



PUBLIC

SQL Anywhere

Document Version: 17.01.0 – 2021-10-15

SQL Anywhere What's New

Content

- 1 SQL Anywhere - Changes and Upgrading. 3**
- 1.1 Upgrading to SQL Anywhere 17.0 Support Package 01 PL 08 (Build 7173). 4
- 1.2 Upgrading to SQL Anywhere 17.0 Support Package 01. 4
- 1.3 Upgrading to SQL Anywhere 17.0 Support Package PL 67 Build 6230. 7
- 1.4 Upgrading to SQL Anywhere 17.0 Support Package PL 41 Build 5745. 8
- 1.5 Upgrading to SQL Anywhere 17.0 Support Package PL 29 Build 4793. 13
- 1.6 Upgrading to SQL Anywhere 17.0 Support Package PL 22 Build 4003. 16
- 1.7 Upgrading to SQL Anywhere 17.0 (build 2000). 19
- 1.8 What's New in SQL Anywhere Version 17.0 (Product-Wide). 29
- 1.9 What's New in SQL Anywhere Server 17.0. 36
 - Performance Enhancements. 43
 - Changes to SQL Statements, Functions, Procedures, and Data Types. 45
 - Changes to Database Administration. 49
 - Changes to the Catalog. 53
 - Changes to Programming Interfaces. 54
 - Changes to Administration Tools. 56
 - Changes to OData Support. 60
 - Changes to Security. 62
 - Documentation Enhancements. 65
 - Miscellaneous Changes and Enhancements. 66
 - Behavior Changes, Deprecated Features, and Features That Are No Longer Supported. 70
- 1.10 What's New in MobiLink Version 17.0. 76
- 1.11 What's New in UltraLite Version 17.0. 80
- 1.12 What's New in Relay Server Version 17.0. 82
- 1.13 What's New in SQL Remote Version 17.0. 85
- 1.14 What's New in Previous Releases of SQL Anywhere. 86
- 1.15 How to Upgrade to the Latest Version of SQL Anywhere. 87
 - SQL Anywhere Server Upgrades. 89
 - MobiLink Upgrades. 125
 - UltraLite Upgrades. 138
 - SQL Remote Upgrades. 142
 - Upgrading the Monitor and Migrating Resources. 143
 - Upgrading Version 16 Databases to Use the SQL Anywhere Cockpit. 145
 - Upgrading Version 17 Databases to Use the SQL Anywhere Cockpit. 146
 - Converting the OData Server from Version 16. 148

1 SQL Anywhere - Changes and Upgrading

This book describes new features in *SQL Anywhere 17* and in previous versions of the software. It also explains how to upgrade previous releases to the latest version of *SQL Anywhere*.

In this section:

[Upgrading to SQL Anywhere 17.0 Support Package 01 PL 08 \(Build 7173\) \[page 4\]](#)

SQL Anywhere 17.0 SP01 PL 08 introduces new enhancements.

[Upgrading to SQL Anywhere 17.0 Support Package 01 \[page 4\]](#)

SQL Anywhere 17.0 SP01 introduces several enhancements.

[Upgrading to SQL Anywhere 17.0 Support Package PL 67 Build 6230 \[page 7\]](#)

SQL Anywhere 17.0 PL 67 Build 6230 introduces several enhancements.

[Upgrading to SQL Anywhere 17.0 Support Package PL 41 Build 5745 \[page 8\]](#)

SQL Anywhere 17.0 PL 41 Build 5745 introduces several enhancements.

[Upgrading to SQL Anywhere 17.0 Support Package PL 29 Build 4793 \[page 13\]](#)

SQL Anywhere 17.0 PL 29 Build 4793 introduces several enhancements.

[Upgrading to SQL Anywhere 17.0 Support Package PL 22 Build 4003 \[page 16\]](#)

SQL Anywhere 17.0 PL22 Build 4003 introduces several enhancements.

[Upgrading to SQL Anywhere 17.0 \(build 2000\) \[page 19\]](#)

As of build 2000, SQL Anywhere 17.0 contains several minor enhancements and behavior changes.

[What's New in SQL Anywhere Version 17.0 \(Product-Wide\) \[page 29\]](#)

SQL Anywhere 17.0 introduces several product-wide new and changed features to learn about, in addition to component-level changes.

[What's New in SQL Anywhere Server 17.0 \[page 36\]](#)

SQL Anywhere 17.0 introduces many new features and changes across the software in the areas of programming interfaces, security, performance, SQL support, catalog changes, and administration tools.

[What's New in MobiLink Version 17.0 \[page 76\]](#)

MobiLink version 17.0 introduces several new, changed, deprecated, or removed features.

[What's New in UltraLite Version 17.0 \[page 80\]](#)

UltraLite version 17.0 introduces new, changed, deprecated, or removed features.

[What's New in Relay Server Version 17.0 \[page 82\]](#)

Relay Server version 17.0 introduces several new, changed, deprecated, or removed features.

[What's New in SQL Remote Version 17.0 \[page 85\]](#)

SQL Remote version 17.0 includes enhancements and behavior changes.

[What's New in Previous Releases of SQL Anywhere \[page 86\]](#)

Use the following links to access previous releases of the SQL Anywhere documentation, where you can read about how and when features were added, enhanced, deprecated, and removed.

[How to Upgrade to the Latest Version of SQL Anywhere \[page 87\]](#)

Major releases, service packs, and patch level updates include enhancements to the software that can require you to upgrade or rebuild your database.

1.1 Upgrading to SQL Anywhere 17.0 Support Package 01 PL 08 (Build 7173)

SQL Anywhere 17.0 SP01 PL 08 introduces new enhancements.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

SQL Anywhere Enhancements

Stronger Algorithms for Database Encryption New encryption algorithms were added for database encryption. This enhancement requires build **7163** or later.

- AES256CTR - AES 256-bit algorithm with CTR block cipher mode
- AES256CTR_FIPS - FIPS-certified version of the AES 256-bit algorithm with CTR block cipher mode
- ARIA256 - ARIA 256-bit algorithm with CBC block cipher mode
- ARIA256CTR - ARIA 256-bit algorithm with CTR block cipher mode

See:

- [CREATE DATABASE Statement](#)
- [Encryption Algorithm Aliases](#)
- [DECRYPT Function \[String\]](#)
- [ENCRYPT Function \[String\]](#)

SHA384 and SHA512 Added to HASH Function The HASH function was enhanced to support the SHA384 and SHA512 algorithms. This enhancement requires build **7160** or later. See [HASH Function \[String\]](#).

1.2 Upgrading to SQL Anywhere 17.0 Support Package 01

SQL Anywhere 17.0 SP01 introduces several enhancements.

To use some of these enhancements and changes, you must upgrade or rebuild your database. When you upgrade or rebuild a SQL Anywhere database to version 17.0 SP01 or later, you cannot run the database on a database server earlier than version 17.0 SP01.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

SQL Anywhere enhancements

parameterization_level option

The default value for the `parameterization_level` option is now Off. Previously, the default value was Simple. [parameterization_level Option](#)

-orp database parameter

You can now specify the `-ek` database option or `-ep` database server option with the `-orp` database server option. [-orp Database Server Option](#)

sni_hostname parameter

When acting as a TLS client, SQL Anywhere and client libraries now support the Server Name Indication (SNI) parameter `sni_hostname`. [CREATE PROCEDURE Statement \[Web Service\]](#), [CREATE FUNCTION Statement \[Web Service\]](#)

File search algorithm revised for improved security

The Microsoft Windows app data and UNIX \$HOME directories are excluded from searches for resource files (`.res` on UNIX, `.dll` on Microsoft Windows).

MiniDumpType option

The SQL Anywhere database server now supports setting the crash dump type to NORMAL or FULL. To set the crash dump type, use the `-md {NORMAL | FULL}` engine option or set `sa_server_option('MiniDumpType', ' {NORMAL | FULL}')`. NORMAL (default) creates a mini crash dump. On UNIX platforms, only NORMAL is supported. [List of Database Server Properties](#), [sa_server_option System Procedure](#), [SQL Anywhere Database Server Executable \(dbsrv17, dbeng17\)](#), [-md Database Server Option](#)

min_tls_version option

The default value for this option has changed from 1.0 to 1.2. [Encryption \(ENC\) Connection Parameter](#)

Support for TDS connections over TLS

SQL Anywhere now supports TDS connections over TLS. Previously, only unencrypted TDS connections were supported. [-ec Database Server Option](#)

Database Server Properties Database server properties now include `StatisticsCleaner`, `DropBadStatistics`, and `DropUserStatistics`. [List of Database Server Properties](#)

.NET Core SQL Anywhere now supports the .NET Core Provider. [SQL Anywhere .NET Core Data Provider](#)

Microsecond TIMESTAMP SQL Anywhere database servers running on Windows versions now support microsecond TIMESTAMP accuracy for special values such as `CURRENT TIMESTAMP`, `TIMESTAMP`, `CURRENT UTC TIMESTAMP`, etc. Previously, only millisecond TIMESTAMP accuracy was supported.

-lt n Database Server Option You can now control the number of lock tables used by SQL Anywhere to record locks on rows. [-lt Databases Server Option](#)

LockTableContentionCount Database Server Property This new property shows the number of times that significant contention has occurred on any of the lock tables. [List of Database Server Properties](#)

NumLockTables Database Server Property This new property shows the number of internal lock tables used by the database. [List of Database Server Properties](#)

allow_optional_semicolon_in_tsq Option When this new option is On, semicolons are allowed as optional statement delimiters in the TSQL dialect, and some mixed dialect procedure definitions that were valid in SA16 will be treated as syntax errors. [allow_optional_semicolon_in_tsq Option](#)

sa_db_option System Procedure The default value for the system procedure sa_db_option using the CollectStmtPerfStats option has changed from ON to OFF. [sa_db_option System Procedure](#)

ODBC SQLSetConnectAttr SQL_ATTR_CURRENT_CATALOG can be set before or after connecting to a database server using SQLSetConnectAttr. The name that is specified by this call defines the default database that the ODBC application will connect to if the database name has not been specified in any other way, for example, using the DatabaseName (DBN) connection parameter. [SQLSetConnectAttr Extended Connection Attributes](#)

LOCK TABLE Statement When IN EXCLUSIVE MODE is specified, any dependent views on that table are locked as well. [LOCK TABLE Statement](#)

MobiLink enhancements

min_tls_version option

The default minimum TLS version for secure network connections has changed from 1.0 to 1.2. [min_tls_version MobiLink Client Network Protocol Option](#)

Open source updates

The SQL Anywhere installer has changed as follows:

Action	Before	After
Replace	<pre>java\commons- collections-3.2.1.jar - java \velocity-1.7-dep.jar</pre>	<pre>java\commons- collections4-4.2.jar java\velocity-engine- core-2.0-dep.jar</pre>
Add		<pre>java\commons-io-2.6 java\commons- lang3-3.8.1.jar java\slf4j-1.7.25.jar</pre>

If you perform your own installs, you may need to update your installer for this change.

HTTPS Session Cookies The MobiLink server now supports the session_id_cookie or client_id_cookie stream options, which allow the server to set a cookie with the given name containing the ml-session-id or ml-client-id value, respectively. [-x mlsrv17 Option](#)

Support for Oracle 19c consolidated database The MobiLink server now supports Oracle 19c consolidated databases.

Support for IBM DB2 v11.5 MobiLink now supports IBM DB2 v11.5.

Monitor and Control MobiLink Lock Tables The database server option -lt has been added to control the number of lock tables on each database. The database properties LockTableContentionCount and

NumLockTables have been added to monitor serious contention and the number of internal lock tables used. -It [Databases Server Option, List of Database Properties](#)

UltraLite enhancements

min_tls_version option

The default value for this option has changed from 1.0 to 1.2. [min_tls_version MobiLink Client Network Protocol Option](#)

64-bit Android Support UltraLite now provides 64-bit Android support. [Deploying an UltraLiteJ application for Android](#)

Application ID SAP_MOBILINK_HDI The MobiLink server now provides the application ID, SAP_MOBILINK_HDI using SQLSetConnectAttr before calling SQLDriverConnect, when it is connecting to an SAP HANA database. This attribute is ignored by all other ODBC drivers.

1.3 Upgrading to SQL Anywhere 17.0 Support Package PL 67 Build 6230

SQL Anywhere 17.0 PL 67 Build 6230 introduces several enhancements.

To use some of these enhancements and changes, you must upgrade or rebuild your database. When you upgrade or rebuild a SQL Anywhere database to version 17.0 build 6230 or later, you cannot run the database on a database server earlier than version 17.0 build 6230.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

Product-wide enhancements

SQL Anywhere Monitor Adobe will stop updating and distributing the Flash Player at the end of 2020. Because the SQL Anywhere Monitor is based on Flash, you cannot use it once Flash support ends. In many cases, tasks that were previously performed in the Monitor can be performed in the SQL Anywhere Cockpit. See [SQL Anywhere Monitor Non-GUI User Guide](#).

Specify the minimum level of TLS support

To enable or disable older versions of TLS when necessary, use `MIN_TLS_VERSION=ver`. For example, to allow TLS version 1 connections over HTTPS, specify `dbsrv17 -xs https (MIN_TLS_VERSION=1.1; . . .)`. Valid versions are **1.0** (the default), **1.1**, and **1.2**. The period is

optional, so you can also specify *10*, *11*, or *12*. Versions of TLS older than TLS version 1 (such as SSL v3 or SSI v3) cannot be enabled.

- [-ec Database Server Option](#)
- [-xs Database Server Option](#)
- [Encryption \(ENC\) Connection Parameter](#)
- [-x mlsrv17 Option](#)
- [min_tls_version MobiLink Client Network Protocol Option](#)

1.4 Upgrading to SQL Anywhere 17.0 Support Package PL 41 Build 5745

SQL Anywhere 17.0 PL 41 Build 5745 introduces several enhancements.

To use some of these enhancements and changes, you must upgrade or rebuild your database. When you upgrade or rebuild a SQL Anywhere database to version 17.0 build 5745 or later, you cannot run the database on a database server earlier than version 17.0 build 5745.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

Product-wide enhancements

Encryption now uses the SAP Cryptographic Library (CommonCryptoLib) In previous releases, SQL Anywhere provided a 64-bit FIPS compliant module for encryption from OpenSSL. The software now uses SAP CommonCryptoLib to provide this functionality. The following changes have been made to support this change:

allow_expired_certs option deprecated

The `allow_expired_certs` option is now ignored because you can no longer use expired certificates for TLS/HTTPS communications. This change affects the `-ec` database server option, the Encryption (ENC) connection parameter, the CREATE PROCEDURE statement [Web service], the CREATE FUNCTION statement [Web service], and the `allow_expired_certs` protocol option.

Deployment changes

The files that you deploy all installations has changed:

Operating system	Removed files	Added files
Microsoft Windows	N/A	sapcrypto.dll
Linux/UNIX	N/A	libsapcrypto.so
macOS	N/A	libsapcrypto.dylib

The files that you deploy to provide FIPS compliant encryption have changed:

Operating system	Removed files	Added files
Microsoft Windows	<ul style="list-style-type: none"> • libeay32.dll • ssleay32.dll 	<ul style="list-style-type: none"> • sapcrypto.dll • sapcryptofips.dll • slcryptokernel.dll • slcryptokernel.dll.sha256
Linux/UNIX	<ul style="list-style-type: none"> • libdbssl.so • libdbssl.so.1.0.0 • libdbcrypto.so.1.0 	<ul style="list-style-type: none"> • libsapcrypto.so • libsapcryptofips.so • libslcryptokernel.so • libslcryptokernel.so.sha256
macOS (FIPS compliant encryption is not supported on macOS)	N/A	N/A

macOS software is 64-bit only

32-bit software for macOS is no longer provided. Use the 64-bit mac OS software. The following files have been removed from the product to support this change:

- System/bin32/mltemplate
- System/bin32/dbisqlc
- System/bin32/dbfhide
- System/bin32/dbdsn
- System/bin32/dbbackup
- System/bin32/dbns17
- System/bin32/dblocate
- System/bin32/dbsupport
- System/bin32/dbping
- System/bin32/dbtsinfo
- System/bin32/newdemo.sh
- System/bin32/dbversion
- System/bin32/dbvalid
- System/bin32/dbupgrad
- System/bin32/dbunload
- System/bin32/sa_config.sh

- System/bin32/sa_config.csh
- System/bin32s/newdemo.sh
- System/bin32s/mltemplate
- System/bin32s/dbversion
- System/bin32s/dbvalid
- System/bin32s/dbupgrad
- System/bin32s/dbunload
- System/bin32s/dbtsinfo
- System/bin32s/dbsupport
- System/bin32s/dbping
- System/bin32s/dbns17
- System/bin32s/dblocate
- System/bin32s/dbisqlc
- System/bin32s/dbfhide
- System/bin32s/dbdsn
- System/bin32s/dbbackup
- System/lib32/libdbldap17.dylib
- System/lib32/libdbscript17_r.dylib
- System/lib32/libdbldap17_r.dylib
- System/lib32/libdbrsa17_r.dylib
- System/lib32/libdbrsa17.dylib
- System/lib32/dbodbc17.bundle
- System/lib32/libdbrsakp17_r.dylib
- System/lib32/dbodbcansi17_r.bundle
- System/lib32/dbodbc17_r.bundle
- System/lib32/libdbicu17_r.dylib
- System/lib32/libdbicu17.dylib
- System/lib32/libdbcapi_r.dylib
- System/lib32/libdbcapi.dylib
- System/lib32/libdbicudt17.dylib
- System/lib32/libdbjodbc17.dylib
- System/lib32/libdbjodbc17.dylib
- System/lib32/libdbodbcansi17_r.dylib
- System/lib32/libdbodbc17_r.dylib
- System/lib32/libdbodbc17_n.dylib
- System/lib32/libdbodbc17.dylib
- System/lib32/libdblib17_r.dylib
- System/lib32/libdblib17.dylib
- System/lib32/libdbtool17_r.dylib
- System/lib32/libdbtasks17_r.dylib

- System/lib32/libdbtasks17.dylib
- System/lib32/libdbodm17.dylib
- System/lib32/libdbodbcinst17_r.dylib
- System/lib32/libdbjdbc17.jnilib
- System/lib32/libdbjodbc17.jnilib
- System/lib32/libdblib17_r.jnilib
- System/lib32/libodbcinst.dylib

SQL Anywhere enhancements

ALTER and CREATE ODATA PRODUCER statements

- The OData Producer's connection pool has been improved with the addition of connection authentication expiry. The producer configuration option `ConnectionAuthExpiry` specifies how often a pooled connection must recheck its authentication. The default frequency is 5 minutes and the maximum value 24 hours.
- There are restrictions on the characters that are supported for `/path-prefix` in the SERVICE ROOT clause.
- The USING clause supports the `AccessControlAllowOrigins` and `AccessControlAllowMethods` configuration parameters. These parameters allow you to configure the OData Producer to return the correct headers with Cross-origin Resource Sharing (CORS) requests. The producer responds to Origin and Access-Control-Request-Method HTTP request headers when the `AccessControlAllowedOrigins` configuration parameter is specified. The `AccessControlAllowedMethods` configuration parameter can be used to limit what HTTP methods are allowed to respond to CORS requests.

CREATE ODATA PRODUCER Statement ALTER ODATA PRODUCER Statement parameterization_level option

The default value for the `parameterization_level` option is now Off. Previously, the default value was Simple.
[parameterization_level Option](#)

Support removed for Red Hat Enterprise Linux 5

SQL Anywhere 17.0.10 is not supported on Red Hat Enterprise Linux 5.

Node.js support Support for Node.js versions 0.10, 0.12, and 4.x has been removed from the SQL Anywhere database server, the Node.js driver available for download from the Node Packaged Modules web site, as well as GitHub. The following files have been removed from the install:

- `jsextenv_v0_10.node`
- `jsextenv_v0_12.node`
- `jsextenv_v4.node`
- `sqlanywhere_xs_v0_10.node`
- `sqlanywhere_xs_v0_12.node`
- `sqlanywhere_xs_v4.node`

The Node.js driver now supports Node.js version 10. The deployment files for this driver are `sqlanywhere_xs_v10.node` and `jsextenv_v10.node`.

