml_ra_ss_download_task_cmd to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_end_upload to my_user;
GRANT EXECUTE ON dbo.ml_ra_ss_upload_prop to my_user;
GRANT EXECUTE ON dbo.ml_ra_unmanage_remote_id to my_user;
GRANT EXECUTE ON dbo.ml_reset_sync_state to my_user;
GRANT EXECUTE ON dbo.ml_set_device to my_user;
GRANT EXECUTE ON dbo.ml_set_device_address to my_user;
GRANT EXECUTE ON dbo.ml_set_listening to my_user;
GRANT EXECUTE ON dbo.ml_set_sis_sync_state to my_user;
GRANT EXECUTE ON dbo.ml_upload_update_device_address to my_user;
GRANT EXECUTE ON dbo.ml_upload_update_listening to my_user;
```

Mobile Link サーバーのアップグレード

バージョン 12 のリモートを使用している場合は、Mobile Link サーバーをバージョン 12 にアップグレードする必要があります。

バージョン 12 の Mobile Link サーバーを使用する前に、影響がありそうな動作の変更を確認してください。SQL Anywhere 12 変更点とアップグレード 1 ページを参照してください。

Mobile Link バージョン 12 サーバーでサポートされるのは、バージョン 10、11、および 12 の SQL Anywhere クライアントと Ultra Light クライアントだけです。これより前のクライアントをサポートする必要がある場合は、そのバージョンのクライアントをサポートしている古いバージョンの Mobile Link サーバーを使用する必要があります。

SQL Anywhere Mobile Link クライアントのアップグレード

運用環境では、統合データベースと Mobile Link サーバーの両方をアップグレードしてから、SQL Anywhere リモートデータベースをアップグレードします。

バージョン 10.0.0 で、Adaptive Server Anywhere は SQL Anywhere に名前が変更されました。

検討すべきアップグレードは複数あります。

- ソフトウェアのアップグレード
- リモートデータベース自体のアップグレード
- アプリケーション全体のアップグレード

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ソフトウェアのアップグレード

dbmlsync と SQL Anywhere データベースサーバーは同時にアップグレードすることをおすすめします。バージョン 12 の Mobile Link クライアントはバージョン 12 データベースサーバーで動作するバージョン 12 データベースのみ同期できます。

バージョン 12 の Mobile Link クライアントでは、同期を実行するためにバージョン 12 以降の Mobile Link サーバーが必要です。バージョン 12 の Mobile Link クライアントは、バージョン 12 より前の Mobile Link サーバーとは同期しません。

Mobile Link のアップグレードの詳細については、「Mobile Link のアップグレード」386 ページを参照してください。

リモートの SQL Anywhere のアップグレード

♦ リモートの SQL Anywhere データベースの手動でのアンロードおよび再ロード

Mobile Link SQL Anywhere リモートデータベースは、SQL Anywhere データベースのアップグレードと同じ手順でアップグレードできます。手順については、「SQL Anywhere サーバーのアップグレード」366 ページを参照してください。

スキーマの変更やデータベースでのその他の重要な変更などがある場合は、手動でアンロードと再ロードを行う必要があります。

1. すべてのデータベースアクティビティを停止します。
2. 同期が正常に完了してから、リモートデータベースを検証し、バックアップします。
3. dbtran ユーティリティを実行してデータベースのトランザクションログの開始オフセットと終了オフセットを表示します。終了オフセットをメモしてください。

   「ログ変換ユーティリティ (dbtran)」『SQL Anywhere サーバー データベース管理』を参照してください。
4. トランザクションログの名前変更これにより、アンロード中にトランザクションログが修正されるのを回避できます。名前変更したログファイルを、オフラインディレクトリなどの安全なロケーションに移動します。
5. データベースをアンロードします。

   「バージョン 9 以前のデータベースのバージョン 12 用への再構築」373 ページを参照してください。
6. 新しいデータベースを初期化します。

   「初期化ユーティリティ (dbinit)」『SQL Anywhere サーバー データベース管理』を参照してください。
7. データを新しいデータベースに再ロードします。

   「バージョン 9 以前のデータベースのバージョン 12 用への再構築」373 ページを参照してください。

8. 新しいデータベースを停止します。