Offline reset password

The `-orp` option allows you to reset a forgotten DBA user's password. To use the `-orp` option, a user other than the DBA user must have been granted the `SYS_OFFLINE_RESET_PASSWORD_ROLE` system role.

i Note

This task is not required if your SQL Anywhere implementation has a fully deployed role-based access control model, where the DBA user has granted the `CHANGE PASSWORD` privilege to the correct power users. In a fully deployed RBAC security model, if the DBA forgets their password, then any power user with the `CHANGE PASSWORD` privilege can reset the DBA user's password. Therefore, if your implementation has a fully developed RBAC security model, you do not need to grant the `SYS_OFFLINE_RESET_PASSWORD` role to any users, and you do not need to use the `-orp` option.

[Resetting the DBA Password \(Command Line\) -orp Database Server Option System Roles](#)

PHP support

The PHP driver now supports PHP 7.2 and 7.3.

ROW constructor

The ROW constructor can be used without parameters to create an empty row without specifying NULLs and if the field in the row is an array, it is initialized to be a zero-length array. [ROW Constructor \[Composite\]](#)

sp_parse_json system procedure

The `maxlen` parameter has been renamed `maxdepth` since it represents the nesting depth of the JSON string, not its length. The `json` parameter has been renamed `jsonstring`. [sp_parse_json System Procedure](#)

-z database server option

The `-z` database server option displays diagnostic messages to the database server console for troubleshooting purposes. This now includes server subsystem shutdown log entries. The log entries take the form of `Shutting down sub-system-name at date-time`.

MobiLink enhancements

SYNCHRONIZE statement

The `KEY` clause lets you specify the database encryption key for a strongly encrypted database so that the transaction logs can be unencrypted. [SYNCHRONIZE Statement \[MobiLink\]](#)

Microsoft SQL Server 2017 support

MobiLink supports synchronization to Microsoft SQL Server 2017 consolidated databases.

32-bit MobiLink server has been removed

The 64-bit MobiLink server is still supported. If you are running a new install of 17.0.10, then the 32-bit MobiLink server files are not present.

If you are upgrading a 17.0.9 or earlier install on a 32-bit system, then you may still have some older MobiLink server files present. These files are not supported and can be removed:

- `mlsrv17.exe`
- `mlserv17.dll`
- `mlodbc17.dll`

- mlsq17.dll
- mljava17.dll
- mldnet17.dll
- dnetodbc17.dll

UltraLite enhancements

Checksum validation enabled by default Checksums are used to detect offline corruption on pages stored to disk, flash, or memory, which can help reduce the chances of other data being corrupted as the result of a bad critical page. Checksum validation is now enabled by default for newly initialized UltraLite databases. [UltraLite checksum_level Creation Option](#)

Windows Phone 8.1 support replaced with Universal Windows Platform Microsoft Windows Phone 8.1 support has been replaced with Microsoft Universal Windows Platform support on Windows 10 and Microsoft Windows 10 Mobile using Microsoft Visual Studio 2017. Files that were previously located in the `UltraLite\WinRT` folder are now located in the `UltraLite\UWP` folder.

Minimum supported Apple OS versions The minimum supported Apple OS versions are now macOS 10.9 and iOS 9.3.

1.5 Upgrading to SQL Anywhere 17.0 Support Package PL 29 Build 4793

SQL Anywhere 17.0 PL 29 Build 4793 introduces several enhancements.

To use some of these enhancements and changes, you must upgrade or rebuild your database. When you upgrade or rebuild a SQL Anywhere database to version 17.0 build 4793 or later, you cannot run the database on a database server earlier than version 17.0 build 4793.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

SQL Anywhere new features and behavior changes

Database Tools Library (DBTools) enhancement

The `DBCcreatedVersion` function now returns `VERSION_UNKNOWN` in the `created_version` field for a database store format that is newer than the version of `DBCcreatedVersion`. Previously, it returned the current version of the `dbtools` function library.

Return values for version 17 DBCreatedVersion include VERSION_UNKNOWN, VERSION_17, VERSION_16, and VERSION_12.

Return values for version 16 DBCreatedVersion include VERSION_UNKNOWN, VERSION_16, and VERSION_12. [DBCreatedVersion\(a_db_version_info *\) Method](#)

The DBChangeLogName function now checks if the database version is newer than it can handle. For example, if the DBChangeLogName version is 16 and the database version is 17, then DBChangeLogName issues an error. [DBChangeLogName\(a_change_log *\) Method](#)

Transaction Log utility (dblog) enhancement

The Transaction Log utility (dblog) now checks if the database version is newer than it can handle. For example, if the dblog version is 16 and the database version is 17, then dblog issues a message and stops. [Transaction Log Utility \(dblog\)](#)

Microsoft ADO.NET Data Provider enhancement

The unmanaged code portion of the Microsoft ADO.NET Data Provider is contained in a DLL that is unloaded by the provider into a directory and subsequently loaded from there into memory. In some situations, this action violates system security policies.

To accommodate this violation, the load procedure for the unmanaged code DLL (dbdata17.dll) has been changed as follows:

1. The provider looks for the dbdata DLL in the Microsoft .NET application's directory. If the DLL is found, then it is loaded and a version check is done. If the DLL version matches the Microsoft ADO.NET provider version, then the application is launched. Otherwise, the next step is performed.
2. The provider looks for the dbdata DLL in the Microsoft ADO.NET provider's directory (this directory could be different than the application directory). If the DLL is found, then it is loaded and a version check is done. If the DLL version matches the Microsoft ADO.NET provider version, then the application is launched. Otherwise, the next step is performed.
3. The provider looks for the dbdata DLL in the %TEMP% directory as described in the documentation. It starts with the directory at index 1 (for example, {16AA8FB8-4A98-4757-B7A5-OFF22COA6E33}_1708.x64_1). If the DLL is found, then it is loaded and a version check is done. If the DLL version matches the Microsoft ADO.NET provider version, then the application is launched. Otherwise, if the DLL was found but the version was wrong, an attempt is made to delete it. If the delete succeeds, then a new DLL is unpacked into the directory. Otherwise, the next directory (index 2, 3, and so on) is searched repeating step 3.

For step 2, no dbdata DLL will be found if the provider DLL is in the global assembly cache (GAC). Typically, the provider DLL is located with the application executable. Ultimately, your application decides how the provider is loaded if it is not through the GAC. Placement of the dbdata DLL as described in step 1 is preferable to that of step 2.

Step 3 is very similar to the previous behavior of the Microsoft ADO.NET Data Provider, except that the provider loads the DLL and performs a version check if the DLL is already present and then attempts to delete it if the version is wrong. Previously, the provider would attempt to delete the DLL first and, if the delete was not successful, load it and perform a version check. In most situations, this change should improve performance.

It is the responsibility of the Microsoft ADO.NET application developer to make a copy of the dbdata DLL during the development/test phase from the %TEMP% directory and embed the DLL in one of the directories described in step 1 or 2. The developer must ensure correct bitness (32/64 bit) and version match (for example, 17.0.8.4103) between the provider and the dbdata DLL for step 1 or 2 to work. [The SQL Anywhere .NET Data Provider Unmanaged Code](#)

ADO.NET SetupVSPackage enhancement

When the Microsoft ADO.NET Data Provider assembly installer, SetupVSPackage, modifies the `machine.config` files, it now indicates which config files are modified and which assemblies are referenced.

By default, SetupVSPackage uses the Location registry key, rather than current directory, to pick up the default location for assemblies. The `/salocation` (short form `/sal`) option can be used to override this default.

To help make it clear which assemblies are being referenced, SetupVSPackage indicates which assembly is referenced and which `machine.config` file is modified by displaying the full path for each file. [.NET Client Deployment](#)

zlib library upgraded to version 1.2.11

The version of zlib used by the database server for data compression has been upgraded to 1.2.11.

New Node.js drivers

Drivers are now provided for versions of Node.js up to and including version 9. [Node.js Application Programming XS JavaScript Application Programming](#)

MobiLink enhancements

Stack trace is written to messages log file

If a 64-bit MobiLink server crash occurs on Microsoft Windows or Linux, then a stack trace is written to the messages log file. It includes the crash address, module relative address, and module information (including base address, and offset). This feature is useful in the cloud where access to the log file is permitted, but not the file system in general. The following is an example of a stack trace on Linux:

```
E. 2017-12-16 10:51:06. Start of stack trace
E. 2017-12-16 10:51:06. 0x000000000047bbcd 0x0007bbcd in
mlsrv17[0x0000000000400000, 0x00000000]
E. 2017-12-16 10:51:06. 0x000000000047bd0e 0x0007bd0e in
mlsrv17[0x0000000000400000, 0x00000000]
E. 2017-12-16 10:51:06. 0x00007f9f820defbf 0x0000dfbf in libdbtasks17_r.so.
1[0x00007f9f820d1000, 0x00000000]
E. 2017-12-16 10:51:06. 0x00007f9f820df1e9 0x0000e1e9 in libdbtasks17_r.so.
1[0x00007f9f820d1000, 0x00000000]
E. 2017-12-16 10:51:06. 0x00007f9f81aef850 0x0000f850 in
libpthread-2.11.3.so[0x00007f9f81ae0000, 0x00000000]
E. 2017-12-16 10:51:06. 0x00007f9f81aef6eb 0x0000f6eb in
libpthread-2.11.3.so[0x00007f9f81ae0000, 0x00000000]
E. 2017-12-16 10:51:06. 0x0000000000434b9f 0x00034b9f in
mlsrv17[0x0000000000400000, 0x00000000]
E. 2017-12-16 10:51:06. 0x00007f9f81305c36 0x0001ec36 in
libc-2.11.3.so[0x00007f9f812e7000, 0x00000000]
E. 2017-12-16 10:51:06. 0x000000000040d169 0x0000d169 in
mlsrv17[0x0000000000400000, 0x00000000]
E. 2017-12-16 10:51:06. End of stack trace
```

The stack trace feature is also supported by 64-bit versions of the SQL Remote Message Agent utility (`dbremote`), the MobiLink SQL Anywhere client utility (`dbmlsync`), and the MobiLink Listener utility (`dblsn`).

This feature is not supported by the 32-bit versions of these applications.

zlib library upgraded to version 1.2.11

The version of zlib used by MobiLink for data compression has been upgraded to 1.2.11.

UltraLite enhancements

Minimum supported Apple OS versions

The following are now the minimum Apple OS versions supported by UltraLite:

- macOS version 10.9
- iOS version 9.3

zlib library upgraded to version 1.2.11

The version of zlib used by UltraLite for data compression has been upgraded to 1.2.11.

1.6 Upgrading to SQL Anywhere 17.0 Support Package PL 22 Build 4003

SQL Anywhere 17.0 PL22 Build 4003 introduces several enhancements.

To use some of these enhancements and changes, you must upgrade or rebuild your database. When you upgrade or rebuild a SQL Anywhere database to version 17.0 build 4003 or later, you cannot run the database on a database server earlier than version 17.0 build 4003.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

SQL Anywhere new features and behavior changes

New parameters for spatial methods that generate SVGs

The `ST_AsSVG`, `ST_AsSVGAggr`, `ST_AsXML`, and `ST_AsText` methods now accept the `MinViewBoxWidth` and `MinViewBoxHeight` format parameters. These parameters allow you to specify numeric values for the minimum viewBox width and height. [ST_AsSVG Method](#) [ST_AsSVGAggr Method](#) [ST_AsText Method](#) [ST_AsXML Method](#)

Unload utility (dbunload) enhancements

The new `-ru file` option has been added to support the unprocessed SQL statements file feature. If the unprocessed SQL statements file cannot be written to the current working directory or the directory specified by `-ru`, then it is written to the temporary files folder specified by either the `dbunload -dt` option or by one of the `SATMP`, `TMP`, `TMPDIR`, or `TEMP` environment variables. [Unload Utility \(dbunload\)](#)

Extraction utility (dbxtract) enhancements

The following options have been added:

- ae** do not stop after reload error
- dt** <dir> directory for temporary files
- kdi** <iters> number of key derivation iterations in new database (with -ek or -ep)
- ru** <file> path of unprocessed SQL file (default "unprocessed.sql")

If the unprocessed SQL statements file cannot be written to the current working directory or the directory specified by -ru, then it is written to the temporary files folder specified by either the dbxtract -dt option or by one of the SATMP, TMP, TMPDIR, and TEMP environment variables. [Extraction Utility \(dbxtract\)](#)

New connection parameter

The SQL Anywhere database server now supports server-side autocommit. In general, the use of server-side autocommit improves the performance of applications.

However, there are some 3rd-party frameworks like Hibernate that wrap SQL statement execution in (using JDBC as an example) setAutoCommit calls. This is equivalent to the following sample JDBC code sequence.

```
while( iterations-- > 0 )
{
    conn.setAutoCommit( true );
    stmt.execute( sql_statement );
    conn.setAutoCommit( false );
}
```

When connected to a 17.0 database server, such a construct results in suboptimal performance because each call to the JDBC setAutoCommit method sends a "SET TEMPORARY OPTION auto_commit='ON' (or 'OFF') to the database server for execution.

This problem has been fixed. A new connection parameter, ClientAutocommit=yes, can be used to cause the client JDBC- or ODBC-based application to revert to client-side autocommit behavior. Setting ClientAutocommit=no corresponds to the default behavior. The ClientAutocommit connection parameter can be used with version 17.0, 16.0, or 12.0.1 ODBC drivers but it has no effect if the database server does not support server-side commits (for example, 16.0 or 12.0.1 servers).

A workaround for better performance would be to move the setAutoCommit calls outside the loop. But in some third-party frameworks, this might not be possible.

```
conn.setAutoCommit( true );
while( iterations-- > 0 )
{
    stmt.execute( sql_statement );
}
conn.setAutoCommit( false );
```

On Microsoft Windows, the *Advanced* tab of the *ODBC Configuration for SQL Anywhere* dialog (using the ODBC Data Source Administrator) has been updated to include this new connection parameter.

New Node.js drivers

Drivers are now provided for versions of Node.js up to and including version 8. [Node.js Application Programming XS JavaScript Application Programming](#)

New PHP drivers

PHP drivers are now provided for PHP version 7. [SQL Anywhere PHP Extension The PHP External Environment](#)

Make a TLS connection without verifying the server's certificate

SQL Anywhere clients can make a TLS connection without verifying the server's certificate by specifying *none* for the `trusted_certificates` protocol option. The MobiLink server, the MobiLink client (dbmlsync), and UltraLite do not support `trusted_certificates="none"`.

Connecting without verifying the server's certificate is not recommended because it is less secure than verifying the certificate because the client can no longer protect against a man-in-the-middle attack. However, the connection is still strongly encrypted and prevents replay attacks, which is more secure than no encryption at all. With the *none* option, no trusted root certificate is required on the client. [trusted_certificate Protocol Option Encryption \(ENC\) Connection Parameter xp_startsmtp System Procedure CREATE PROCEDURE Statement \[Web Service\]](#)

Change in how the value for the CacheAllocated database server property is calculated

In previous releases, the `CacheAllocated` value included `MainHeapPages`. Now, the main heap no longer comes out of the cache, so the `CacheAllocated` property returns a value that may differ significantly from the value is returned in previous releases. [List of Database Server Properties](#)

MobiLink new features and behavior changes

Support for SAP HANA consolidated databases

Previous releases of MobiLink supported SAP HANA remote data sync to synchronize SQL Anywhere clients with an SAP HANA consolidated database. Now, use an SAP HANA 2.0 consolidated database for synchronization. [SAP HANA Consolidated Database](#)

Support for Microsoft SQL Server 2016 consolidated databases

The MobiLink server now supports Microsoft SQL Server 2016 consolidated databases. [Microsoft SQL Server and Microsoft Azure Consolidated Databases](#)

Support for Microsoft Azure consolidated databases

The MobiLink server now supports Microsoft Azure consolidated databases. The MobiLink plug-in for SQL Central does not support Microsoft Azure consolidated databases. [Microsoft SQL Server and Microsoft Azure Consolidated Databases](#)

New allow_unencrypted_basic_proxy_auth MobiLink client network protocol option

The new `allow_unencrypted_basic_proxy_auth` MobiLink client network protocol option permits basic HTTP proxy authentication. [allow_unencrypted_basic_proxy_auth MobiLink Client Network Protocol Option](#)

UltraLite new features and behavior changes

New sample for Universal Windows Platform (UWP)

A `CustDb` sample has been added for UltraLite for WinRT. The sample is a Visual Studio 2015 project that targets the Universal Windows Platform (UWP). The project is located in `%SQLANYSAMP17%\UltraLite.WinRT\Windows10UWP\CustDb`.

Bitcode support added for iOS

The UltraLite library for iOS now includes bitcode with arm64 and armv7 architectures.

1.7 Upgrading to SQL Anywhere 17.0 (build 2000)

As of build 2000, SQL Anywhere 17.0 contains several minor enhancements and behavior changes.

To use some of these enhancements and changes, you must upgrade or rebuild your database. When you upgrade or rebuild a SQL Anywhere database to version 17.0 build 2000 or later, you cannot run the database on a database server earlier than version 17.0 build 2000.

SQL Anywhere Server: General changes

Mac OS X 10.11 is now supported

SQL Anywhere now supports Mac OS X 10.11. When using Mac OS X 10.11, you may need to make the following adjustments to your application:

- If you are using APIs that require libdbcapi, then you must set either DYLD_LIBRARY_PATH or, if DYLD_LIBRARY_PATH does not work with system integrity protection enabled, then use SQLANY_API_DLL. [Installing DBD::SQLAnywhere on UNIX/Linux](#) [Installing sqlanydb on UNIX/Linux](#) [The PHP extension on UNIX/Linux](#) [Ruby programming XS](#) [JavaScript application programming](#)
- If you are using Java with native libraries, such as JDBC, then you must set either DYLD_LIBRARY_PATH or, if DYLD_LIBRARY_PATH does not work with system integrity protection enabled, use the -Djava.library.path=/path/to/sqlanywhere/lib64 command-line option. [How to load the SQL Anywhere JDBC driver](#)
- If you are using external functions, you must set either LD_LIBRARY_PATH or provide the full path to the shared library. [CREATE FUNCTION statement \[External call\]](#) [CREATE PROCEDURE statement \[External call\]](#)
- If you are using a custom install layout, then you may still need to set DYLD_LIBRARY_PATH or, if DYLD_LIBRARY_PATH does not work with system integrity protection enabled, then use SQLANY_API_DLL. Use install_name_tool to provide additional search paths rather than using DYLD_LIBRARY_PATH.

Statement performance summary update (database upgrade or rebuild required)

Use statement performance summary to identify and troubleshoot slow statements. The SYS_RUN_PROFILER_ROLE is required to view the gathered data. [Tip: Identify the cause of slow statements](#)

The following changes have been made to support this update:

New GTSYSPERFCACHESTMT system view The GTSYSPERFCACHESTMT system view contains SQL text for an expensive statement. [GTSYSPERFCACHESTMT system view](#)

New GTSYSPERFCACHEPLAN system view

The GTSYSPERFCACHEPLAN system view contains a graphical execution plan for expensive statements. [GTSYSPERFCACHEPLAN system view](#)

New `sp_top_k_statements` system procedure

The `sp_top_k_statements` system procedure identifies the specified number of statement/plan combinations with the longest runtime. [sp_top_k_statements system procedure](#)

New `sp_find_top_statements` system procedure

The `sp_find_top_statements` system procedure details performance statistics collected for each logged statement/plan procedure. [sp_find_top_statements system procedure](#)

Enhancement to the `-k` database server option

Specifying `-k` when starting the database server disables the statement performance summary for the Microsoft Windows Performance Monitor. [-k database server option](#)

Enhancement to the `CREATE TEMPORARY TRACE EVENT SESSION` statement

The new `WHERE` clause allows you to trace an event conditionally based on its properties. [CREATE TEMPORARY TRACE EVENT SESSION statement](#)

New `-pf` option

Specify the `-pf` option when using the SQL Anywhere database server executable (`dbsrv17`, `dbeng17`) or the Broadcast Repeater utility (`dbns17`) to write the process ID into the specified file. [-pf database server option Broadcast Repeater utility dbns17](#)

New ability to control lag between primary and mirror servers (database upgrade or rebuild required)

The new `lagtime` mirroring option specifies an approximate amount of time that a mirror server can lag behind the primary server in applying the transaction log. The primary server reduces its rate of transactions when the lag time approaches the value of this setting. [SET MIRROR OPTION statement Performance considerations with database mirroring systems](#)

New `MILLISECOND`, `MICROSECOND`, and `EXTRACT` functions (database upgrade or rebuild required)

Previously, you used the `DATEPART` function to obtain the millisecond and microsecond portions of a `TIMESTAMP` expression. Now, you can use the new `MILLISECOND` and `MICROSECOND` functions. Also, a new `EXTRACT` function returns the various date parts of a `TIMESTAMP` and `TIMESTAMP WITH TIMEZONE` expression. The `EXTRACT` function is in the ANSI/ISO SQL Standard. [EXTRACT function \[Date and time\] MILLISECOND Function \[Date and Time\]](#)

Performance counters now 64-bit (database upgrade or rebuild required)

Performance counters have been changed from 32-bit to 64-bit. [DB_PROPERTY Function \[System\] sa_db_properties System Procedure](#)

Enhancement to the Unload Utility (`dbunload`) to continue unloading after errors `-ac` option

Previously, if an error was encountered when rebuilding a database (the `dbunload -ac`, `-an`, or `-ar` options), then the rebuild operation would stop and the error had to be addressed before attempting the rebuild operation again. If there were many errors, then this was a time consuming process. The new `-ae` option allows you to request that the unload operation continue past any errors and through to completion, and then decide how to address the errors. If a statement in the `reload.sql` script fails, then the reload operation continues on to the next statement instead of stopping. Statements that fail are recorded in the `unprocessed.sql` script, and displayed in the command line output unless the `-q` option is specified. This enhancement allows you to see all of the errors at once and decide whether to fix all of the reported errors and rebuild the database from scratch, or apply the `unprocessed.sql` script on the unfinished database. [Unload Utility \(dbunload\)](#)

Enhancement to altering global temporary tables

Previously, if a global temporary table was referenced by a connection, then the equivalent of a lock was placed on the table until the connection closed, regardless of whether or not the lock was needed. ALTER statements by other connections on the global temporary table were prohibited and would fail until the connection that started the lock ended. Now, an explicit schema lock is placed on the table when a connection initially references it but is only retained if the connection references it in cursor, or if the table has rows. If the table is not referenced by a cursor and does not have any rows, then the schema lock is dropped during a commit or rollback, and other connections can execute an ALTER statements on the table. Use the `sa_locks` system procedure to determine which global temporary tables have schema locks on them. [sa_locks system procedure](#)

Enhancement to `xp_startsmtp` system procedure

Previously when starting an email session, the database server only supported PLAIN authentication of user IDs and passwords with SMTP servers. Now, the database server supports CRAM-MD5 authentication, as well as PLAIN authentication.

When you use the `xp_startsmtp` system procedure with the `smtp_auth_username` and `smtp_auth_password` parameters, the database server uses CRAM-MD5 authentication. If the SMTP server does not support CRAM-MD5 authentication, then the database server uses PLAIN authentication. If you specify the `-fips` database server option, then only PLAIN authentication is used. The `-fips` database server option does not support CRAM_MD5 authentication.

[xp_startsmtp system procedure -fips database server option](#)

New `sp_disk_info` system procedure Use this system procedure to retrieve disk drive information for a given path. [sp_disk_info system procedure](#)

Execute CREATE INDEX and LOAD TABLE statements in parallel by using the new BEGIN PARALLEL WORK statement

The BEGIN PARALLEL WORK statement can improve performance by executing a list of CREATE INDEX or a list of LOAD TABLE statements in parallel. The number of available logical processors on the computer that the database server runs on, as well as the settings of the `-gtc` option and the `max_parallel_statements` option limits the number of statements that can execute at the same time.

The following statements, options, and utilities support this update.

- BEGIN PARALLEL WORK statement [BEGIN PARALLEL WORK statement](#)
- CREATE INDEX statement [CREATE INDEX statement](#)
- LOAD TABLE statement [LOAD TABLE statement](#)
- The new `max_parallel_statements` option limits the number of statements inside a BEGIN PARALLEL WORK statement that can execute at the same time. [max_parallel_statements option](#)
- The Unload utility (`dbunload`) supports a new `-bp` option to create `reload.sql` files that are optimized for the parallel execution of CREATE INDEX and LOAD TABLE statements. When the `-bp` option is specified, CREATE INDEX and LOAD TABLE statements are listed inside BEGIN PARALLEL WORK statements in the `reload.sql` file. [Unload utility \(dbunload\)](#)

Use the LOAD TABLE statement with specific views

You can now use the LOAD TABLE statement to load content into a regular view. The view must be based on a base table and its definition must be the following:

```
CREATE VIEW
[owner.]view-name
AS SELECT * FROM base-table-name
```

To support this update, the GRANT statement now allows you to grant the LOAD privilege to a view. [LOAD TABLE statement](#) [GRANT statement](#)

CREATE TABLE statement and CREATE EXISTING TABLE statement AT location string now accepts an ESCAPE CHARACTER clause

In the rare case where one of the delimiters within the location clause must be interpreted literally, you can specify an escape character that is used to escape the location delimiter. In these cases, this column contains the escape character that was specified during the proxy table creation.

[CREATE TABLE statement](#) [CREATE EXISTING TABLE statement.](#)

SQL Anywhere Server: Security changes

New ACCESS DISK INFORMATION system privilege

A new system privilege, ACCESS DISK INFORMATION, was added to allow a user to access information regarding the total disk size and the remaining disk space by using the sp_disk_info system procedure. This system privilege is included in the user-defined COCKPIT_ROLE role. It is required to run the new sp_disk_info system procedure. [System privileges](#)

Increased security for exchanging passwords between clients and database servers

When connecting to the database server, clients now use RSA encryption to more securely encrypt the user password before they send it to the database server. This update does not apply to TDS connections. For applications that are not secured using TLS encryption and when using the GRANT CONNECT, CREATE USER, or ALTER USER statements to create or alter a user, the password in the statement is not encrypted. The software version of both the client and the database server must be at least version 17.0, build 2000.

Strong encryption keys are now created using Password-Based Key Derivation Function #2 (PBKDF2), part of the PKCS#5 standard (database upgrade or rebuild required)

When encrypting a database or enabling table encryption in a database, the software now uses Password-Based Key Derivation Function #2 (PBKDF2), which is part of the PKCS#5 standard. To protect encryption keys from brute-force attacks, the software repeatedly applies a cryptographic hash to the encryption key. By default, 2000 iterations are applied. You can change the number of iterations that are applied when you create the encrypted database.

The following features were changed to support this update:

- New -kdi option for the Initialization utility (dbinit) and Unload utility (dbunload) . [Unload utility \(dbunload\)](#) [Initialization utility \(dbinit\)](#).
- New -li option for the Initialization utility (dbinit). [Initialization utility \(dbinit\)](#)
- KEY DERIVATION ITERATIONS clause of the CREATE ENCRYPTED DATABASE and CREATE ENCRYPTED FILE statements. [CREATE ENCRYPTED DATABASE statement](#) [CREATE ENCRYPTED FILE statement](#)
- KeyDerivationIterations database property. [List of database properties](#)

SQL Anywhere clients and database servers support client-side certificates

Previously, only server-side certificates, which assure clients that they are connecting to a trusted database server, were supported. Now client-side certificates are also supported, ensuring that clients only connect to a database server if they are using a certificate that is signed by a source that the database server trusts.

[Using client-side certificates for TLS Encryption \(ENC\) connection parameter -ec database server option identity protocol option identity_password protocol option trusted_certificates protocol option](#)

New allow_expired_certs encryption option for TLS and HTTPS

Setting this option on the client allows the client libraries to trust a root certificate and database server certificate that have either expired or are not yet valid. Setting this option on the database server allows the database server to accept an identity file that has either expired or is not yet valid. [Encryption \(ENC\) connection parameter -ec database server option -xs database server option Network protocol options CREATE PROCEDURE statement \[Web service\] xp_startsmtp system procedure.](#)

Update to client file security

A new connection parameter, ClientFileValidator, allows you to specify when the database server reads local files. If the database server determines that the request is directly from the client, or if a callback returns TRUE, then the file is transferred.

In previous releases, there was a validator in place for some clients that was called when a file request came from an unverified source. This update changes this behavior. To use the previous behavior, set the ClientFileValidator connection parameter to TrustedServer on the client. [ClientFileValidator \(CFV\) connection parameter LOAD TABLE statement](#)

SQL Anywhere Server: Catalog changes

New columns added to the SYSPROCEDURE system view and the SYSVIEWS compatibility view (database upgrade or rebuild required)

Six columns have been added to the SYSPROCEDURE system view:

- dialect
- is_deterministic
- is_external
- external_language
- external_name
- sql_security

Additionally, the SYSVIEW compatibility view has a new check_option column.

[SYSPROCEDURE system view](#) [SYSVIEW system view](#)

SQL Anywhere Server: Web service changes

Web service procedures return multiple headers with the same name (database upgrade or rebuild required)

Previously, web service procedures only returned the first instance of a header value if the HTTP server returned multiple headers with the same name. Now, the HTTP web procedure result set allows multiple headers with the same name by including a third column in the result set that indicates the instance of the header. Web service procedures that specify multiple headers with the same name in the HEADERS clause now send all of those headers to the web server. Upgrade your database to version 17 SP1 to access this update.

A new database property, `HttpClientMultipleHeaders`, indicates whether the database supports multiple headers with the same name.

[Result set retrieval from a web service](#) [Quick start to using the database server as a web client](#) [List of database properties](#) [Lesson 2: Sending requests from a web client and receiving responses](#)

Web service procedures allow finer control over status codes

The EXCEPTIONS option of the SET clause allows you to control status code handling. [CREATE PROCEDURE statement \[Web service\]](#) [CREATE FUNCTION statement \[Web service\]](#)

Support for HTTP/HTTPS connection queuing

Previously, when the database reached its HTTP/HTTPS connection limit, additional attempts to connect to the database were rejected.

Now, when a database server reaches its HTTP/HTTPS connection limit, additional connections attempts are queued. When a connection becomes available, the first connection in the queue is processed.

Standard connections are not queued. When a database server reaches its connection limit, any new standard connection attempt fails.

The personal database server reserves a minimum of 3 connections for standard connections.

The network database server does not reserve connections for standard connections. Reserve standard connections by using the `reserved_connections` database option.

The following properties and options were added to support this update:

- `HttpConnectionsQueued`, `HttpQueueCount`, `HttpQueueMaxCount`, and `HttpQueueTimedOut` database server properties [List of database server properties](#)
- `reserved_connections` option. [reserved_connections option](#)

[Database and database server connection limits](#)

SQL Anywhere Server: Programming changes

Embedded SQL: Support for wide merge

Embedded SQL now supports wide merges by using the ARRAY clause of the EXECUTE statement [ESQL]. [EXECUTE statement \[ESQL\]](#) [Wide merges using Embedded SQL](#)

ODBC: Support for wide merge

ODBC now supports wide merges using `SQLSetStmtAttr (SQL_ATTR_PARAMSET_SIZE)` and `SQLBindParameter`. [Executing statements with bound parameters](#)

JDBC: Support for batched merge

JDBC now supports wide or batched merges using `PreparedStatement.addBatch()` and `PreparedStatement.executeBatch()`. [JDBC batch methods](#)

SQL Anywhere C API: Support added for wide merges

Wide merge is now supported in version 4 of the SQL Anywhere C API.

.NET 4.0 no longer supported

The .NET 4.0 version of the SQL Anywhere .NET Data Provider has been removed. Microsoft no longer provides security updates, technical support or hotfixes for .NET 4.0. Later versions of .NET 4 continue to be supported by the SQL Anywhere .NET Data Provider.

New versions of Node.js supported

Drivers are provided for versions of Node.js such as 0.10, 0.12, 4.x.y and 5.x.y. [Node.js application programming XS JavaScript application programming](#)

DataNucleus and OpenJPA support

Adapters have been implemented for DataNucleus and OpenJPA.

An adapter for SQL Anywhere has been added to DataNucleus. See <http://www.datanucleus.org/products/datanucleus/datastores/rdbms.html> ↗

An adapter for SQL Anywhere has been written for OpenJPA (adoption by the OpenJPA project is pending). <https://github.com/sqlanywhere/OpenJPA> ↗

Upgraded Jetty support The OData server now uses Jetty 9.3.7.

Drivers are provided for versions of Node.js such as 0.10, 0.12, 4.x.y and 5.x.y. [How to set up an OData server](#)

SQL Anywhere Server: Behavior changes

Minimum communication packet size is now 1000

The minimum size of communication packets allowed by client connections and database servers is now 1000 bytes. Previously 500 bytes was the minimum.

You do not need to upgrade clients to use this update. Older clients with minimum packet sizes less than 1000 that connect to a version 17 SP1 database server use the new minimum value. However, clients running version 17 SP1 software cannot connect to database servers that are running earlier versions of the software and are using a default packet size of less than 1000 bytes.

[CommBufferSize \(CBSIZE\) connection parameter](#) and [-p database server option](#)

Spatial geometries can now be created outside of the spatial reference system boundaries

Previously, geometries could only exceed SRS boundaries by fifty percent in each direction, after which an error was raised. The same error was also raised on round-earth SRS if the geometry points exceeded the allowable range. Geometries can now exceed the SRS boundaries as long as all points can be represented in the SRS coordinate system and the geometry remains trivially valid. Geometries that exceed SRS boundaries by more than fifty percent are not indexable.

A new ST_Geometry method, ST_IsIndexable, detects geometries that are not indexable. See [ST_IsIndexable method List of spatial predicates Indexes on spatial columns](#)

Administration tool changes

DLL dependencies Both 32-bit and 64-bit administration tools like Interactive SQL and SQL Central (the Java-based tools) now require `msvcr100.dll`.

Interactive SQL enhancements

-c and -f options for Interactive SQL utility (dbisql) can be used together Previously, if the -f option was specified, then the -c option was ignored. Now both options can be used together.

Improved reporting of errors in multi-line SQL statements

When you execute multi-line SQL statements from a file and one of those statements fails, Interactive SQL reports the file name and line number where the error occurs.

Connect and Disconnect buttons added to the Toolbar

Connect and disconnect from databases and servers by using the toolbar menu items in Interactive SQL.

Interactive SQL option to treat and display SQL statement warnings as errors.

Specify the `-we` option to have Interactive SQL treat and display warnings in executed SQL statements as errors. When the `-q` option is specified Interactive SQL suppresses warnings except when the `-we` option is also specified. [Interactive SQL utility \(dbisql\)](#)

Cockpit enhancements

The following changes were made to the Cockpit:

Ability to view historical event information Previously, you could only view historical information about an event while the event was active. Now you can view the historical information for an event regardless of the event's current status.

Simplified connection counts

Previously, all connections to the database server, including those made by the Cockpit itself, were included in the connections counts for the database server. Now, connections made by the Cockpit are excluded from the connection counts.

Email notification of Cockpit alerts

Set up the Cockpit to send email notifications of alerts. Specify which type of alert requires an email notification and set the frequency at which the emails are sent. [Enabling the Cockpit to Send Alert Emails](#).

New alerts for the Cockpit

When there is low disk space on a drive that contains a database file, such as the database file, `dbspace`, or transaction log, an alert occurs.

When there is a high percentage of database page look ups that are not satisfied by finding the page in the cache, an alert occurs.

New privilege added to the COCKPIT_ROLE user-defined role

Now the `COCKPIT_ROLE` user-defined role includes the `ACCESS DISK INFORMATION` system privilege. To access the full functionality of the Cockpit, the `ACCESS DISK INFORMATION` system privilege is required. [Upgrading Version 17 Databases to Use the SQL Anywhere Cockpit \[page 146\]](#).

Convert your Cockpit database to a new database

When the Cockpit is running with a temporary database, you can convert it to a permanent database without losing your settings and alert history.

The `cockpitdb` option for the `sa_server_option` system procedure, the `-cdb` database server option, and the `CockpitDB` database server property were updated to support this update. [sa_server_option system procedure -cdb database server option List of database server properties](#)

SQL Anywhere Profiler enhancements

Automatic analysis of your workload

The Profiler automatically analyzes your workload and highlights areas you can focus on improve performance. This report addresses questions about whether performance is limited by:

- blocking
- database server hardware
- server workers
- backups, checkpoints, or other internal database server activity

[SQL Anywhere Profiler](#)

New profiling option to target the information collected during a profiling session.

Previously, you could only run comprehensive profiling, which collected information about all activity occurring in your database. Now you can run targeted profiling, which reduces the amount of information collected to database server performance statistics and SQL statements which fulfill specified criteria. Targeted profiling allows you to profile databases, such as those in production environments, for extended periods of time. [Running a Targeted Profiling Session \(Profiler\)](#)

Improved Server Load panel

The *Server Load* panel includes two new values, *Unscheduled requests* and *Cache Size*. The *Server Load* panel also includes presets for choosing the values to graph, and options for configuring the graph. [Troubleshooting: Detect When Hardware Resources Affect Performance by Using the Workload Summary and Server Load Graph \(Profiler\)](#)







Send SAP Support diagnostic information about your database server

When requested by Support, use the Profiler to take a snapshot of your database server's diagnostic information, save it to a file, and then email it to support. You can also use the Profiler to view your database server's statement performance summary. [Sending SAP Support a Snapshot of Your Database Server'S Diagnostics \(Profiler\)](#)

Improved functionality for editing filter expressions

When you edit a filter expression in the *Edit Filter Expression* window, you now have access to and can use all clauses in the filter syntax grammar. As well, the user interface now includes embedded help with examples.

Start the Profiler from the Start menu or from the command line

Start the Profiler by clicking  *Start*  *Programs*  *SQL Anywhere 17*  *Administration Tools*  *SQL Anywhere Profiler* . You can also start the Profiler by using the Profiler utility (dbprof). [Profiler Utility \(dbprof\)](#)

MobiLink changes

New allow_expired_certs MobiLink client network protocol option

Use the allow_expired_certs protocol option to accept a server certificate that has either expired or is not yet valid and continue with the synchronization.

New MobiLink utility to help in the development of large-scale synchronization systems

The MobiLink template utility (mltemplate) is a productivity tool for advanced MobiLink developers, which provides assistance in creating MobiLink synchronization scripts and other objects that accompany them. [Advanced feature: MobiLink Template utility](#)

Use the new MLTEMPLATE_JARS environment variable with this utility. [MLTEMPLATE_JARS environment variable](#)

MobiLink behavior changes

New default values for buffer_size MobiLink client network protocol option

The default values for the buffer_size client network protocol option have changed. The default value is now 256 KB (up from 64 KB), and the Android, iOS, Linux ARM, Microsoft Windows Phone/Store default value has been increased to 64 KB from 16 KB. The Microsoft Windows Mobile default value remains unchanged at 16 KB. [buffer_size MobiLink client network protocol option](#)

New Fetch array size option for Oracle consolidated databases

Use the Fetch array size option to specify the number of rows fetched from an Oracle consolidated database. Increasing the number of rows can improve performance by reducing the number of round trips to fetch data. [SQL Anywhere 17 - Oracle ODBC driver](#)

MobiLink server executable change: mlsrv17.dll (Microsoft Windows) and libmlserv17_r.so (UNIX/Linux)

The MobiLink server has been split into two executables:

Microsoft Windows	mlsrv17.exe
	mlserv17.dll
<hr/>	
UNIX and Linux	mlsrv17
	libmlserv17_r.so
<hr/>	

[Microsoft Windows 32-bit applications](#) [Microsoft Windows 64-bit applications](#) [64-bit applications on UNIX and Linux](#)

UltraLite changes

UltraLite concurrency

When an UltraLite application opens multiple databases, requests on separate databases now run concurrently. [UltraLite concurrency](#)

New UploadUnknown database property

This property returns a values that indicates whether the most recent synchronization failed after the upload was sent but before the upload acknowledgement was received from the synchronization server. [UltraLite database properties](#)

New methods added to the UltraLite C++ API

Three new methods have been added to the UltraLite C++ API that allow you to get or set column or parameter values through a byte array: ULResultSet::GetBytes, ULResultSet::SetBytes, and ULPreparedStatement::SetParameterBytes.

WinRT API changes

The UltraLite for WinRT API is now supported on Microsoft Windows 10.

UltraLiteJ changes

UltraLiteJ is now supported on Android x86.

As well, UltraLiteJ for Android now supports data synchronization over TCP/IP, including with TLS. Two new classes support this feature: StreamTCPIPParams and StreamTLSParams. These classes set stream parameters for a TCP/IP and TLS synchronization respectively. There are two new constants in class SyncParams: TCPIP_STREAM and TLS_STREAM. These constants create a SyncParams object by the method Connection.createSyncParams.