9. 新しいデータベースのトランザクションログを消去します。
10. 次のオプションを使用して、新しいデータベースで dblog を実行します。
    - -z を使用して、メモした終了オフセットを指定します。
    - -x を使用して、相対オフセットをゼロに設定します。

    次に例を示します。

    ```
    dblog -x 0 -z 137829 database-name.db
    ```

    「トランザクションログユーティリティ (dblog)」『SQL Anywhere サーバー データベース管理』を参照してください。
11. dbmlsync を起動し、移動した元のログファイルの場所を指定します。

    「dbmlsync 構文」『Mobile Link クライアント管理』を参照してください。
12. 古いログファイルが不要な場合は、データベースオプション delete_old_logs を設定します。

    「delete_old_logs オプション [SQL Remote]」『SQL Anywhere サーバー データベース管理』を参照してください。

アプリケーションのアップグレード

Mobile Link アプリケーションの新しいバージョンを配備する場合、同期スクリプトには新しいスクリプトバージョンを使用することをおすすめします。たとえば、既存のアプリケーションで **v1** というスクリプトバージョンを使用している場合は、アップグレードしたアプリケーションでは **v2** という名前のスクリプトバージョンを使用します。この 2 つのスクリプトバージョンは同時に使用できます。これにより、一度にではなく、段階的にリモートデータベースをアップグレードすることがより簡単になります。

バージョン 9.0.0 以降では、Mobile Link サーバー の -zd オプションは削除されています。配備環境で -zd オプションを使用している場合、アップグレードを行うには、最後のダウンロードタイムスタンプが最初のパラメーターとして採用されるようダウンロードスクリプトを変更します。また、クライアントをアップグレードして名前付きパラメーターの使用を開始することで、スクリプトパラメーターを任意の順序で配置できます。

Mobile Link のアップグレード時の SQL Anywhere モニター

SQL Anywhere モニターは、Mobile Link サーバーをアップグレードする前または後にいつでもアップグレードできます。

不要な警告を回避するため、サーバーを停止する前に、サーバーの SQL Anywhere モニターでブラックアウト期間をスケジュールしてください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

QAnywhere のアップグレード

QAnywhere アプリケーションをアップグレードすると、統合データベース、アプリケーション、クライアントメッセージストアをアップグレードできます。

統合データベースをアップグレードする方法の詳細については、「統合データベースのアップグレード」387 ページを参照してください。

アプリケーションをアップグレードする場合は、このリリースの新しい機能と変更された動作を確認してください。 SQL Anywhere 12 変更点とアップグレード 1 ページを参照してください。

♦ QAnywhere メッセージストアのアップグレード

1. QAnywhere ファイルを配備します。

   「QAnywhere アプリケーションの配備」『Mobile Link サーバー管理』を参照してください。

2. 次の手順に従って、メッセージストアをアップグレードします。

   -su オプションまたは -sur オプションを指定して、QAnywhere Agent を実行します。次の項を参照してください。

   - 「-su qaagent オプション」『QAnywhere』
   - 「-sur qaagent オプション」『QAnywhere』

QAnywhere Ultra Light クライアントとスタンドアロンクライアントのアップグレード

12.0.1 より前のメッセージストアをアップグレードするには、メッセージストアを消去してから 12.0.1 ユーティリティを使用して再作成します。新しいメッセージストアを初めて同期するときに、クライアントプロパティと受信確認されていないメッセージがすべてメッセージストアに伝達されます。

♦ QAnywhere Ultra Light クライアントのアップグレード

1. プロシージャーのアップグレードを開始する直前にメッセージストアの同期を実行します。

   警告
   作成されたがアップグレード前にサーバーに同期されていないメッセージやプロパティは、すべて失われます。

2. 対応する次の Ultra Light データベースを削除して、メッセージストアを削除します。del qanywhere.udb

3. 12.0.1 ユーティリティを使用して、メッセージストアを再作成し、再初期化します。「クライアントメッセージストアの設定」『QAnywhere』を参照してください。

4. 初期同期を実行して、メッセージストアを再配置します。

♦ QAnywhere スタンドアロンクライアントのアップグレード

1. プロシージャーのアップグレードを開始する直前にメッセージストアの同期を実行します。

警告
作成されたがアップグレード前にサーバーに同期されていないメッセージやプロパティは、すべて失われます。

2. 対応する次の Ultra Light データベースを削除して、メッセージストアを削除します。del qanywhere.udb