Documentation changes

New tutorials

A new tutorial has been added for creating an LDAP user authentication environment. [Tutorial: Creating an LDAP user authentication environment](#)

1.8 What's New in SQL Anywhere Version 17.0 (Product-Wide)

SQL Anywhere 17.0 introduces several product-wide new and changed features to learn about, in addition to component-level changes.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

New Product-Wide Features

New HTML-5, database server monitoring tool: SQL Anywhere Cockpit

The Cockpit is a database server monitoring tool that provides an up-to-date view of the availability, capacity, and performance of your database server. It runs in a browser and can provide information to the SAP DB Control Center. See [SQL Anywhere Cockpit](#).

Use the Cockpit to track who is logged on to a database server; display both database server and client statistics, and disconnect users from databases.

The Cockpit runs on the same database server, which you want to monitor. It is light-weight and is designed to run continuously in the background so that it is always available. But, you can also easily start and stop it to investigate issues.

The Cockpit supports version 17 and version 16 databases running on a version 17 database server. As described below database users must have the COCKPIT_ROLE user-defined role to connect to the Cockpit.

The following changes have been made to support the SQL Anywhere Cockpit:

New database server option to start the SQL Anywhere Cockpit on your database server When starting your database server specify the `-cdb` database server option to start the SQL Anywhere Cockpit. See [-cdb database server option](#).

New option for the `sa_server_option` to control the SQL Anywhere Cockpit on a running database server

Use `sa_server_option` system procedure and the Cockpit parameter to start and stop the Cockpit on a running database server. See [sa_server_option system procedure](#).

New COCKPIT_ROLE user-defined role required to use the Cockpit

To connect to the Cockpit, users must have exercise rights to the COCKPIT_ROLE user-defined role. Newly created databases include this user-defined role, but by default no user is granted the exercise rights to this role. To use the Cockpit with version 16 databases running on a version 17 database server, add the COCKPIT_ROLE to the database, and then grant users the exercise right to this role.

For convenience in the sample database the DBA user is granted exercise rights to this role.

By default, the COCKPIT_ROLE user-defined role includes all the system privileges required to perform actions within the Cockpit. See [COCKPIT_ROLE user-defined role](#).

Secure feature for managing the Cockpit

Control whether users can start or stop the Cockpit, with the `-sf` database server option. This feature controls whether users can run the `sa_server_option` system procedure with the CockpitDB option. See [-sf database server option](#) and [Cockpit Security](#).

New database server properties to support the Cockpit

The CockpitURL database server property returns the URL to access your Cockpit. The CockpitDB database server property returns the set of Cockpit options currently being used by the database server. See [List of database server properties](#).

Removal of dbconsole

The Cockpit replaces the graphical administration tool dbconsole, which has been removed from the software.

SQL Anywhere Profiler, a new database development and troubleshooting tool

The Profiler is a development tool for tuning and troubleshooting your database applications. Use it to quickly view SQL statements that are currently running in your database, compare run times of stored procedures, and analyze how the objects in your database interact with each other. The Profiler supports version 17 databases.

The Profiler replaces most of the application profiling administration tools in SQL Central.

The following changes have been made to support the Profiler.

- The Application Profiler and the SQL Anywhere Performance Monitor in SQL Central have been removed from the software and have been replaced with the Profiler. If you open Profiler and connect

to a pre-version 17 database, the Application Profiling tools are not available. Similarly, if you open a database server, the [Performance Monitor](#) tab has been removed.

If you want to use these features with a pre-version 17 database, use the version of SQL Central that matches the version of your database.

- The database tracing feature is deprecated. The diagnostic tracing levels `optimization_logging` and `optimization_logging_with_plans` are ignored by the software. The [Database Tracing Wizard](#) in the SQL Anywhere plug-in for SQL Central has been removed. The new Profiler contains the ability to record and view database operations.
- The [Deadlocks](#) tab and [Auditing](#) tabs in the SQL Anywhere plug-in for SQL Central have been removed. The new Profiler contains a [Blocking](#) tab that shows both blocking and deadlock information. You can still generate auditing information, use the `CREATE TEMPORARY TRACE EVENT SESSION` statement to generate an ETD file, and then view the file using the `dbmanageetd` utility.
- New `SYS_RUN_PROFILER_ROLE` system role is required to connect to the Profiler. Newly created databases include the user-defined role `SYS_RUN_PROFILER_ROLE` system role. This role is required to connect to the Profiler. By default no user is granted the exercise rights to this role. For convenience in the sample database the DBA user is granted exercise rights to this role. See [System roles](#).

Product-Wide Behavior Changes and Deprecated or Removed Features

Removal of default DBA/sql user, and change to default minimum password length

Previously, when you created a new database, if you did not specify a DBA user for the database, a user was automatically created with user ID DBA and password sql. Now, the database server requires you to supply a user ID and password. In addition, the minimum password length for users has changed from 3 to 6, but you can override this length at database creation time.

These changes are designed to improve security in new databases. If you have databases with a DBA/sql user ID, then change it because this is the user login that a malicious user is likely to try when attempting to access a database.

CREATE DATABASE statement

The DBA USER and DBA PASSWORD clauses, used for specifying the user ID and password for initial database DBA user, are now required when creating a database. Also, a new optional clause, MINIMUM PASSWORD LENGTH, has been added to allow you to override the (new) default minimum password length of 6. See [CREATE DATABASE statement](#)

Initialization utility (dbinit)

The `-dba` option, used for specifying the user ID and password for initial database DBA, is now required when creating a database. Also, a new option, `-mpl`, has been added to optionally allow you to override the (new) default password length of 6. For example, `-mpl 3` sets the minimum password length to 3. See [Initialization utility \(dbinit\)](#).

min_password_length database option

The default length for the `min_password_length` database option is now 6 instead of 3. See [min_password_length option](#).

-su database server option and the utility database (utility_db)

The `-su` database server option, used to set the password for the utility database (`utility_db`), now accepts the option of specifying `password; user-ID, password;` or, alternatively, `none`. The password

must be 6 characters or longer. This restriction is true even when using the personal database server (dbeng17.). See [-su database server option](#) and [The utility database \(utility_db\)](#).

util_db.ini file

The support for the util_db.ini file, deprecated in version 10, has been removed. Use the -su option to set the password for the utility database (utility_db). See [-su database server option](#).

Sample database

The sample database (demo.db), used extensively throughout the documentation examples and tutorials, as well as by many of the software samples, continues to have a DBA/sql user to allow continued compatibility. This is true for any database created using the `newdemo.bat` script (`newdemo.sh` on Linux). As well, databases created using `newdemo.bat` have a minimum password length of 3 instead of 6. See [The sample database](#).

SQL Anywhere 17 Demo data source

Previously, when using the SQL Anywhere Demo ODBC data source to connect to the [demo.db](#) database in the administration tools, utilities, or in SQL statements, you did not need to specify the password (`sql`) because the password was stored as a setting in the ODBC data source. Now the database requires the password `sql` when you use the [SQL Anywhere 17 Demo](#) ODBC data source.

SQL Anywhere 17 CustDB data source

Previously, when using the SQL Anywhere CustDB ODBC data source to connect to the [demo.db](#) database in the administration tools, utilities, or in SQL statements, you did not need to specify the MobiLink user ID (`ml_server`) and password (`sql`) because the user ID and password were stored as a setting in the ODBC data source. Now the database requires the user ID `ml_server` and password `sql` when you use the [SQL Anywhere 17 CustDB](#) ODBC data source.

Documentation examples and tutorials

Any SQL, command line, or programming examples in the documentation that show connections to, or creation of, SQL Anywhere databases other than the sample database specify a user ID and password for a DBA user. Previously, these examples either did not specify a DBA user, or gave DBA/sql for the user ID and password. In the latter case, the examples have been changed to use DBA/passwd for the DBA user ID and password, to satisfy the new default password length requirement of 6.

All other documentation examples that connect to the sample database, or any database created using the `newdemo` script (for example, the database used in the application profiling tutorials) remain the same, since these databases continue to have the DBA/sql user ID and password.

This includes tutorials and examples in products like MobiLink and UltraLite, where connections to, or creation of, SQL Anywhere databases take place.

MobiLink, UltraLite, Relay Server, and SQL Remote products

There are no changes to these products as a result of changing the default password length, or eliminating the default DBA/sql user ID and password when initializing a database.

Most SQL Anywhere and MobiLink samples and sample databases included with the software avoid embedded passwords

Whenever possible, the samples included with the software are designed to demonstrate best practices regarding the handling of database passwords. For security reasons, you should never include passwords within your application code. Therefore most samples avoid embedded passwords. When a password is necessary to build or run a sample, you are required to provide the database password.

For example, many samples use a demonstration database (demo.db) that is included in the root of the samples directory. For this database, the database administrator user ID is *DBA* and the password is *sql* (note that passwords are case sensitive).

In some sample Microsoft Visual Studio projects, database credentials are still included in configuration files to support Microsoft Visual Studio design-time features which require the ability to connect to the sample database in the background. Good practice dictates that this sensitive configuration information be removed from the application project once software development has been completed.

SSL version 3.0 no longer supported

SSL version 3.0 is no longer supported. Use TLS version 1.0 or later instead.

Strong encryption now achieved using OpenSSL

i Note

These changes were released in version 12 and version 16 Support Packages.

SQL Anywhere version 16.0 up to build 1695 and version 12.0.1 up to build 3986 included a Certicom encryption module that provided strong encryption used throughout the software. Now, SQL Anywhere includes an OpenSSL encryption module for the strong encryption. The Certicom encryption module has been removed.

Read the following descriptions to determine how you may be impacted by this change.

FIPS-certified encryption now requires the private key of an identity file to be encrypted using AES

- OpenSSL supports FIPS-certified AES encryption for the private key of an identity file. New servers using the FIPS-certified OpenSSL encryption module will not start when using an identity file that has its private key encrypted with 3DES. You must re-encrypt the identity file using AES. To do this, run a command similar to the following using an upgraded viewcert utility:

```
viewcert -p -o new-file-name -op new-password -ip old-password old-file-name
```

The new and old passwords can be the same. See [Certificate Viewer utility \(viewcert\)](#).

- The sample server identity file (*rsaserver.id*) and client identity file (*rsaclient.id*) have been modified so that the private keys are encrypted using AES rather than 3DES.
- Trusted root certificate files specified using *trusted_certificates* do not need to be modified.
- Versions of the server that use the Certicom encryption module do not start when using an identity file that has its private key encrypted using AES.
- See also [FIPS-certified encryption technology](#).

TLS handshake

With Certicom, a client could trust any certificate in the signing chain and the TLS handshake would succeed. With OpenSSL, the client must trust the root certificate in the chain.

Self-signed certificates must now have the Certificate Signing attribute set

Self-signed certificates must now have the Certificate Signing attribute set when using the identity encryption option (for example, the *-x mlsrv* and *-xs dbsrv* options). To determine if a certificate the Certificate Signing attribute is set, use the viewcert utility and look for the Certificate Signing attribute in the Key Usage portion of the output. If your self-signed certificates do not have the Certificate Signing attribute set, then regenerate the certificates.

Create Certificate utility (createcert) now uses AES encryption instead of 3DES

The Create Certificate utility (createcert) now uses AES rather than 3DES encryption for encrypting the private key in the server identity file.

A new option, `-3des`, has been added to the Create Certificate utility. This option creates a 3DES-encrypted server identity file that can be used by database servers in previous releases. New servers using FIPS-certified encryption cannot start using 3DES-encrypted certificates; however, if you are not using FIPS-certified encryption, then you can use 3DES-encrypted certificates. See [Certificate Creation utility \(createcert\)](#).

View Certificate utility (viewcert) now uses AES encryption instead of 3DES

The View Certificate utility (viewcert) now uses AES rather than 3DES encryption when you specify the `-p` option to PEM-encode the output and when you specify the `-ip` and `-op` options to set the password.

The View Certificate utility supports a new option, `-3des`, that encrypts output and passwords using 3DES instead of AES. See [Certificate Viewer utility \(viewcert\)](#).

Database server now loads the FIPS-certified encryption libraries at startup on 32-bit Windows

Previously, the 32-bit database server loaded the FIPS-certified encryption libraries on Windows only when needed. Now, on 32-bit Windows, the server always attempts to load these libraries at startup. If loading fails, then no error is returned unless an attempt is made to use FIPS-certified encryption. On 64-bit Windows, the 32-bit database server no longer supports FIPS-certified encryption. Use the 64-bit database server instead. FIPS-certified encryption is separately licensed.

Deploying FIPS-certified encryption

If you are deploying FIPS-certified encryption, then there are new shared libraries to deploy; these files are included in your software. The former files, `sbgse2.dll` and `libsbgse2.so`, are no longer installed by the software. The new files to deploy are:

- 64-bit Windows: `libeay32.dll`, `ssleay32.dll`, and `msvcr100.dll`
- 32-bit Windows: `libeay32.dll`, `ssleay32.dll`, and `msvcr90.dll`
- Linux: `libcrypto.so` and `libssl.so`

Starting with 17.0.6 build 2844 and 17.0.7 build 3381, `libcrypto.so` is renamed to `libdbcrypto.so` and `libssl.so` is renamed to `libdbssl.so`.

Note

On Windows, although 32-bit and 64-bit FIPS-certified OpenSSL libraries for encryption are provided, use the 64-bit libraries on a 64-bit system.

See [Files required for Embedded SQL clients](#).

MobiLink-related changes and information

Connecting to a MobiLink server using client-side certificates now requires the Digital Signature certificate attribute to be set

TLS connections to a MobiLink server using client-side certificates now require the client-side certificate to have the Digital Signature attribute set. If the attribute is not set, then the connection will fail.

To determine if a certificate has the Digital Signature attribute set, use the View Certificate utility (viewcert) and look for the Digital Signature attribute in the Key Usage portion of the output. If

your client-side certificates do not have the Digital Signature attribute set, then you must regenerate the certificates. See [Certificate Viewer utility \(viewcert\)](#).

FIPS-certified end-to-end encryption now requires the private key to be encrypted using AES

If the private key file provided to a MobiLink server by the `e2ee_private_key` file option of the `-x` command-line option is encoded using 3DES and the server is using FIPS-certified encryption, then the private key file needs to be regenerated with the private key encrypted using AES.

How to update a MobiLink deployment that uses non-FIPS-certified TLS (includes HTTPS) and client-side certificates

1. If your client-side identity certificates do not have the Digital Signature attribute set and the client connects directly to the MobiLink server, then you must regenerate and deploy client-side certificates with the Digital Signature attribute set.
2. Update the server-side binaries.
3. Update the client-side binaries.

How to update a MobiLink deployment that uses FIPS-certified TLS (includes HTTPS) and client-side certificates

These steps update the client identity certificates twice if the Digital Signature attribute is missing from client-side identity certificates. This procedure can make the update less disruptive because synchronizations can continue without having to coordinate the client-side and server-side updates to occur at the same time.

1. If your current client-side identity certificates do not have the Digital Signature attribute set and the client connects directly to the MobiLink server, then you must regenerate and deploy client-side certificates with the Digital Signature attribute set.
2. Update the server-side binaries (remembering to include the new FIPS-certified encryption libraries) and deploy server identity certificates with AES-encrypted private keys.
3. Update the client-side binaries (remembering to include the new FIPS-certified encryption libraries) and deploy client identity certificates with AES-encrypted private keys.

How to update a MobiLink deployment that uses FIPS-certified end-to-end encryption

1. Regenerate the primary key file referenced by the `e2ee_private_key` encryption option.
2. Shut down the MobiLink server.
3. Update the MobiLink server binaries, remembering to include the new required FIPS-certified encryption libraries.
4. Change the `e2ee_private_key` option to point to the new private key file (or replace the old file), updating the `e2ee_private_key_password`, if required.
5. Restart the MobiLink server.

UltraLite changes

UltraLite no longer supports FIPS-certified encryption on Windows Mobile.

SAP replaces Sybase in Windows registry entries

Previously, registry entries were recorded under `Software\Sybase`. Now, these registry entries appear under `Software\SAP`.

For example, `HKEY_LOCAL_MACHINE\Software\Sybase\SQL Anywhere\16.0` becomes `HKEY_LOCAL_MACHINE\Software\SAP\SQL Anywhere\17.0`.

For 64-bit Windows, `Software\Wow6432Node\Sybase` becomes `Software\Wow6432Node\SAP`.

SAP JRE replaces Oracle JRE in the software

i Note

This change was released in a version 16 Support Package.

Previously, the software included an Oracle JRE for clients to use. Now, the software uses the SAP JRE.

The SAP JRE may perform differently than the Oracle JRE. Use the `java_vm_options` option (SQL Anywhere), and/or the `-sl java` option (MobiLink) to optimize your Java VM startup settings.

JDK 8 recommended for JDBC application development

Previously, version 1.7.0 of the Java Development Kit (JDK 7) was recommended for use in JDBC applications. Now, JDK 8 is recommended.

Support for Windows XP removed

SQL Anywhere is no longer supported on Windows XP.

Support for Windows Vista removed

SQL Anywhere is no longer supported on Windows Vista.

1.9 What's New in SQL Anywhere Server 17.0

SQL Anywhere 17.0 introduces many new features and changes across the software in the areas of programming interfaces, security, performance, SQL support, catalog changes, and administration tools.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

The `readme.txt` files for all of the SQL Anywhere support packages and patch-level upgrades can be found at <http://sqlsupport.sap.com/readme/index.html>.

Top Features

OData support (database upgrade required)

The SQL Anywhere database server can act as an OData server

This functionality replaces the OData Server utility.

OData Producer information is stored in the database and OData server information is specified by network protocol options. See [OData server architecture](#).

The following changes have been made to the database server to support this feature:

New system privileges

- `MANAGE ODATA` system privilege
- `VERIFY ODATA` system privilege

See [System privileges](#).

New and enhanced SQL statements

- CREATE ODATA PRODUCER statement. See [CREATE ODATA PRODUCER statement](#).
- ALTER ODATA PRODUCER statement. See [ALTER ODATA PRODUCER statement](#).
- COMMENT statement. See [COMMENT statement](#).
- DROP ODATA PRODUCER statement. See [DROP ODATA PRODUCER statement](#).

New system view

- SYSODATAPRODUCER system view. See [SYSODATAPRODUCER system view](#).

New database server properties

- ODataAddresses
- ODataSecureAddresses

See [List of database server properties](#).

Enhancement to the -xs database server option

The -xs database server option now supports the ODATA protocol. See [-xs database server option](#).

New OData protocol options

- ExitOnError (EXIT) protocol option. See [ExitOnError \(EXIT\) protocol option](#).
- HttpMyIP (ME) protocol option. See [HttpMyIP protocol option](#).
- SecureMyIP protocol option. See [SecureMyIP protocol option](#).
- LogFile (LOG) protocol option. See [LogFile \(LOG\) protocol option](#).
- LogVerbosity protocol option. See [LogVerbosity protocol option](#).
- QuietConsole (QUIET) protocol option. See [QuietConsole \(QUIET\) protocol option](#).
- SecureServerPort (HTTPSPORT) protocol option. See [SecureServerPort \(HTTPSPORT\) protocol option](#).
- ServerPort (PORT) protocol option. See [ServerPort \(PORT\) protocol option](#).
- SSLKeyStore (KEYSTORE) protocol option. See [SSLKeyStore \(KEYSTORE\) protocol option](#).
- SSLKeyStorePassword (KEYSTOREPASSWORD) protocol option. See [SSLKeyStorePassword \(KEYSTOREPASSWORD\) protocol option](#).

Database column names in OData Producer service operations

The ServiceOperationColumnNames option allows service operations to use the result set column names from the database when naming the properties of the ComplexType used in the Return Type. The default behavior is to generate the names of properties for complex types returned by service operations. See [How to configure the OData server](#).

CSRF tokens

A new CSRFTokenTimeout option enables the feature and represents the number of seconds that CSRF tokens are valid for. The CSRF tokens feature protects OData Producers from cross-site request forgery attacks. See [How to protect against cross-site request forgery attacks](#).

Repeatable requests

The repeatable request feature allows clients to handle unreliable HTTP communications with an OData Producer. If an OData Producer fails to receive a response to a data modification request, then the client can repeat the request without risking database corruption. See [How to set up repeatable requests](#).

Extended OSDL support for OData Producer service models

New OData Service Definition Language (OSDL) syntax adds the ability to perform the following tasks:

- Explicitly set the name of tables that are exposed through an OData Producer.
- Explicitly include or exclude columns.
- Define entity sets with generated keys.
- Define associations between entities, including complex associations that use an underlying association table.
- Define navigation properties.
- Define concurrency token on entities for optimistic concurrency control.
- Define service operations to be exposed through an OData Producer.

i Note

This feature was first released in a version 16 Support Package.

See [How to create an OData Producer service model](#).

Multiple OData Producer support

The OData server now supports multiple customizable OData Producers that allow you to establish multiple database connections. Use the new embedded HTTP server option, Producers, in your OData server configuration to create OData Producers.

i Note

This feature was first released in a version 16 Support Package.

Optimistic Concurrency Control (ETags)

OData Producers now support Optimistic Concurrency Control as defined by versions 2.0 of the OData Specification. The concurrencytoken clause of the ENTITY OSDL statement is used to generate ETags that identify the state of an entity instance at the time the instance is requested.

The SQL used to generate an ETag uses SHA256 hash functions and can be complex given the types of properties (columns) and number of properties included in the concurrency token. See [ENTITY statement](#).

i Note

This feature was first released in a version 16 Support Package.

Service Operations and valid OData identifiers

OData Producers can accommodate most database procedures and functions that have parameters starting with the @ character.

Service operation names and parameter names, with the exception of @name parameters, must be valid OData identifiers.

Deep inserts

OData Producers now supports deep insert requests. These are requests to insert an entity where some links are not references to existing entities and contain new inlined entities that are also to be inserted. Deep inserts may be nested 10 deep.

Support added for point-in-time recovery (PITR) (database upgrade required)

Restore a database to a specified time stamp or to an offset in the transaction log. The following changes have been made to allow you to use this feature:

Enhanced -ad database option

Specify more than one directory for the transaction log location when restoring a database. To do this, use a semicolon as a delimiter between the directory names. See [-ad database option](#).

New -ruo database option

Recover to a specified offset in the transaction log by using this option. See [-ruo database option](#).

New -ru database option

Recover to a specified time stamp by using this option. See [-ru database option](#).

Several command line utilities now report more time-related information

The Transaction Log utility (dblog), the Log Translation utility (dbtran), and the Information utility (dbinfo), now include more time-related information in their output.

Several new database properties The following database properties have been added but are for internal use only: CurrentTimelineID, CurrentTimelineSignature, PreviousTimelineID, and TimelineBranchOffset.

New -ft option for the Translation Log utility (dblog)

The Translation Log utility (dblog) has a new option, the -ft option, for specifying a timeline when reloading SQL Remote consolidated databases. See [Transaction Log utility \(dblog\)](#).

See also:

- [Point-in-time recovery](#)
- [Tutorial: Restoring the database to an offset in the log](#)
- [Tutorial: Restoring the database to a point in time](#)

Improvements to the parameterization of statements (database upgrade required)

The following enhancements allow your applications to resolve identifiers and parameters for SQL statements at execution time. These enhancements improve security by reducing the need to use EXECUTE IMMEDIATE statements and by increasing protection against SQL injection. These enhancements also provide more flexibility to your application when the name of an object, or the value for a statement option, is not known until execution time.

Support added for indirect identifiers

Specify a variable for the object name in a statement by using an enhancement to the identifier syntax using square brackets nested in back quotes (``[@variable-name]``). Supported objects you can indirectly refer to in this manner include owners, tables, columns, and the new mutexes and semaphores. See [Indirect identifiers](#).

Support added for table reference variables (new TABLE REF data type)

Create a variable of type TABLE REF to reference a base table, temporary table, or view, and then refer to the table reference variable instead of specifying the object name directly in DML statements and as parameters in functions and procedures. See [TABLE REF data type](#).

Support added for variables in some statements

The following statements now support variables for parameters in their syntax. The variables are resolved just prior to the statement being executed.

- ALTER DATABASE statement (database upgrade required)

- CREATE MIRROR SERVER statement (database upgrade required)
- ALTER MIRROR SERVER statement (database upgrade required)
- CREATE SUBSCRIPTION statement (database upgrade required)
- CREATE PROCEDURE statement [Web services] (database upgrade required)
- ALTER PROCEDURE statement (database upgrade required)
- CREATE FUNCTION statement [Web services] (database upgrade required)
- ALTER FUNCTION statement (database upgrade required)
- CREATE EVENT statement (database upgrade required)
- ALTER EVENT statement (database upgrade required)
- CREATE EXTERNLOGIN statement
- DROP EXTERNLOGIN statement
- CREATE SERVER statement
- ALTER SERVER statement
- CREATE SEMAPHORE statement
- DROP SEMAPHORE statement
- NOTIFY SEMAPHORE statement
- WAITFOR SEMAPHORE statement
- CREATE MUTEX statement
- DROP MUTEX statement
- LOCK MUTEX statement
- RELEASE MUTEX statement

Support added for user locks: mutexes and semaphores (database upgrade required)

Build user-defined mutexes and semaphores into your application logic to achieve locking behavior and control and communicate the availability of resources. See [Mutexes and semaphores](#).

Several changes have been made to support this feature:

- New mutex-related statements: CREATE MUTEX, LOCK MUTEX, RELEASE MUTEX, and DROP MUTEX. See [CREATE MUTEX statement](#), [LOCK MUTEX statement](#), [RELEASE MUTEX statement](#), and [DROP MUTEX statement](#).
- New semaphore-related statements: CREATE SEMAPHORE, WAITFOR SEMAPHORE, NOTIFY SEMAPHORE, and DROP SEMAPHORE. See [CREATE SEMAPHORE statement](#), [WAITFOR SEMAPHORE statement](#), [NOTIFY SEMAPHORE statement](#), and [DROP SEMAPHORE statement](#).
- New ISYSMUTEXSEMAPHORE system table to store the definitions of all permanent mutexes and semaphores in the database. Access its content by using the new SYSMUTEXSEMAPHORE system view. See [SYSMUTEXSEMAPHORE system view](#).
- New sp_list_mutexes_semaphores system procedure to return the list of all mutexes and semaphores in the database. See [sp_list_mutexes_semaphores system procedure](#).
- Enhancements to sa_conn_info system procedure to add two new columns, LockObject and LockObjectType, to store the name and type of the object associated with the lock. See [sa_conn_info system procedure](#).
- Enhancements to the sa_locks system procedure to include information about locks on mutexes. See [sa_locks system procedure](#).

- New system privileges to support the administration and usage of mutexes and semaphores: CREATE ANY MUTEX SEMAPHORE, UPDATE ANY MUTEX SEMAPHORE, and DROP ANY MUTEX SEMAPHORE. See [System privileges](#).
- Two new connection properties, LockObjectOID and LockObjectType, allow you to query the object ID and type of object that a connection is blocked on. See [List of connection properties](#).

Support added for running a database in a simulated time zone (database upgrade required)

You can now create a simulated time zone and set your database to use this time zone, if you want your database to behave as though it is running on a time zone other than the system time zone of the database server. See [Time zone management](#).

Several enhancements have been made to support this feature:

- New statements: CREATE TIME ZONE, ALTER TIME ZONE, and DROP TIME ZONE. See [CREATE TIME ZONE statement](#), [ALTER TIME ZONE statement](#), and [DROP TIME ZONE statement](#).
- Enhanced syntax for the COMMENT ON statement allows you to store a comment for a time zone object in the system tables. See [COMMENT statement](#).
- New time_zone database option to specify the time zone that the database uses for time zone calculations. See [time_zone option](#).
- New ISYSTIMEZONE system table to store definitions of simulated time zone objects in the database. Access its content by using the new SYSTIMEZONE system view. See [SYSTIMEZONE system view](#).
- New database properties: Timezone and CurrentTimezoneOffset. See [List of database properties](#).
- New time_zone connection property.
- New special values: CURRENT SERVER TIME, CURRENT SERVER DATE, and CURRENT SERVER TIMESTAMP. See [CURRENT SERVER TIME special value](#), [CURRENT SERVER DATE special value](#), and [CURRENT SERVER TIMESTAMP special value](#).

Node.js

A Node.js driver is available for download from the Node Packaged Modules web site as well as GitHub. The Node.js JavaScript API can be used to connect to SQL Anywhere databases, issue SQL queries, and obtain result sets.

Support for JavaScript external environment added

JavaScript stored procedures and functions can be called from the database in the same manner as user-defined SQL stored procedures and functions

A JavaScript driver is now included with SQL Anywhere. The XS JavaScript API can be used to connect to databases, issue SQL queries, and obtain result sets. See [The JavaScript external environment](#).

JavaScript

A JavaScript driver is now included with SQL Anywhere. The XS JavaScript API can be used to connect to databases, issue SQL queries, and obtain result sets.

Enhancements to the CREATE TABLE statement (database upgrade required)

Using the OR REPLACE clause

The CREATE TABLE statement now supports the use of the OR REPLACE clause. See [CREATE TABLE statement](#).

Creating new tables based on the schema of another table

You can now create a table based directly on the definition of another table; clone a table with additional columns, constraints, and LIKE clauses; or create a table based on a SELECT statement. See [CREATE TABLE statement](#).

Support added for the %TYPE and %ROWTYPE attributes when creating or declaring variables, casting or converting values, creating or altering tables, and creating procedures

Use the %TYPE and %ROWTYPE attributes to define the data type(s) based on the data type of other objects. When creating schema objects such as columns, use the %TYPE attribute to set the data type of the object you are creating or altering, to the data type of a column in a table or view. Use the %ROWTYPE attribute to set the data types to the composite data type for a row in a table or view. When creating variables, you can also use the %TYPE and %ROWTYPE attributes to set the data type to the data type of temporary objects such as variables and cursors. See [%TYPE and %ROWTYPE attributes](#).

The following statements and functions are enhanced by this feature:

- CREATE TABLE statement (%TYPE attribute only)
- ALTER TABLE statement (%TYPE attribute only)
- CREATE PROCEDURE statement
- ALTER PROCEDURE statement
- CREATE FUNCTION statement
- ALTER FUNCTION statement
- CREATE DOMAIN statement (%TYPE attribute only)
- CREATE VARIABLE statement
- DECLARE statement
- CAST function (%TYPE attribute only)
- CONVERT function (%TYPE attribute only)

Enhancements to support SAP HANA (database upgrade or rebuild required)

The following enhancements for creating, altering, and dropping SAP HANA remote servers have been added:

- Support for SAP HANA syntax for creating, altering, and dropping remote servers using CREATE REMOTE SOURCE, ALTER REMOTE SOURCE, and DROP REMOTE SOURCE statements, instead of CREATE SERVER, ALTER SERVER, and DROP SERVER statements.
- Support for SAP HANA syntax for specifying underscores in server class names. For example, you can specify HANAODBC or HANA_ODBC.
- Support SAP HANA syntax for creating proxy tables using the CREATE VIRTUAL TABLE. See [CREATE TABLE statement](#).

See [CREATE SERVER statement](#), [ALTER SERVER statement](#), and [DROP SERVER statement](#).

In this section:

[Performance Enhancements \[page 43\]](#)

SQL Anywhere 17.0 introduces many enhancements to performance.

[Changes to SQL Statements, Functions, Procedures, and Data Types \[page 45\]](#)

SQL Anywhere 17.0 introduces many new features and enhancements to its SQL support.

[Changes to Database Administration \[page 49\]](#)

SQL Anywhere 17.0 introduces many new features and changes to administration.

[Changes to the Catalog \[page 53\]](#)

SQL Anywhere 17.0 introduces changes to system objects. All catalog changes require a database upgrade.

[Changes to Programming Interfaces \[page 54\]](#)

SQL Anywhere 17.0 introduces new features and changes in the area of programming interfaces.

[Changes to Administration Tools \[page 56\]](#)

SQL Anywhere 17.0 introduces several new and changed graphical administration tools.

[Changes to OData Support \[page 60\]](#)

SQL Anywhere 17.0 introduces enhancements to OData support.

[Changes to Security \[page 62\]](#)

SQL Anywhere 17.0 introduces many new features and changes to security.

[Documentation Enhancements \[page 65\]](#)

SQL Anywhere 17.0 introduces documentation enhancements.

[Miscellaneous Changes and Enhancements \[page 66\]](#)

SQL Anywhere 17.0 a variety of general enhancements.

[Behavior Changes, Deprecated Features, and Features That Are No Longer Supported \[page 70\]](#)

SQL Anywhere 17.0 includes changes to behavior. Some features have been deprecated or are no longer supported in this release.

1.9.1 Performance Enhancements

SQL Anywhere 17.0 introduces many enhancements to performance.

Improved performance during recovery

Several enhancements have been made to improve performance during database recovery. The speed up of recovery depends on the following factors:

The concurrency level of the workload The greater the level of concurrency of the workload the greater recovery time is improved.

The hardware resources that are available on the machine performing the recovery The more CPU cores and disk I/O spindles available, the less time required for recovery.

The mix of DDL and DML operations in the transaction log DML operations are better candidates to benefit from parallel recovery than DDL operations, which must be serialized, adding to the time it takes to recover the database.

Existence of primary keys Operations on tables with primary keys greatly speed up the recovery process

These database recovery improvements affect the following features:

Database mirroring The mirror server can now more quickly catch up to changes happening on the primary server because log operations are applied in parallel instead of sequentially.

Normal database recovery Normal database recovery is improved. In addition, applying log operations using the server switch -a is also improved.

SQL Anywhere On-Demand Edition The SQL Anywhere On-Demand Edition uses database mirroring to propagate changes in the cloud infrastructure. Wait time on the primary server is reduced with these changes, which means better response times across the cloud infrastructure.

Improvements to start-up speeds for databases running a mirror log A few internal improvements have been made to speed up the start-up of databases running a mirror log.

Enhancements to intra-query parallelism

Internal enhancements have been made to intra-query parallelism. Consider setting the `max_query_tasks` option to something higher than 1 to enable use of this feature. You can also set the option to 0 to allow the database server to decide the number on a per-query basis.

Improvements to server and client plan caching

Previously, plan caching was limited to statements in procedures, and for simple bypass statements. Now, all statements (potentially) qualify for plan caching, and client statements containing constant values may be automatically parameterized to increase reuse potential for the respective plans. Enhancements have also been made to significantly increase the robustness of the plan caching feature.

New `sp_plancache_contents` system procedure

Returns the contents of a connection's plan cache, including statistics on how long executions of plans in the cache took. See [sp_plancache_contents system procedure](#).

New `MANAGE CACHED PLANS` system privilege

Allows a user to access statistics related to cached plans on any connection. See [System privileges](#).

New `parameterization_level` database option

Sets the level of parameterization for client statements. You can override the setting of this option for an individual query by using a new query hint, `parameterization_level`, in the `SELECT` statement. The addition of this option adds a corresponding connection property that you query to return the current parameterization setting. See [parameterization_level option](#).

New `parameterization_level` query hint in `INSERT`, `UPDATE`, and `DELETE` statements

You can now set the parameterization level of an `INSERT`, `UPDATE`, and `DELETE` statement by using a new `parameterization_level` query hint option within the statement. See [INSERT statement](#), [UPDATE statement](#), and [DELETE statement](#).

New `ParameterizationPrepareCount` server, database, and connection property

Returns a counter value that reflects the number of prepare requests that have been made for automatically parameterized statements. See [ParameterizationPrepareCount property](#).

Changes to the graphical plan

If a statement is parameterized by the database server, then the graphical plan displays the parameterized version of the SQL statement, rather than the original SQL statement.

Improvements to query optimization

Enhancements to prefetching have been made to improve query performance during index scans.

Topology aware scheduling

Topology aware scheduling schedules tasks to use a single core per socket before attempting to use another core on the same socket. Use the `sa_server_option` system procedure to set topology aware scheduling on (the default setting) or off. Topology aware scheduling may improve performance for certain workloads. In cases where there is data contention in the workload, topology aware scheduling may not improve performance and can be turned off.

Improvement in use of temporary table IDs

Previously, temporary tables used during a transaction are assigned table IDs; however, the database server limits the number of temporary tables IDs and it was possible to run out of temporary table IDs during a transaction. Now, the database server can reuse temporary table IDs when it makes sense to, without impacting correctness or performance, thereby diminishing the chances of running out of IDs.

Improvements to the cache manager

Previously, initial cache size was large if the max cache size was large. Now, the initial cache is not impacted by a large max cache size setting.

1.9.2 Changes to SQL Statements, Functions, Procedures, and Data Types

SQL Anywhere 17.0 introduces many new features and enhancements to its SQL support.

Support for deprecated features will be removed in future versions. Applications should be altered to use recommended feature replacements instead of relying on deprecated features.

SQL Statements

To use any of these syntax enhancements, you must either upgrade your database, or initialize your database with a version 17 database server.

Statement labels are now optional for the LEAVE and CONTINUE statements (database upgrade required)

You are no longer required to specify a statement label when executing a LEAVE and CONTINUE statement in a SQL procedure, trigger, or batch. In the case of the LEAVE statement, if you do not specify a statement label, the innermost loop is exited. See [LEAVE statement](#), and [CONTINUE statement](#).

Support for the GOTO statement in SQL procedures, triggers, and batches (database upgrade required)

Previously, the GOTO statement was only supported in Transact-SQL procedures, triggers, and batches. Now, you can use the GOTO statement in any procedure, trigger, or batch. See [GOTO statement](#).

Enhancement to the CALL statement

The CALL statement now supports the use of the AS USER...IDENTIFIED BY clause, which allows you to execute a stored procedure as a different user. See [CALL statement](#).

Changes to CREATE DOMAIN and ALTER DOMAIN statements to support creation of row and array domains

Previously, when specifying rows and arrays in SQL statements, the full definition of the row or array needed to be repeated inline every time the row or array was used. Now, you can create a domain to hold the row or array definition and specify the domain in SQL statements, instead. The CREATE DOMAIN and ALTER DOMAIN statements have been extended to support the specification of ROW or ARRAY as the data type in the definition. For example:

```
CREATE DOMAIN MyRow ROW( a INT, b INT );
```

See [CREATE DOMAIN statement](#), and [ALTER DOMAIN statement](#).

Changes to SELECT and FETCH statements to support the use of row variables

The SELECT statement now includes the INTO VARIABLE clause to support specifying row variables and the INTO TABLE clause to support explicitly creating a new table. The FETCH statement now allows you to specify a row variable in the INTO clause. See [SELECT statement](#), and [FETCH statement](#).

Support added for semicolon delimiters in Transact-SQL procedures and batches - upgrade required

The semicolon is now supported as a statement delimiter in Transact-SQL procedures, user-defined functions, and statement batches.

You must upgrade older databases to make use of this enhancement.

Support added for revoking exercise rights for a role (REVOKE ROLE statement) - upgrade or rebuild required

Previously, if you revoked a role from a user or role, both exercise and administration rights were revoked. You can now specify whether you want to revoke exercise rights, administration rights, or both (the default), by using the new { ADMIN | EXERCISE } OPTION FOR clause of the REVOKE ROLE statement. See [REVOKE ROLE statement](#).

Enhancement to the SET MIRROR OPTION statement

This statement supports the new max_logfile_size option. See [SET MIRROR OPTION statement](#).

Enhancement to the CREATE EVENT statement

Using the OR REPLACE clause

The CREATE EVENT statement now supports the use of the OR REPLACE clause. See [CREATE EVENT statement](#).

Enhancement to the EXTERNAL NAME clause

The EXTERNAL NAME clause now supports more granular specification of libraries, including generic operating system, specific operating system, and processor architecture. The EXTERNAL NAME clause is part of the ALTER TEXT CONFIGURATION, CREATE FUNCTION, and CREATE PROCEDURE statements. See [ALTER TEXT CONFIGURATION statement](#), [CREATE FUNCTION statement \[External call\]](#), and [CREATE PROCEDURE statement \[External call\]](#).

Enhancements to the UNLOAD statement

@@rowcount global variable now incremented during unloading

The UNLOAD statement now sets the @@rowcount variable to the number of rows unloaded. See [UNLOAD statement](#).

Unloading tables

The UNLOAD TABLE statement has been extended to support the use of ALL for the QUOTES clause, and a new WITH COLUMN NAMES clause. See [UNLOAD statement](#).