3. QAnywhere クライアントアプリケーションを次回に起動するときに、メッセージストアが自動的に再作成されます。
4. 初期同期を実行して、メッセージストアを再配置します。

Ultra Light のアップグレード

このバージョンのソフトウェアで既存のアプリケーションを使用する前に、新機能と動作の変更のリストを確認して、アプリケーションに影響がないかどうかを確認してください。次の項を参照してください。

- 「Ultra Light の新機能」195 ページ
- 「Ultra Light の動作の変更と廃止予定機能」200 ページ

Ultra Light Java Edition

Ultra Light Java Edition データベースを Ultra Light 12.0.1 で開くと、データベースがアップグレードされて、Ultra Light 12.0.0 と互換性がなくなります。

Ultra Light ユーティリティの使用

SQL Anywhere の複数のバージョンが同じコンピューターにインストールされている場合は、システムパスに注意して、使用している Ultra Light ユーティリティがバージョン 12 であることを確認してください。「ユーティリティのバージョンとアップグレード手順」369 ページを参照してください。

Ultra Light バージョン 12

Ultra Light のファイルフォーマットは、バージョン 9 と 10 の間に変更され、バージョン 11 と 12 の間に再度変更されました。Ultra Light バージョン 12 では、このソフトウェアの以前のバージョンを使用して作成された Ultra Light データベースを読み込むことはできません。

バージョン 10 と 11 のデータベースのアップグレード

バージョン 10 と 11 の Ultra Light データベースの場合、バージョン 11 の ulunloadold ユーティリティを使用して ulload で使用できる XML ファイルを生成してから、バージョン 12 のデータベースを作成します。データベースがアップグレードされたら、バージョン 10 または 11 のアプリケーション、ユーティリティ、ソフトウェアに接続できなくなります。「以前のバージョンの Ultra Light で作成したデータベースのアップグレード」397 ページを参照してください。

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

バージョン 9 以前のデータベースのアップグレード

バージョン 9 以前の Ultra Light データベースの場合、バージョン 9 の ulunloadold ユーティリティを使用して ulload で使用できる XML ファイルを生成してから、バージョン 12 のデータベースを作成します。Ultra Light 12 でバージョン 9 以前のデータベースに接続しようとすると、データベースの開始時にエラーが発生します。「以前のバージョンの Ultra Light で作成したデータベースのアップグレード」397 ページを参照してください。

既存のソフトウェアとの互換性

- バージョン 12 では、Ultra Light for AppForge はサポートされていません。
- Ultra Light 12 のデータベースファイルは、バージョン 12 のクライアントアプリケーション、またはバージョン 12 の Ultra Light エンジンからの接続のみをサポートします。
- Ultra Light バージョン 12 ランタイムと Ultra Light バージョン 12 エンジンは、バージョン 9 以前の Ultra Light で作成したデータベースファイルおよびアプリケーションコードでは使用できません。

注意
Palm OS は、Ultra Light バージョン 12 の時点でサポートされなくなりました。

以前のバージョンの Ultra Light で作成したデータベースのアップグレード

Ultra Light のファイルフォーマットは、バージョン 9 と 10 の間に変更され、バージョン 11 と 12 の間に再度変更されました。Ultra Light バージョン 12 では、このソフトウェアの以前のバージョンを使用して作成された Ultra Light データベースを読み込むことはできません。

説明

- データベースは、デスクトップでのみアップグレードできます。
- データベースのバックアップコピーを作成してください。
- 同期されていない変更が含まれている可能性がある運用データベースの場合は、データベースを同期します。

♦ Windows での Ultra Light データベースのアップグレード

1. バージョン 9 以前の Ultra Light でデータベースを作成した場合は、コマンドプロンプトを開き、*%SQLANY12%¥UltraLite¥Unload¥V9* に移動します。

   バージョン 10 または 11 の Ultra Light でデータベースを作成した場合は、コマンドプロンプトを開き、*%SQLANY12%¥UltraLite¥Unload¥V11* に移動します。

2. ulunload ユーティリティを実行して、データベースの内容を含む XML ファイルを作成します。

3. ulload コマンドラインユーティリティを使用するか、または Sybase Central の Ultra Light プラグインで**データベースロードウィザード**を使用して、スキーマとデータを新しいバージョン 12 のデータベースにロードします。Ultra Light データベースのデフォルトの暗号化には、現在 UTF8 が使用されています。これを変更したい場合は、utf8_encoding パラメーターを明示的にオフに設定する必要があります。

4. 生成された XML ファイルをチェックして、UTF-8 エンコーディングの設定を確認します。「Ultra Light utf8_encoding 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

♦ Linux での Ultra Light データベースのアップグレード

1. コマンドプロンプトを開いて、*$SQLANY12/ultraLite/unload/v11* に進みます。

2. ulunload ユーティリティを実行して、データベースの内容を含む XML ファイルを作成します。

3. ulload コマンドラインユーティリティを使用するか、または Sybase Central の Ultra Light プラグインで**データベースロードウィザード**を使用して、スキーマとデータを新しいバージョン 12 のデータベースにロードします。Ultra Light データベースのデフォルトの暗号化には、現在 UTF8 が使用されています。これを変更したい場合は、utf8_encoding パラメーターを明示的にオフに設定する必要があります。

4. 生成された XML ファイルをチェックして、UTF-8 エンコーディングの設定を確認します。「Ultra Light utf8_encoding 作成パラメーター」『Ultra Light データベース管理とリファレンス』を参照してください。

SQL Remote のアップグレード

SQL Remote インストール環境には、1 つの統合データベースと多数のリモートデータベースが含まれています。また、各サイトには、SQL Remote Message Agent があります。

それぞれのサイトでは、SQL Remote Message Agent がメッセージの送受信を行います。メッセージは SQL 文の形式で、データベースサーバーがその SQL 文の実行を処理します。

SQL Remote のアップグレード要件は、次のとおりです。

- **ソフトウェアアップグレードは一度に 1 サイトずつ実行できる**　旧バージョンの Message Agent (dbremote) は、バージョン 12 の Message Agent とメッセージを交換することができます。バージョン 5 の SQL Remote では、compression データベースオプションの値が -1 (マイナス 1) に設定されている場合、バージョン 5 の Message Agent とバージョン 12 の Message Agent でメッセージを交換できます。インストール環境全体のソフトウェアを同時にアップグレードする必要はありません。「compression オプション [SQL Remote]」『SQL Anywhere サーバー データベース管理』を参照してください。

- **データベースのアップグレード**　SQL Anywhere バージョン 9 以前を使用していたリモートデータベースまたは統合データベースをアップグレードする場合は、データベースをアン

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ロードして再ロードすることでデータベースファイルフォーマットをアップグレードする必要があります。すべてのデータベースを同時にアップグレードする必要はありません。

データベースをアンロードおよび再ロードする方法については、「バージョン 9 以前のデータベースのバージョン 12 用への再構築」 373 ページを参照してください。

- **Adaptive Server Enterprise 統合データベースのアップグレード**　SQL Remote では、Adaptive Server Enterprise 統合データベースはサポートされなくなりました。Adaptive Server Enterprise データベースを同期するには、Mobile Link にアップグレードする必要があります。

  SQL Remote から Mobile Link への移行については、http://www.sybase.com/detail?id=1034174 を参照してください。

- **SQL Remote のアップグレード**　バージョン 11.0.0 では、SQL Remote の VIM と MAPI の各メッセージシステムがサポートされなくなりました。VIM または MAPI を使用するデータベースを SQL Anywhere バージョン 12 にアップグレードする場合は、メッセージタイプを File、FTP、または SMTP に変更してください。メッセージタイプが MAPI または VIM の場合、*dbremote.exe* は起動しません。

例

以下に、バージョン 5 の SQL Remote のアップグレードの一例を示します。

1. 統合データベースサーバーと SQL Remote Message Agent をアップグレードしてから、統合データベースをアンロードして再ロードすることでデータベースファイルをアップグレードします。すべてのメッセージがリモートサイトのバージョン 5 のソフトウェアと互換性を持つように、compression データベースオプションを -1 に設定してください。 統合データベースのアンロードと再ロードの方法については、「同期やレプリケーションに関連するデータベースの再構築」『SQL Anywhere サーバー SQL の使用法』を参照してください。