System Procedures

System procedures and user-defined procedures can be altered, replaced, or dropped while executing

Previously, attempting to alter, replace, or drop a procedure that was being executed would result in an error. Now the alter, replace, or drop succeeds. Current executions use the procedure definition from when the procedure started executing. New calls to the procedure after an alter, replace, or drop, use the new definition. If a single transaction or connection calls a procedure, and that procedure is altered while executing, then if the same transaction or connection calls the same procedure again, it executes using the altered definition.

sa_db_option system procedure enhancement (database upgrade required) The val parameter of the sa_db_option system procedure now accepts a LONG VARCHAR data type rather than CHAR (128).

New sp_disk_info system procedure (database upgrade required)

The `sp_disk_info` system procedure returns disk drive information for a given file or directory path. To run the `sp_disk_info` system procedure you need the new ACCESS DISK INFO system privilege. To prevent users from running the `sp_disk_info` system privilege, the new `sp_disk_info` secure feature has been added, which is part of the `local_io` secure feature set. See [sp_disk_info system procedure](#), [-sf database server option](#), and [System privileges](#).

sa_get_dtt system procedure enhancement (database upgrade required)

The `sa_get_dtt` system procedure now returns the number of outstanding I/Os in the disk's I/O queue over which random access takes place. This can noticeably improve performance, depending on your hardware. Recalibrate your database to benefit from this enhancement.

See [sa_get_dtt system procedure](#).

New sa_list_statements system procedure (database upgrade required)

The `sa_list_statements` system procedure returns the list of statements in use by the current connection. See [sa_list_statements system procedure](#).

New sp_read_etd system procedure (database upgrade required)

The `sp_read_etd` system procedure reads a specified event trace data (ETD) file and returns the contents of the file as a set of rows. See [sp_read_etd system procedure](#).

sa_server_option system procedure enhancements (database upgrade required)

The `sa_server_option` system procedure now accepts a LONG VARCHAR value for the `val` parameter. It also now supports topology-aware scheduling. Topology-aware scheduling schedules tasks to use a single core per socket before attempting to use another core on the same socket. See [sa_server_option system procedure](#).

xp_startsmtp system procedure enhancement (database upgrade required)

The `xp_startsmtp` system procedure now makes SMTPS connections to database servers supporting SMTPS. See [xp_startsmtp system procedure](#).

xp_getenv system procedure improvement (database upgrade or rebuild required)

The `xp_getenv` system procedure has been changed so that it is a procedure that runs with invoker privileges regardless of the invoker/definer setting. The procedure returns a LONG NVARCHAR value. Previously, it returned a LONG BINARY value. See [xp_getenv system procedure](#).

i Note

This feature was first released in a version 16 Support Package.

sa_cpu_topology system procedure improvement (database upgrade or rebuild required)

The `sa_cpu_topology` system procedure has been modified to include information about user-selected physical processors specified using the `-gta` database server option or the `ProcessorAffinity` server property. This feature allows the database server to make use of newly added processors during runtime (also known as hot-add and hot-remove). See [sa_cpu_topology system procedure](#).

The restrictions and limitations for the `-gt`, `-gta`, and `-gtc` database server options are preserved.

i Note

This feature was first released in a version 16 Support Package.

sp_list_directory system procedure enhancements (database upgrade required)

In addition to returning the path, file type, and file name of all files and directories in a specified location, the `sp_list_directory` system procedure now also returns the date the file was created, last modified, and last accessed, as well as the owner and any access permissions required for the file or directory. See [sp_list_directory system procedure](#).

Functions

Functions can be altered, replaced, or dropped while executing

Previously, attempting to alter, replace, or drop a function that was being executed would result in an error. Now the alter, replace, or drop succeeds. Current executions use the function definition from when the function started executing. New calls to the function after an alter, replace, or drop, use the new definition. If a single transaction or connection calls a function, and that function is altered while executing, then if the same transaction or connection calls the same function again, it executes using the altered definition.

New `READ_SERVER_FILE` function

Use the `READ_SERVER_FILE` function to read data from the specified file on the server and return the full or partial contents of the file as a `LONG BINARY` value. See [READ_SERVER_FILE function](#).

New documentation for the `EXTENDED_PROPERTY` function

The `EXTENDED_PROPERTY` function returns the value of a given database server property and allows an optional property-specific string parameter to be specified. See [EXTENDED_PROPERTY function \[System\]](#)

Previously, this function was available for internal use only and was not documented.

Data Types

Improved support for `BYTE` strings

To improve support for `BYTE` strings, the following enhancements and changes have been made:

New `BYTE_STUFF` function

Deletes multiple bytes from one string and replaces them with different bytes. See [BYTE_STUFF function](#).

New `BYTE_INSERTSTR` function

Inserts a string into another string at a position specified in bytes. See [BYTE_INSERTSTR function](#).

New `BYTE_LOCATE` function

Finds and replaces a `BYTE` string with another `BYTE` string, and returns the new results. See [BYTE_LOCATE function](#).

New `BYTE_REPLACE` function

Returns the position of one `BYTE` string within another. See [BYTE_REPLACE function](#).

Changes to `BYTE_SUBSTR` function

Previously, this function returned `BINARY`, `LONG BINARY`, `VARCHAR`, `LONG VARCHAR`, `NVARCHAR`, or `LONG NVARCHAR`, depending on the type of the input string expressions. Now, this function only returns `BINARY` or `LONG BINARY`. See [BYTE_SUBSTR function](#).

Changes to REPLACE function

This function can now return LONG BINARY. If the inputs to this procedure are binary strings, then this function behaves the same as the new BYTE_REPLACE function. See [REPLACE function](#).

Changes to LOCATE function

If the inputs to this procedure are binary strings, then this function behaves the same as the new BYTE_LOCATE function. See [LOCATE function](#).

Changes to STUFF function now returns LONG BINARY

Previously, the STUFF and INSERTSTR functions returned LONG NVARCHAR or LONG VARCHAR, even if the input expressions were BINARY. Now, if all of the input expressions are BINARY, then these functions return LONG BINARY. See [STUFF function](#).

Changes to INSERTSTR function now returns LONG BINARY

Previously, the INSERTSTR function returned LONG NVARCHAR or LONG VARCHAR, even if the input expressions were BINARY. Now, if all of the input expressions are BINARY, then this function returns LONG BINARY. See [INSERTSTR function](#).

1.9.3 Changes to Database Administration

SQL Anywhere 17.0 introduces many new features and changes to administration.

Support added for user-defined upgrades (database upgrade required)

Create and run your own upgrade SQL scripts against a SQL Anywhere database to upgrade it. This feature is supported by using new syntax (SCRIPT FILE) provided in the ALTER DATABASE...UPGRADE statement, as well as in the *Upgrade Database Wizard* in SQL Central. See [User-defined upgrades \[page 92\]](#).

Support added for rebuilding a production database with minimum downtime

You can now rebuild a production database while the database is running, which reduces downtime. Several new items have been added to support minimum downtime rebuild:

New options for the Unload utility (dbunload) (database rebuild required) The Unload utility (dbunload) now supports the -ao, -aob, -aot, and -dt options. See [Unload utility \(dbunload\)](#).

New option for the Backup utility (dbbackup) The Backup utility (dbbackup) now supports the -wa option. See [Backup utility \(dbbackup\)](#).

New info type for the db_backup function The db_backup function now supports the new DB_BACKUP_INFO_WAIT_AFTER_END info type. See [db_backup function](#).

The DBChangeLogName method now changes the a_change_log value The parameter (const a_change_log *pcl) has been changed to (a_change_log *pcl). See [DBChangeLogName\(a_change_log *pcl\) method](#).

The a_change_log structure supports three new member data fields The new member data fields are starting_offset, current_relative_offset, and end_offset. The zap_current_offset, zap_starting_offset, and encryption_key members now have type const char*. See [a_change_log structure](#).

The an_unload_db structure supports new members The new members are online_rebuild, online_rebuild_from_backup, shouldstoprtn, online_rebuild_max_apply_sec, and temporary_directory. See [an_unload_db structure](#).

New Stop processing callback routine The Stop processing callback routine is called when a tool needs to decide if it should continue processing or stop processing. See [Callback functions](#).

Validation enhancements

The Validation utility (dbvalid) supports in-memory backup validation When you use dbvalid to autostart a database server, you can specify that backup validation is performed using in-memory backup validation mode. Use this mode to validate backup databases without modifying them or their transaction log files. See [Validation utility \(dbvalid\)](#).

Enhancement to in-memory database validation

The -im database server option has been extended to provide in-memory backup validation for a database without modifying it (-im v/-im valid). See [-im database server option](#).

Enhancements to validation The VALIDATE statement, Validation utility (dbvalid), and sa_validate system procedure have been enhanced with options that eliminate the possibility of false reporting of database corruption. In addition, sa_validate now returns a code to indicate successful or failed validation, along with the name of the invalid table in the case of failure.

The VALIDATE statement supports two new clauses:

WITH DATA LOCK

Prevents transactions from modifying the table schema or data by applying exclusive data locks on the specified tables. Transactions can read, but not modify the table data or schema.

When the FOREIGN KEY clause is specified, then exclusive data locks are also applied to the tables that contain the corresponding primary keys.

WITH SNAPSHOT

Ensures that only committed data is checked by applying snapshot isolation. Transactions can read and modify the data. This clause requires that the database have snapshot isolation enabled (with the allow_snapshot_isolation database option). Because this clause uses snapshot isolation, performance is often affected.

The Validation utility supports two new options:

-wl Runs validation with exclusive data locks applied to the table(s), prohibiting other threads from modifying the data in the table(s) being validated.

-ws Runs validation with snapshot isolation applied to the table(s).

sa_validate system procedure enhancements: sa_validate supports a new parameter, `isolation_type`, that allows you to specify whether data locks or snapshot isolation should be applied to the table(s) during validation.

The sa_validate system procedure now returns a code that can be used to determine if validation succeeded or failed. Also, if validation fails for a table, then sa_validate now returns the name of the table that caused the failure.

See [VALIDATE statement](#), [Validation utility \(dbvalid\)](#), and [sa_validate system procedure](#).

Support added for tracking database server property values

Rather than having to poll, analyze, and store database server property values at regular intervals, you can now configure your database or database server to track database server properties that return numeric values for a specified amount of time or until a specified amount of memory has been reached. See [Database server property tracking](#).

The following changes have been made to support this feature:

New sp_property_history system procedure (database upgrade required)

Returns values for all database server properties tracked by the database. See [sp_property_history system procedure](#).

New PROPERTY_IS_TRACKABLE function

Returns whether or not you can maintain historical data for the specified database server property by storing its tracked values. See [PROPERTY_IS_TRACKABLE function](#).

New database server options for sa_server_option system procedure (database upgrade required)

- PropertyHistoryList
- PropertyHistorySize

See [sa_server_option system procedure](#).

New database option for sa_db_option system procedure (database upgrade required)

- PropertyHistoryList

See [sa_db_option system procedure](#).

New database server options

- -phl database server option. See [-phl database server option](#).
- -phs database server option. See [-phs database server option](#).

New database server properties

- CacheMisses
- CompletedReq
- ConnCount
- CurrRead
- CurrWrite
- DiskWrite
- PropertyHistoryList
- PropertyHistoryListActual
- PropertyHistorySize
- PropertyHistorySizeBytes

See [List of database server properties](#).

New database property

- PropertyHistoryList

See [List of database properties](#).

New system privilege

- MANAGE ANY PROPERTY HISTORY system privilege (database upgrade required)

See [System privileges](#).

Enhancement to the -sf database server option

The MANAGE_PROPERTY_HISTORY feature has been added to the MANAGE_SERVER feature set. See [-sf database server option](#).

Support added for starting and stopping connection listeners while the database server is running (database upgrade or rebuild required)

You can now start and stop HTTP, HTTPS, TCP/IP, and shared memory connection listeners without having to restart the database server.

Several enhancements have been made to support this feature:

System procedures

Three new system procedures have been added: `sp_start_listener`, `sp_stop_listener`, and `sp_http_listeners`. See [sp_start_listener system procedure](#), [sp_stop_listener system procedure](#), and [sp_http_listeners system procedure](#).

Database server properties

The following database server properties have been added: `HttpListeners`, `HttpsListeners`, and `TcpipListeners`. See [Descriptions of database server properties](#).

New MANAGE LISTENERS system privilege

If you create a new database using version 16.0 and later, or upgrade a pre-16.0 database, then this privilege is found in the `SYS_AUTH_SSO_ROLE` and `SYS_RUN_REPLICATION_ROLE` system roles.

If you have upgraded a version 16.0 database, then this privilege is granted to the `UPGRADE_ROLE` system privilege. See [System privileges introduced in upgrades](#).

New MANAGE_LISTENERS feature The `MANAGE_LISTENERS` feature has been added to the `MANAGE_SERVER` feature set. See [-sf database server option](#).

Enhancements to event tracing (database upgrade required) New trace events have been added to log each statement that executes inside a scheduled event.

Enhancements to SQL Anywhere auditing (database upgrade or rebuild required)

The SQL Anywhere auditing feature now allows users to track additional events and log auditable events to an audit log file or the system event log. Audit events are a sub-set of system trace events. See [sa_enable_auditing_type system procedure](#), [sa_disable_auditing_type system procedure](#), and [audit_log option](#).

The following database objects have been changed to support enhanced auditing:

sa_enable_auditing_type system procedure The `sa_enable_auditing_type` system procedure now supports logging invocations of `xp_cmdshell`.

sa_disable_auditing_type system procedure The `sa_disable_auditing_type` system procedure now supports logging invocations of `xp_cmdshell`.

audit_log option The `audit_log` option has been added so that you can specify the type and location of the audit log.

The following system procedures require the MANAGE AUDITING system privilege and now include the include_audit_events parameter:

- `sp_trace_events` system procedure. See [sp_trace_events system procedure](#).
- `sp_trace_event_fields` system procedure. See [sp_trace_event_fields system procedure](#).
- `sp_trace_event_session_events` system procedure. See [sp_trace_event_session_events system procedure](#).
- `sp_trace_event_session_targets` system procedure. See [sp_trace_event_session_targets system procedure](#).
- `sp_trace_event_session_target_options` system procedure. See [sp_trace_event_session_target_options system procedure](#).

i Note

This feature was first released in a version 16 Support Package.

Enhancements to determining what statements are currently being executed (database upgrade required for sa_stack_track system procedure)

Previously you could only use the LastStatement property and the associated RememberLastStatement option and property to return the last statements executed by the database server. However, in cases where the last client statement caused additional statements to run (for example, when a client statement called a stored procedure that executes additional SQL statements), the results did not include the additional statements.

Now you can use the STACK_TRACE function and the sa_stack_trace system procedure to return complete information about the statements that the database server is currently running using three new parameters: stack_frames, detail_level, and connection_id. See [STACK_TRACE function](#), and [sa_stack_trace system procedure](#).

Enhancements to auto-restarting a database (-ufd database server option)

You now have more control over the action that the database server takes when a fatal error or assertion failure occurs on a database. The -ufd database server option has been extended to support two new auto-restart behaviors using two new values: restart_escalate and restart_abort. The existing restart option is still supported but deprecated; use the equivalent restart_escalate value instead. For more information on what these new -ufd values do, see [-ufd database server option](#).

1.9.4 Changes to the Catalog

SQL Anywhere 17.0 introduces changes to system objects. All catalog changes require a database upgrade.

New Catalog Features

The SYSPROCPARM system view is now updated at each checkpoint

Previously, the SYSPROCPARM system view was only updated when a procedure or function was created or altered, or when an ALTER PROCEDURE RECOMPILE statement was executed.

Procedure and function values in SYSPROCPARM became out-of-date if they relied on another object, such as a table, view or procedure, that was then altered.

Now, SYSPROCPARM is updated whenever a checkpoint is run if the out-of-date procedure or function meets the following conditions:

- The procedure or function has been referenced since it was altered.
- The procedure either has a RESULT clause or is not a recursive procedure with calls nested ten deep to other procedures that do not have RESULT clauses.

See [SYSPROCPARM system view](#).

New catalog objects

The following objects have been added to the catalog:

- ISYSDATABASEVARIABLE and SYSDATABASEVARIABLE
- ISYSMUTEXSEMAPHORE and SYSMUTEXSEMAPHORE
- ISYSODATAPRODUCER and SYSODATAPRODUCER
- ISYSTIMEZONE and SYSTIMEZONE

1.9.5 Changes to Programming Interfaces

SQL Anywhere 17.0 introduces new features and changes in the area of programming interfaces.

Support added for using one CLR external environment per database server

Previously, the database server used one CLR external environment per database running on the database server. Now, you can configure the database server to use one CLR external environment for all databases running on the database server. The following new properties and database server options have been added as part of this feature:

- -sclr database server option
- SingleCLR database server property
- SingleCLRInstanceVersion database server option

Support added for using any .NET version with a CLR external environment

You can use any version of .NET in a CLR external environment. Use the ALTER EXTERNAL ENVIRONMENT statement to identify which version of .NET you want to use. See [The CLR external environment](#).

Support added for using one Java VM per database server

Previously, the database server used one Java VM per database running on the database server. Now, you can configure the database server to use one Java VM for all databases running on the database server. The following new properties and options have been added as part of this feature:

- JavaVM database property
- JavaVM database server property
- SingleJVM database server property
- -sjvm database server option
- SingleJVMLocation database server option
- UseSingleJVMInstance database server option

Embedded SQL: Support added for wide updates and deletes

Embedded SQL now supports wide updates and deletes using the ARRAY clause of the EXECUTE statement [ESQL]. See [EXECUTE statement \[ESQL\]](#), [UPDATE statement](#), [DELETE statement](#), and [Wide deletes using Embedded SQL](#).

ODBC: Support added for wide updates and deletes

ODBC now supports wide updates and deletes using SQLSetStmtAttr(SQL_ATTR_PARAMSET_SIZE) and SQLBindParameter. See [Executing statements with bound parameters](#).

JDBC: Support added for batched updates and deletes

JDBC now supports wide or batched updates and deletes using PreparedStatement.addBatch() and PreparedStatement.executeBatch(). See [JDBC batch methods](#).

SQL Anywhere C API: Support added for wide inserts

Supports version 4 of the SQL Anywhere C API wide inserts. The following methods are now available when `_SACAPI_VERSION` is defined as 4.

```
sqlany_set_batch_size( a_sqlany_stmt stmt, size_t num_rows )
```

This method sets the number of rows in the batch for the specified statement. For wide insert, this is the number of rows inserted. The default value is 1, meaning a single row execution. Setting the batch size to 0 results in an error.

```
sqlany_get_batch_size( a_sqlany_stmt stmt )
```

This method gets the number of rows in the batch for the specified statement.

```
sqlany_set_param_bind_type( a_sqlany_stmt stmt, row_size )
```

This method sets the byte size of a row when row-wise binding is used. This value determines the number of bytes to skip to find the next bound value. The default value for `row_size` is 0, indicating column-wise binding.

SQL Anywhere C API: Support added for wide fetches

Support for wide fetches has been added to version 4 of the SQL Anywhere C API. The following methods are now available when `_SACAPI_VERSION` is defined as 4.

- `sqlany_set_rowset_size` method
- `sqlany_set_rowset_pos` method
- `sqlany_get_rowset_size` method
- `sqlany_set_column_bind_type` method
- `sqlany_bind_column` method
- `sqlany_fetched_rows` method

SQL Anywhere C API: Support added for national character types

Support for national character (NCHAR, NVARCHAR, LONG NVARCHAR, NTEXT) types has been added to version 4 of the SQL Anywhere C API. This functionality is available when `_SACAPI_VERSION` is defined as 4. The `native_type` field of the `a_sqlany_column_info` structure includes `DT_NVARCHAR`, `DT_NFIXCHAR`, `DT_NSTRING`, and `DT_LONGNVARCHAR` as possible values.

SQL Anywhere C API: Support added for callbacks

Support for callbacks has been added to version 3 of the SQL Anywhere C API. The `sqlany_register_callback` function is available when `_SACAPI_VERSION` is defined as 3. This function can be used to register callback functions.

i Note

This enhancement was first released in a version 16 Support Package.

JDBC driver

The JDBC driver support for escape sequences has been enhanced to include the `TIMESTAMPADD` and `TIMESTAMPDIFF` functions. See [JDBC escape syntax](#).

i Note

This enhancement was first released in a version 16 Support Package.

ODBC driver

The ODBC driver support for escape sequences has been enhanced to include the `TIMESTAMPADD` and `TIMESTAMPDIFF` functions.