2. リモートデータベースサーバーと Message Agent を 1 つずつアップグレードしてから、リモートデータベースをアンロードして再ロードすることでデータベースファイルフォーマットをアップグレードします。compression データベースオプションを -1 以外の値に設定すると、統合データベースサーバーに送信されるメッセージに対して圧縮機能とコード化機能を利用できます。リモートデータベースのアンロードと再ロードの方法については、「同期やレプリケーションに関連するデータベースの再構築」『SQL Anywhere サーバー SQL の使用法』を参照してください。

3. すべてのリモートデータベースサーバーと Message Agent をアップグレードし終わったら、統合サイトの compression データベースオプションを -1 以外の値に設定します。

SQL Anywhere モニターのアップグレードおよびリソースとメトリックの移行

♦ 移行ユーティリティを使用したモニターリソースとデータの移行

1 つのモニターから、**新しく**作成したモニターへリソースとメトリックを移行するには、移行ユーティリティを使用します。

警告
モニターをアンインストールすると、アプリケーションだけでなく、リソース、収集したメトリックも削除されます。

現在のモニターのリソースとメトリックを保持する場合は、次の操作を実行します。

1. モニターの新しいバージョンをインストールします。
2. リソースとメトリックを移行します。
3. モニターの古いバージョンをアンインストールします。

1. 既存のモニターデータベースファイル *samonitor.db* をコピーします。たとえば、11.0.1 モニターに移行するには、次のファイルをコピーします。

   C:¥Documents and Settings¥All Users¥Documents¥SQL Anywhere 11¥Monitor¥samonitor.db

2. 新しいモニターをインストールします。インストールメディアの *Monitor* ディレクトリにある *setup.exe* ファイルを実行し、表示される指示に従います。インストールが完了したら、新しいモニターを停止します (稼働している場合)。

   注意
   コンピューター上で一度に実行できるモニターのバージョンは 1 つだけです。

   バージョン 11.0.1 のモニターが稼働しているコンピューターにバージョン 12.0.1 のモニターをインストールすると、バージョン 11.0.1 のモニターが停止します。

3. コマンドプロンプトで移行ユーティリティを実行します。次の表を使用して、移行ユーティリティのロケーションを確認します。

オペレーティングシステム	モニタータイプ	移行ユーティリティのデフォルトロケーション
Windows	モニター Developer Edition	*C:¥Program Files¥SQL Anywhere 12¥Bin32¥run_migrator.cmd*
	モニター Production Edition	*C:¥Program Files¥SQL Anywhere 12¥Bin32¥run_migrator.cmd*
Linux	モニター Developer Edition	*/opt/sqlanywhere12/bin32/run_migrator.sh*
	モニター Production Edition	*/opt/samonitor12/bin32/run_migrator.sh* または */opt/samonitor12/bin64/run_migrator.sh*

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

次のオプションを指定して移行ユーティリティを実行します。

- **-c**　リソースと構成設定のみを移行することを指定します。デフォルトでは、リソースと収集されたデータの両方が移行されます。
- **-t temporary-directory**　テンポラリファイルのディレクトリを指定します。デフォルトでは、テンポラリファイルは、run_migrator ファイルと同じディレクトリに作成されます。

> **注意**
> モニター移行で作成されるテンポラリファイルは、移行プロセスの最後で削除されます。-t オプションを使用して、これらのテンポラリファイルのディレクトリを指定します。テンポラリファイルには、バージョン 11.0.1 のモニターデータベースファイルと同様の領域が必要です。指定のディレクトリに十分な領域があることを確認してください。

- **source-filename**　データのアンロード元となる古いモニターファイルのパスとファイル名を指定します。たとえば、バージョン 11.0.1 の *samonitor.db* ファイルのパスを指定します。
- ***destination-filename***　データの再ロード先となる新しいモニターファイルのパスとファイル名を指定します。たとえば、バージョン 12.0.1 の *samonitor.db* ファイルのパスを指定します。

次に例を示します。

```
C:¥Program Files¥SQL Anywhere 12¥run_migrator.cmd -t c:¥monitorbackup c:¥Program Files
¥SQL Anywhere 11¥Monitor¥samonitor11.db C:¥Program Files¥SQL Anywhere 12¥Monitor
¥samonitor12.db
```

バージョン 11.0.1 モニターおよびバージョン 12.0.1 モニターのデータベースファイルのデフォルトのロケーションを次の表に示します。

オペレーティングシステム	モニタータイプ	11.0.1 ディレクトリ	12.0.1 ディレクトリ