Previously, calling the ODBC `SQLGetInfo` function to retrieve the version of the ODBC driver (`SQL_DRIVER_VER`) returned a string that did not include the build number of the driver. Now, the format of the string returned is `xx.yy.zzzz` where `xx` is the 2-digit major version number, `yy` is the 2-digit minor version number, and `zzzz` is the build number (for example, `16.00.1234`). See [ODBC escape syntax](#).

i Note

This enhancement was first released in a version 16 Support Package.

.NET 2.0 data provider removed

The previous version of SQL Anywhere included a .NET version 2.0 data provider. This provider has been removed. The .NET version 3.5 provider can be used in its place. Note that .NET versions 3.5, 4.0, and 4.5 are supported.

Spatial API: New `ST_Buffer` method

The `ST_Buffer` method returns the `ST_Geometry` value representing all points within the specified distance of the geometry-expression. See [ST_Buffer method](#).

PHP support

Deprecated PHP functions removed (`sqlanywhere_*` family)

The deprecated PHP functions `sqlanywhere_commit`, `sqlanywhere_connect`, and so on have been removed. These functions were deprecated and replaced by the `sasql_commit`, `sasql_connect`, and so on family of functions in version 11.

Python support

Django driver (`sqlany-django`)

Django is a Python-based framework for creating web sites. The new `sqlany-django` driver allows customers to use the SQL Anywhere database server as a back end for a Django-based web site.

The most current software and documentation for the SQL Anywhere Django driver is available from the PyPI (Python Package Index) web site (<https://pypi.python.org/pypi>). Search for `sqlany-django`.

SQLAlchemy driver (`sqlalchemy-sqlany`)

SQLAlchemy is a Python-based toolkit and object relational mapper. The new `sqlalchemy-sqlany` dialect allows users to create SQLAlchemy applications that can communicate with a SQL Anywhere database server.

The most current software and documentation for the SQL Anywhere SQLAlchemy dialect is available from the PyPI (Python Package Index) web site (<https://pypi.python.org/pypi>). Search for `sqlalchemy-sqlany`.

1.9.6 Changes to Administration Tools

SQL Anywhere 17.0 introduces several new and changed graphical administration tools.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

Monitor enhancements

i Note

Adobe will stop updating and distributing the Flash Player at the end of 2020. Because the SQL Anywhere Monitor is based on Flash, you cannot use it once Flash support ends. In many cases, tasks that were previously performed in the Monitor can be performed in the SQL Anywhere Cockpit. See [SQL Anywhere Monitor Non-GUI User Guide](#).

The Monitor provides you with information about the health and availability of SQL Anywhere databases, MobiLink servers, and MobiLink server farms. It can also provide information about the availability of web servers, proxy servers, and host computers.

Monitor web servers, proxy servers, and host computers

Monitor the availability of web servers, including non-SQL Anywhere database servers, proxy servers, and host computers. For example, you can create a resource to monitor the availability of a printer in your network. See [Adding a web service resource](#).

Create custom metrics and alerts for a database resource

You can now create your own metrics and alert thresholds to monitor data that you are interested in. You can also create widgets displaying data collected from the user-defined metric. See [Defining custom metrics for a SQL Anywhere database resource](#).

Organize alerts

Previously, the Monitor displayed alerts by severity and then by time. Alerts can now be organized chronologically and by alert type. When alerts are organized by type, the *Alert Type Details* window displays all the alerts of that type in one column and the alert details in another column. All alerts of a given type can be resolved, marked active, or deleted from this window. See [Defining custom metrics for a SQL Anywhere database resource](#).

Resolved alerts are now hidden

The *Alerts List* widget now hides resolved alerts by default. See [Defining custom metrics for a SQL Anywhere database resource](#).

Enhancements to the accessibility features



The font size and color theme can now be changed in the Monitor. Improvements to keyboard navigation have also been implemented. See [Customizing accessibility features](#).

SQL Anywhere Monitor no longer supports Relay Server

Support for monitoring Relay Servers has been removed from the Monitor.

Interactive SQL enhancements

Manage multiple connections within one instance of Interactive SQL with a new tabbed interface

Manage multiple connections within one instance of Interactive SQL with tabs. To open a new tab, clicking  *Windows* .

Improved History tab now contains the content from the Messages tab

The *History* tab in the *Results* pane of Interactive SQL now shows history and execution messages, warnings, and errors. The *Messages* tab has been removed. The format of the history file has changed, and it is no longer compatible with older versions of the software.

New visual indicator when database locks are present

The status bar at the bottom of the Interactive SQL window now displays an indicator for the number of database locks being held, as well as an indicator when other connections are blocked. The new [Locking Viewer](#) window also displays this information with more detail. See [Viewing locks \(Interactive SQL\)](#).

Enhanced Microsoft Excel interoperability

Data can now be imported and exported from Microsoft Excel more easily using the INPUT and OUTPUT statements, or the *Import Wizard* and *Export Wizard*. See [Importing data from files \[Interactive SQL\]](#), [Importing data with the Import Wizard](#), and [Exporting query results to a CSV or Microsoft Excel spreadsheet file \[Interactive SQL\]](#).

All results from all statements are now displayed

Previously, only the first result from the last statement was displayed in Interactive SQL by default. Now, results from all executed statements are displayed in the [Results](#) pane.

Compare graphical plans

You can now compare two graphical plans using the [Compare Plans](#) window in Interactive SQL. Use the [Plan Viewer](#) in Interactive SQL to create and save the graphical plans. Then, open the two plans in the [Compare Plans](#) tool and compare them. See [Tutorial: Comparing plans in Interactive SQL](#).

Improvements to Interactive SQL shortcuts for macOS

Interactive SQL keyboard shortcuts for use on macOS have been updated to take advantage of Mac conventions:

macOS shortcut	Action
Command+Q	Closes Interactive SQL.
Command+R	Executes all text in the SQL Statements pane. Previously this was accomplished using F5.
Command+Option+R	Executes the text that is selected in the SQL Statements pane.
Control+R	Executes the selected SQL statement, and then selects the next statement. This shortcut allows you to step through a series of SQL statements.

Shortcuts for executing COMMIT and ROLLBACK statements have been removed on macOS. See [Keyboard shortcuts \(Interactive SQL\)](#)

Improvements to accuracy of execution times

The execution time for SQL statements has been improved to include the time taken for the application to fetch results from executed statements.

Text complete improvements

Improvements have been made to the text-completer feature. For example when you complete a table name that is not owned by you, the owner is automatically added, and when you complete an identifier which contains blanks, the identifier is quoted as part of the completion.

Localized numbers and dates

Numbers and dates are now displayed using the region-specific format of your computer. Previously, the default behavior was to the same format regardless of the locale of the computer.

For example, when Interactive SQL runs in locales that use a comma as the decimal separator, a semicolon is used as the default field delimiter in the data that is imported and exported as text. In locales that use a period as the decimal separator, the default field delimiter is still a comma.

You can change the default behavior by clicking **Tools > Options > Import/Export** and adjusting the text options.

SQL Central enhancements

Sybase Central now called SQL Central

The graphical administration tool, Sybase Central, has been renamed SQL Central.

Database object filtering

Filter the list of database objects that appear in the right pane using the *Filter* field in the top right corner of the right pane. The *Filter* field filters items that contain the given text. See [SQL Central navigation](#)

Organize your database objects by creating folders

In addition to the system-defined database-object folders in the left pane, you can create your own folders. You can create folders and manually add the database objects, or you can create smart folders, which are dynamically populated and maintained based on an expression that you create. All folder hold one type of database object See [Creating folders](#).

New option in the Compare Database Schema tool

Previously, the contents of the password fields were always included when comparing database schemas. Now, you can specify whether the passwords should be compared during the database comparison. See [Comparing database schemas](#).

The display language can be overridden using the SQL Anywhere registry key

SQL Central respects the SQL Anywhere registry key SQL Anywhere executable: `... \Software\SAP\SQL Anywhere\17.0` and the SALANG environment variable. Previously, SQL Central used the language registry value from the registry key `... \Software\SAP\SQL Central\17.0.0` This registry key has been retired.

See [Language Selection utility \(dblang\)](#) and [Registry settings on installation](#).

The Relay Server plug-in is no longer supported

The Relay Server plug-in for SQL Central is no longer supported.

Deployment Wizard for Windows enhancements

Deployment Wizard for Windows

The Deployment Wizard for Windows can now create upgrade install images. This type of install is intended to update an existing installation of the software. See [The Deployment Wizard for Windows](#).

Miscellaneous changed or removed features

Graphical administration tools upgraded to use Java Runtime Edition 1.8.0 (JRE 8)

Graphical administration tools that use Java now use Java Runtime Edition 1.8.0 (JRE 8).

Previously, version 1.7.0 of the Java Run-time Edition (JRE 7) was included with the software. Now, the administration tools use Java Run-time Edition 1.8.0 (JRE 8) from SAP.

The fast launcher option for SQL Central and Interactive SQL has been removed

The fast launcher option in SQL Central or Interactive SQL that reduced the startup time for these graphical administration tools has been removed. Modern computers have made this feature irrelevant.

Connect window changes

The *Save as an ODBC Data Source* option that created an ODBC data source based on the information provided in the *Connect* window has been removed.

1.9.7 Changes to OData Support

SQL Anywhere 17.0 introduces enhancements to OData support.

OData Enhancements

Enhanced log generation

Generated logs now identify the OData Producer and the request associated with each log event.

i Note

This feature was first released in a version 16 Support Package.

ConnectionPoolMaximum

The OData Producer ConnectionPoolMaximum default configuration option is set to half of the database server's maximum connections.

i Note

This feature was first released in a version 16 Support Package.

Association improvements

Associations are now annotated as referential constraints in the metadata, and association properties of referential constraints are now visible by default.

An association may have OnDelete attributes in the metadata, which documents how the dependent entity instance is affected when the associated principal entity instance is deleted.

i Note

This feature was first released in a version 16 Support Package.

Improved metadata caching

OData Producers use fewer objects when caching metadata for users that have identical access from the Producers' prospective.

i Note

This feature was first released in a version 16 Support Package.

OSDL file enhancement

OSDL files support added escape sequences in quoted strings.

i Note

This feature was first released in a version 16 Support Package.

Model paths use servlet contexts

When specifying the *Model* OData Producer option path, the path is relative to the servlet's context or the server's current directory. The path and file name are processed for environment variable references in the `${variable-name}` format.

i Note

This feature was first released in a version 16 Support Package.

Proxy table insertions

All key properties must be explicitly specified when creating entities in entity sets that are proxy tables in a SQL Anywhere database.

i Note

This restriction was first released in a version 16 Support Package.

substringof return value

The `substringof(s1, s2)` filter returns whether the `s1` string is a substring of `s2`.

i Note

This change was first released in a version 16 Support Package.

OData Behavior Changes

Search strings are restricted to 254 bytes

When using long search strings with OData filter functions, such as `startswith`, `substringof`, and `indexof` searches are performed on the first 254 bytes only.

i Note

This restriction was first released in a version 16 Support Package.

The OData Server utility and OData Server Stop utility are not supported

The database server can be started as an OData server. OData Producer information is stored in the database. See [OData server architecture](#).

The Service utility (`dbsvc`) for Windows no longer supports the `-t OData` option.

The OData servlet namespace has changed

The OData servlet namespace has changed to `com.sap.odata.producer.servlets.ODataServlet`. The default namespace in the metadata has changed to `SAPSQLData`. See [How to set up an OData server](#).

StartsWith() string filters are no longer restricted to 254 characters

StartsWith has been optimized to use LIKE when the search string is a literal or parenthesized literal whose length is less than 126.

i Note

This feature was first released in a version 16 Support Package.

1.9.8 Changes to Security

SQL Anywhere 17.0 introduces many new features and changes to security.

Significant improvements to securing password information in the database

Several changes have been made in this release to secure password information in the catalog and during database operations. This is the list of new features. Additional information is listed in the Catalog Changes and Behavior Changes sections.

New ACCESS USER PASSWORD system privilege

This system privilege allows a user to access views that display password hashes and perform operations that involve accessing passwords, such as unloading, extracting, or comparing databases. See [System privileges](#).

If you create a new database using version 16.0 and later, or upgrade a pre-16.0 database, then this privilege is found in the SYS_AUTH_SSO_ROLE and SYS_RUN_REPLICATION_ROLE system roles.

If you have upgraded a version 16.0 database, then this privilege is granted to the UPGRADE_ROLE system privilege. See [System privileges introduced in upgrades](#).

New -up option for the Unload utility (dbunload) and the Extract utility (dbxtract)

This option allows the unloading of passwords. There are behavior changes associated with its use. See [UNLOAD utility](#), and [Extract Utility \(dbxtract\)](#).

Changes to viewing password information in SQL Central (database upgrade or rebuild required)

If a user does not have the ACCESS_USER_PASSWORD system privilege, then database server prompts the user to designate a password when:

- duplicating an LDAP server using copy/paste or drag and drop methods or drag-and-drop
- testing the connection to an LDAP server if the LDAP server's definition has been modified in the property sheet but not yet saved to the database, and its access password hasn't been modified.
- testing the connection to a remote server if the remote server definition has been modified in the property sheet but not saved to the database, and the remote server has an external login for the current user.

Previously, when creating a new external login or LDAP server in SQL Central by copying and pasting an existing one, the password was copied too. Now, the password is only copied if you have the ACCESS_USER_PASSWORD system privilege. If you do not have this privilege, then you are prompted to supply a password for the new external login or LDAP server.

Previously, when creating a new user in SQL Central by copying and pasting an existing user, the password was copied too. Now, you are prompted to specify a password for the new user (even if you have the ACCESS_USER_PASSWORD system privilege).

New password obfuscation options for storing passwords on the client computer

The EncryptedPassword connection parameter has been renamed to EncodedPassword (short form ENP) to better suit the options available for encoding and decoding of passwords stored in ODBC data sources. New options include password encoding/decoding for a specific computer and user/computer combination. On Microsoft Windows, the Microsoft Cryptography API is now used to encode passwords intended to be used only on the computer or by the user/computer where the data source was created. The EncryptedPassword connection parameter is still supported for backwards compatibility. Data sources will now contain the short form ENP instead of EncryptedPassword. See [EncodedPassword \(ENP\) connection parameter](#).

On Microsoft Windows, the *ODBC Configuration for SQL Anywhere* window of the *ODBC Data Source Administrator* has new options to support password encoding. See [ODBC Configuration for SQL Anywhere window: Login tab](#).

The Data Source utility (dbdsn) has new options to support password encoding for a specific computer and user/computer combination. See [Data Source utility \(dbdsn\)](#).

New obfuscation options for configuration files

The File Hiding utility (dbfhide) supports new options for encoding of files for a specific computer and user/computer combination on UNIX and Linux systems. See [File Hiding utility \(dbfhide\)](#).

Support added for using RSA key pairs to encrypt, sign, and verify messages (database upgrade required)

Support has been added for using RSA key pairs to encrypt small blocks of data and to sign and verify messages. See [Message encryption using public and private keys](#).

Several changes have been made to support this feature:

- New sp_generate_key_pair system procedure to create RSA key pairs (database upgrade or creating a new database required). See [sp_generate_key_pair system procedure](#).
- New SECURE_SIGN_MESSAGE function to digitally sign messages. See [SECURE_SIGN_MESSAGE function \[String\]](#).
- New SECURE_VERIFY_MESSAGE function to verify digitally signed messages. See [SECURE_VERIFY_MESSAGE function \[String\]](#).
- Enhancements to ENCRYPT function to support the RSA encryption algorithm and OAEP and PKCS1 padding types. See [ENCRYPT function \[String\]](#).
- Enhancements to DECRYPT function to support the RSA encryption algorithm and OAEP and PKCS1 padding types. See [DECRYPT function \[String\]](#).

Support added for database isolation

Support has been added for enabling database isolation for a database server. When database isolation is turned on, each database behaves as though it is the only database running on the database server.

The following changes have been made:

- New -edi database server option to enable or disable database isolation. See [-edi database server option](#).
- The DATABASE_ISOLATION feature has been added to the SERVER_SECURITY feature set. This feature can be used to temporarily disable database isolation for the current connection. See [-sf database server option](#).

New special values for determining the user context when running procedures

The database server now supports four special values that allow you to determine the logged in user, invoking user, effective user, and the procedure owner when running procedures. These special values are particularly helpful in cases where you have nested procedures, and procedures that are defined with SQL SECURITY INVOKER and SQL SECURITY DEFINER, and the executing user isn't immediately clear:

- SESSION USER special value. See [SESSION USER special value](#).
- INVOKING USER special value. See [INVOKING USER special value](#).
- EXECUTING USER special value. See [EXECUTING USER special value](#).
- PROCEDURE OWNER special value. See [PROCEDURE OWNER special value](#).

As part of this feature, the following reserved words have been added to the software:

- invoking
- executing
- session_user
- invoking_user
- executing_user
- procedure_owner

Changes to procedure permissions for databases initialized or upgraded using the invoker security model

If a database is initialized or upgraded using the legacy definer security model, then the following procedures will continue to verify that the current logged in user has permission to execute the procedure. For databases that have been upgraded or initialized using the more secure invoker model, these stored procedures now verify that the current effective user has the appropriate permissions to execute the procedure.

The affected procedures are:

- sa_server_option
- sa_db_option
- sp_create_secure_feature_key
- sp_alter_secure_feature_key
- sp_drop_secure_feature_key

See [sa_server_option system procedure](#), [sa_db_option system procedure](#), [sp_create_secure_feature_key system procedure](#), [sp_alter_secure_feature_key system procedure](#), and [sp_drop_secure_feature_key system procedure](#).

The operating system certificate store can now be used for secure connections

TLS connections (including HTTPS web procedures, LDAP User Authentication and secure SMTP connections) can now use the operating system certificate store to obtain a trusted certificate for secure connections. If the server certificate is signed by any certificate in the store, then the connection succeeds. Support for this feature is included in the SQL Anywhere plug-in for SQL Central.

You can now specify * as the certificate file name when using the trusted_certificates protocol option. This value causes the client software to search the operating system certificate store for an appropriate root certificate.

You can now specify * as the certificate file name when using the file option of the CERTIFICATE clause of the CREATE PROCEDURE or CREATE FUNCTION statement. This value causes the database server to search the operating system certificate store for an appropriate root certificate.

When making a TLS or HTTPS connection, if none of the certificate_name, certificate_company, or certificate_unit protocol options are set, then the host name of the database server is verified against the host name of the database server certificate. To prevent this host name check, pick one of the following options:

When making a TLS connection

Set the new skip_certificate_name_check protocol option to ON. See [Network protocol options, skip_certificate_name_check protocol option \(client side only\)](#), and [Encryption \(ENC\) connection parameter](#).

When creating or altering a web client procedure that makes a secure request to an HTTP server

Set the new skip_certificate_name_check protocol option to ON. See [CREATE PROCEDURE statement \[Web service\]](#) or [CREATE FUNCTION statement \[Web service\]](#).

When starting an email session under SMTP (database upgrade required) Set the skip_certificate_name_check parameter of the xp_startsmtp system procedure to 1. See [xp_startsmtp system procedure](#).

Database encryption is more secure (database upgrade or rebuild required)

Database encryption is more secure due to improvement of the initialization vector.

PAM user authentication supported for UNIX and Linux systems (database upgrade or rebuild required)

PAM user authentication (PAMUA) is available on all supported UNIX and Linux platforms. See [PAM user authentication](#).

LDAP user authentication

The LDAP user authentication (LDAPUA) libraries have been reimplemented and have new file names. There are two libraries, one for FIPS-certified encryption and the other for non-FIPS.

Support added for Linux running on ARM processor

SQL Anywhere is now supported on Linux-based ARM processors (for example, Raspberry Pi).

1.9.9 Documentation Enhancements

SQL Anywhere 17.0 introduces documentation enhancements.

Improved backup and recovery documentation

The topics on backing up and restoring your database, and recovering data, have been reorganized to make them easier to navigate, to provide more information about the types of backups available, and to standardize terminology. Additionally, tasks have been reorganized by backup tool, so that you can more easily find information about the capabilities of the tool you are using. See [Database backup and recovery](#).

1.9.10 Miscellaneous Changes and Enhancements

SQL Anywhere 17.0 a variety of general enhancements.