Windows XP	モニター Developer Edition	C:¥Documents and Settings¥All Users ¥Documents¥SQL Anywhere 11¥Monitor ¥samonitor.db	C:¥Documents and Settings¥All Users ¥Documents¥SQL Anywhere 12¥Monitor ¥samonitor.db
Windows XP	モニター Production Edition	C:¥Documents and Settings¥All Users ¥Documents¥SQL Anywhere 11 Monitor ¥samonitor.db	C:¥Documents and Settings¥All Users ¥Documents¥SQL Anywhere 12 Monitor ¥samonitor.db
Windows Vista 以降のバージョンの Windows	モニター Developer Edition	C:¥Users¥Public ¥Documents¥SQL Anywhere 11¥Monitor ¥samonitor.db	C:¥Users¥Public ¥Documents¥SQL Anywhere 12¥Monitor ¥samonitor.db
Windows Vista 以降のバージョンの Windows	モニター Production Edition	C:¥Users¥Public ¥Documents¥SQL Anywhere 11 Monitor ¥samonitor.db	C:¥Users¥Public ¥Documents¥SQL Anywhere 12 Monitor ¥samonitor.db

オペレーティングシステム	モニタータイプ	11.0.1 ディレクトリ	12.0.1 ディレクトリ
Linux	モニター Developer Edition	/opt/sqlanywhere11/samonitor.db	/opt/sqlanywhere12/samonitor.db
Linux	モニター Production Edition	/opt/samonitor11/samonitor.db	/opt/sqlanywhere12/samonitor.db

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

索引

記号

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

A

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

B

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

C

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

D

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

E

F

G

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

H

I

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

J

K

L

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

M

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

N

O

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

P

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

Q

R

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

S

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

T

U

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

V

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

W

X

あ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

い

う

え

お

か

き

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

く

け

こ

さ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

し

す

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

せ

そ

た

ち

つ

て

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

と

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

な

に

ね

の

は

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ひ

ふ

へ

ほ

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

よ

ら

り

 Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1

ろ

わ

Copyright © 2012, iAnywhere Solutions, Inc. - SQL Anywhere 12.0.1