Miscellaneous New Features

Support added for database-scope variables (database upgrade required)

Previously, you could only create connection-scope variables. Now you can create database-scope variables. Database-scope variables are a great way to share values between connections and their initial values at declaration time persist after a database restart. Their intended use is to store small, infrequently changing, shared values. Storing large or frequently changing values may affect the performance of your application, and is not recommended.

Several new features have been added to the software to support database-scope variables.

Changes to the CREATE VARIABLE statement

The syntax has now includes an optional DATABASE keyword to create database-scope variables.

See [CREATE VARIABLE statement](#).

Changes to the DROP VARIABLE statement

A new syntax has been added for dropping database-scope variables.

See [DROP VARIABLE statement](#).

New system privileges

Four new system privileges have been added:

- CREATE DATABASE VARIABLE system privilege
- MANAGE ANY DATABASE VARIABLE system privilege
- SELECT PUBLIC DATABASE VARIABLE system privilege
- UPDATE PUBLIC DATABASE VARIABLE system privilege

See [System privileges](#).

New SYSDATABASEVARIABLE system view

The SYSDATABASEVARIABLE system view and ISYSDATABASEVARIABLE system table contain information about all database-scope variables. See [SYSDATABASEVARIABLE system view](#).

Changes to the VAREXISTS function

The VAREXISTS function supports a new parameter for specifying the owner of a database-scope variable. See [VAREXISTS function](#).

Changes to the DROP VARIABLE, SET, and UPDATE statements

The DROP VARIABLE, SET, and UPDATE statements now support the specification of the owner of the database-scope variable. See [DROP VARIABLE statement](#), [SET statement](#), and [UPDATE statement](#).

Enhancements to the variables documentation

The documentation for variable support in the product has been simplified and improved. See [Variables](#).

Enhancements to remote servers (database upgrade required)

- Support for altering a remote server to be read-only, using the new READ ONLY clause of the ALTER SERVER statement.
The CREATE SERVER statement has also been extended to support the same syntax as the READ ONLY clause in the ALTER SERVER statement, while also maintaining backwards compatibility with previous versions of the software.
- Support for specifying a default login when creating remote servers (new DEFAULT LOGIN clause). If a user does not have an external login, the default login information is used for the connection instead.
- Support for an optional REMOTE keyword in the CREATE SERVER, ALTER SERVER, and DROP SERVER statements (for example, CREATE REMOTE SERVER).

See [CREATE SERVER statement](#), [ALTER SERVER statement](#), and [DROP SERVER statement](#).

Support added for automatic commit on the database server

Automatic commit can now be enabled on the database server. In previous versions, if enabled, automatic commit was performed by client software. Now the client can request that the server commit requests automatically. This results in a performance improvement. See [auto_commit option](#).

Support added for pivoting and unpivoting table expressions

You can now pivot and unpivot table data using two clauses, PIVOT and UNPIVOT, in the FROM clause of a query to create pivoted- or unpivoted-derived tables. Pivoting rotates column data into rows and aggregates data in a meaningful way for your business needs. Unpivot is a similar operation, but rotates row data into columns. Unpivoting is useful for normalizing un-normalized data, such as several columns of similar data that must be joined with other data. See [PIVOT clause](#), [UNPIVOT clause](#), and [Tutorial: Pivoting table data](#).

Enhancements to creating and altering directory access servers (database upgrade required)

Previously, you specified a READONLY clause in the USING string when creating or altering a directory access server to control whether directory access is read-only. Now, a READ ONLY clause is provided for use independent of the USING clause. The previous READONLY syntax is still supported, but it is deprecated and may be removed in a future release. Additionally, you can now control whether an external login (externlogin) is required to access a directory access server using the new ALLOW { SPECIFIC | ALL } USERS clause in your CREATE SERVER or ALTER SERVER statement. See [CREATE SERVER](#), and [ALTER SERVER statement](#).

Support added for reading and setting multiple HTTP headers with the same name

You can now specify the instance of the HTTP header that you want to read or set. The following changes have been made to support this feature:

- The HTTP_HEADER function now supports the `instance` parameter. See [HTTP_HEADER function](#).
- The HTTP_RESPONSE_HEADER function now supports the `instance` parameter. See [HTTP_RESPONSE_HEADER function](#).
- The sa_set_http_header system procedure now supports the `instance` parameter (database upgrade required). See [sa_set_http_header system procedure](#).
- The sa_http_header_info system procedure may return multiple rows with the same name if the request contains multiple HTTP headers with the same name (database upgrade required). See [sa_set_http_header_info system procedure](#).

To access the new functionality of the HTTP_HEADER and HTTP_RESPONSE_HEADER functions, use a SQL Anywhere version 17 database server. Accessing the new functionality of the sa_set_http_header system procedure and the sa_http_header_info system procedure requires a database upgrade or rebuild.

Support added for HTTP OPTIONS method

The OPTIONS method is now supported in HTTP web services, stored procedures, and functions. Among other things, this feature allows cross-origin resource sharing (CORS) to be supported by web services. See [CREATE SERVICE statement \[HTTP web service\]](#), [CREATE PROCEDURE statement \[Web service\]](#), and [CREATE FUNCTION statement \[Web service\]](#).

Support added for cache warming to a steady state (database upgrade required)

To improve performance, you can record cache contents during a steady state period and restore this state when necessary. See [Cache warming](#).

The following changes have been made to support this feature:

New SQL statements

- ALTER DATABASE SAVE CACHE statement
- ALTER DATABASE RESTORE CACHE statement
- ALTER DATABASE DROP CACHE statement

See [ALTER DATABASE statement](#).

New system procedures

- sp_db_cache_contents system procedure. See [sp_db_cache_contents system procedure](#).
- sp_read_db_pages system procedure. See [sp_read_db_pages system procedure](#).

CONTAINS search conditions are now allowed over unflattenable query expressions

CONTAINS search conditions are now allowed in the WHERE clause of a query over unflattenable query expressions such as views or derived tables. See [CONTAINS search condition](#).

Support added for the Native Component Supportability (NCS) library

On Microsoft Windows and 64-bit Linux, the database server can now use the NCS (Native Component Supportability) library to send status events to the SAP Solution Manager. See [-ncs database server option](#) and [-ncsd database server option](#).

Enhancements to initial values for variables

Previously, when creating or declaring variables, you could only initialize the variable using a constant expression, a special value, or a built-in function of constant expressions. Now you can specify any arbitrary expression such as other variables, user-defined functions, or subqueries. See [CREATE VARIABLE statement](#), and [DECLARE statement](#).

Changes to trigger firing order when ORDER is not specified

When you define multiple triggers on a column, if more than one trigger specifies the same event type, called overlapping event types, then you must give each trigger a unique order in which you want the database server to fire it. For example, a trigger with UPDATE and INSERT event types, and a trigger with an UPDATE OF event type, are considered to have overlapping event types and therefore each trigger requires that a unique firing order be specified.

Failure to specify an order in these cases means that the database server fires the triggers in an implementation-specific way that is not always predictable, and is subject to change from release to release. *A change to the implementation-specific order has been introduced in this release.*

Because changes like this one can be introduced, order triggers that have overlapping event types. This practice guarantees a predictable firing order for triggers with overlapping event types in future releases.

UPDATE and UPDATE OF event type triggers require a unique ordering, just as two UPDATE event type triggers would. See [Transact-SQL triggers](#).

UNIX and Linux systems now use POSIX semaphores

Previously, UNIX and Linux systems used SYSV semaphores, which limited connections to database servers. Now, all UNIX and Linux systems, with the exception of macOS and Solaris, use POSIX semaphores.

macOS does not support POSIX semaphores and Solaris uses its own semaphores, which act the same as POSIX semaphores.

Improved security for UNIX and Linux shared memory

To improve security, when UNIX or Linux maps files on disk into memory, the files are now removed from the file system immediately following creation. Previously, these files were accessible to anyone with permission to access the SATMP location.

Improvements to detecting CPU topology

The database server now more accurately detects CPU topology on Linux, particularly when running in a virtual machine.

Server Licensing utility (dblic) now supports core-based licensing The Server Licensing utility (dblic) now supports core-based licensing in addition to per-seat licensing. Processor-based licensing is deprecated. See [Server Licensing utility \(dblic\)](#).

Service utility (dbsvc) for Linux supports systemd services The Service utility (dbsvc) for Linux now supports the systemd service interface. The -i, -li, and -lt options have been added to the utility in support of this new feature. This utility now supports spaces and other special characters in the service name, service arguments, and SQL Anywhere installation path. See [Service utility \(dbsvc\) for Linux](#).

Database, Connection, and Database Server Options, Parameters, and Properties

For descriptions of the options and properties mentioned in this section, visit the appropriate link below:

- [List of database options](#)
- [List of database properties](#)
- [Database server options](#)
- [List of database server properties](#)
- [Connection parameters](#)
- [List of connection properties](#)
- [Network protocol options](#)

The following enhancements have been made to database, connection, and database server options, parameters, and properties:

-ufd database server option improvement

To provide a more flexible means of handling database assertion failures, the -ufd dbsrv17 database server option has been extended with two new options:

- restart_escalate
- restart_abort

New auto_commit option

When the auto_commit option is set to ON, the database server automatically commits after every request.

For APIs, such as Embedded SQL, where autocommit was not previously supported, these applications can now use the `auto_commit` option to have the database server automatically commit after each execution.

New database options

- `max_connections` database option
- `connection_type` database option
- `time_zone` database option

New database server options

- `-uq` database server option

New connection parameters

- Enhancement to `CommBufferSize (CBSIZE)` connection parameter

New database properties

- `TimeWithoutClientConnection` database property
- `BackupInProgress` database property
- `MaxConnections` database property
- `Timezone` database property
- `CurrentTimezoneOffset` database property

New database server properties

- `HasSecureFeatureKey` database server property
- `HasSecuredFeature` database server property

New connection properties

- `HasSecuredFeature` connection property
- `connection_type` connection property
- `max_connections` connection property
- `time_zone` connection property

New network protocol options

- `LogRename (LRENAME)` protocol option

1.9.11 Behavior Changes, Deprecated Features, and Features That Are No Longer Supported

SQL Anywhere 17.0 includes changes to behavior. Some features have been deprecated or are no longer supported in this release.

Behavior Changes

Running Interactive SQL (dbisql) as a console application

In SQL Anywhere 17, you can run Interactive SQL (dbisql) as a console application. Results of executed statements are printed to the command window. The format of these printed result sets may be different from how they appeared in earlier versions of dbisql, depending on your preference settings.

In version 16 and earlier, the width of columns could be based either on the data in the column or on the declared width of the column, depending on whether the result set had been cached by dbisql. Starting in version 17, character column widths are always based on the data they contain, or the width of the column label, whichever is wider. Only the data in the first 50 rows is used to determine the width of the column.

If you require result sets to be displayed using specific column widths, use the OUTPUT statement with the FORMAT FIXED option and user-supplied column widths.

Default value for the isql_command_timing option changed In previous releases, the default value for the isql_command_timing option was Off. The default value is now On. See [GET_IDENTITY function \[Miscellaneous\]](#).

Indexes no longer migrated with SAP HANA tables When SAP HANA tables are migrated to SQL Anywhere, indexes are not migrated along with them and must be created manually after the migration.

System views no longer display password hashes

Previously, system views allowed users to view password hashes. Now, access to password hashes is controlled using the following methods:

- The new ACCESS USER PASSWORD system privilege required for accessing some views
- New restricted system views
- A change to how values are displayed in password-related columns

Spatial API: the ST_TransformGeom method now supports the upgraded PROJ.4 library

With the upgraded library, results of particular projections will include new values that are more accurate than the values that resulted from projections calculated with the older PROJ.4 library.

Previously, the PROJ.4 library accepted no_defs as a transform parameter, but now the parameter must be specified as +no_defs. Ensure that any code or SRS definitions are updated.

The upgraded PROJ.4 library supports vertical data. Projections using a +to_meter cause Z coordinates to convert to meters using the provided scale factor. You can specify a different conversion factor for the Z coordinate by using the new +vto_meter setting. To avoid conversions of vertical data, set +vto_meter=1.0.

UPPER, UCASE, LOWER, and LCASE functions now return LONG (N)VARCHAR - upgrade or rebuild required

Previously, when the UPPER, UCASE, LOWER, and LCASE functions were called on NCHAR data, or on CHAR data in a database that used the UCA collation, the data was described as having the same length as the input. Now, when calling these functions, LONG NVARCHAR is returned for NCHAR data and LONG VARCHAR is returned for CHAR data in a database that uses the UCA collation. See [UPPER function](#), [UCASE function](#), [LOWER function](#), and [LCASE function](#).

Recreate and recompile materialized views that call these functions, and recompile any views and procedures that call these functions.

HTTP and HTTPS connections without associated session IDs are ignored when determining whether or not to shut down the database server

Previously, all HTTP and HTTPS connections were considered when determining whether the conditions have been met for shutting down the database server. Now, when determining whether or not to shut down

the database server, only HTTP and HTTPS connections with associated session IDs are considered; HTTP and HTTPS connections without session IDs are ignored. This behavior change has the following effects:

- When clicking *Shut down* on the database server messages window, the warning window no longer appears if there are only HTTP and HTTPS connections without associated session IDs.
- Using the dbstop utility to conditionally stop a database server now succeeds if there are only HTTP and HTTPS connections without associated session IDs.
- Using the STOP SERVER statement to conditionally stop a database server now succeeds if there are only HTTP and HTTPS connections without associated session IDs.

GET_IDENTITY function now requires privileges

Previously you did not need any privileges on a table to use the GET_IDENTITY function to return and reserve an identity value. Now, you must have INSERT privilege on the table to use the function. See [GET_IDENTITY function \[Miscellaneous\]](#).

Change to UtilCmdsPermitted connection property The UtilCmdsPermitted connection property now returns an empty string for all connections except the current connection.

Changes to the Unload utility (dbunload)

Previously, you needed the SELECT ANY TABLE and SERVER OPERATOR privileges to perform an unload with reload operation (the -ac, -an, and -ar options) using the Unload utility (dbunload). Now, in addition to these system privileges, you need the new ACCESS USER PASSWORD system privilege introduced in this release.

Previously, when performing an unload *without* reload (that is, you do not specify -ac, -an, or -ar), passwords were unloaded by default. Now, you must specify the -up option to unload passwords, and you must have the ACCESS USER PASSWORD system privilege to do so. See [Unload utility \(dbunload\)](#).

Change to customized max_connections database option

A new max_connections database option has been introduced. If you already have a user-defined max_connections database option, it continues to use the user-defined behavior. To use the behavior of the new max_connections database option, you must upgrade your database to version 17.0. If the max_connections database option is set during an upgrade, the upgrade fails. The current database server connection limits are maintained. See [max_connections option](#).

Catalog changes related to securing password information in system views - upgrade required

The following views are new or have been changed to support improvements for hiding sensitive information such as password hashes.

View name	Change description
SYSSYNC	No change to the view, but you must now have the new ACCESS USER PASSWORD system privilege, as well as the SELECT ANY TABLE system privilege to select from this system view. See SYSSYNC system view .

View name	Change description
SYSSYNC2	Previously, the option and server_connect columns displayed sensitive data such as password hashes if the user had SELECT ANY TABLE system privilege. Now, these columns display three asterisks (***) if a value is stored, or a NULL if no value is stored. Additionally, the view now includes the server_protocol column found in the SYSSYNC system view. See SYSSYNC2 consolidated view .
SYSSYNCS	Previously, this view selected from the ISYSSYNC and ISYSPUBLICATION system tables. Now, it selects from the SYSSYNC2 and SYSPUBLICATION system views, where sensitive data such as password hashes, is suppressed. See SYSSYNCS consolidated view .
SYSSYNCSUSERS	Previously, this view selected from the ISYSSYNC system table. Now, it selects from the SYSSYNC2 system view, where sensitive data such as password hashes, is suppressed. See SYSSYNCSUSERS consolidated view .
SYSSYNCPUBLICATIONDEFAULTS	Previously, this view selected from the ISYSSYNC and ISYSPUBLICATION system tables. Now, it selects from the SYSSYNC2 and SYSPUBLICATION system views, where sensitive data such as password hashes, is suppressed. See SYSSYNCPUBLICATIONDEFAULTS consolidated view .
SYSSYNCSUBSCRIPTIONS	Previously, this view selected from the ISYSSYNC, ISYSUSER, and ISYSPUBLICATION system tables. Now, it selects from the SYSSYNC2, SYSSYNCSUSERS, and SYSPUBLICATION system views, where sensitive data such as password hashes, is suppressed. See SYSSYNCSUBSCRIPTIONS consolidated view .
SYSUSERPERM	The password column was previously BINARY(128) but is now CHAR(3). Also, the password column previously displayed password hashes, but now displays asterisks to indicate when a password is stored, and NULL if no password is stored. Use the SYSUSERPASSWORD system view to access the password hashes. See SYSUSERPERM compatibility view (deprecated) , and SYSUSERPASSWORD system view .
SYSUSERAUTH	This view selects from the SYSUSERPERM system view, and is changed by virtue of the changes to that system view (password hashes are now hidden). See SYSUSERAUTH compatibility view (deprecated) .

View name	Change description
SYSUSER	The password and dual_password columns were previously BINARY(128) but are now CHAR(3). Also, the columns previously displayed password hashes, but now display asterisks to indicate when a password is stored, and NULL if no password is stored. Use the SYSUSERPASSWORD system view to access the password hashes. See SYSUSER system view , and SYSUSERPASSWORD system view .
SYSUSERPASSWORD	NEW. Use this view to access the password hashes stored for login IDs now that password hashes are no longer displayed in the SYSUSER system view. See SYSUSERPASSWORD system view .
SYSEXTERNLOGIN	The remote_password column was previously VARBINARY(1024) but is now a CHAR(3). Also, the column previously displayed password hashes but now displays asterisks to indicate when a password is stored, and NULL if no password is stored. See SYSEXTERNLOGIN system view .
SYSEXTERNLOGINPASSWORD	NEW. Use this view to access the password hashes stored for remote logins, now that password hashes are no longer displayed in the SYSEXTERNLOGIN system view. See SYSEXTERNLOGINPASSWORD system view .
SYSLDAPSERVER	The ldsrv_access_dn_pwd column was previously VARBINARY(1024) but is now a CHAR(3). Also, the column previously displayed password hashes but now displays asterisks to indicate when a password is stored, and NULL if no password is stored. See SYSLDAPSERVER system view .
SYSLDAPSERVERPASSWORD	NEW. Use this view to access the password hashes stored for LDAP servers now that password hashes are no longer displayed in the SYSLDAPSERVER system view. See SYSLDAPSERVERPASSWORD system view .
SYSSYNCPROFILE	You must now have the SELECT ANY TABLE and ACCESS USER PASSWORD system privileges to query this system view. See SYSSYNCPROFILE system view .
SYSSYNCPROFILE2	NEW. This view is the same as the SYSSYNCPROFILE system view, except that it does not have the profile_defn column, which can contain sensitive password information. No privileges are required to access this system view. See SYSSYNCPROFILE2 system view .

Deprecated Features

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

Support for 2048 (2K) byte database page size deprecated

The 2048 byte value for database page size is deprecated and no longer recommended. The 2048-byte page size is still supported but may be removed in a future release. See [CREATE DATABASE](#), [Unload utility \(dbunload\)](#), [Initialization utility \(dbinit\)](#), and [-gp database server option](#).

sa_make_object system procedure is now deprecated

Previously, you used the sa_make_object system procedure in scripts to create or modify objects in the database. This approach has been deprecated in favor of using a CREATE statements with an OR REPLACE clause instead. Using CREATE OR REPLACE is more efficient, and offers the correct behavior when trying to create an object that already exists. See [sa_make_object system procedure \(deprecated\)](#).

Removed Features

Support for SQL Anywhere Server on Windows Mobile discontinued

SQL Anywhere Server is no longer supported on Windows Mobile. However, UltraLite is still supported on Windows Mobile.

TLS and HTTPS connections no longer support the RC4-MD5 cipher

The RC4-MD5 cipher is no longer supported for TLS and HTTPS communications.

Discontinued: CONNECTION CLOSE clause, ALTER SERVER statement

Support for the CONNECTION CLOSE clause of the ALTER SERVER has been removed. Use the DROP REMOTE CONNECTION statement instead. See [ALTER SERVER statement](#).

The -sm database option is no longer supported

The -sm database option, which was deprecated in earlier versions, is no longer supported.

All -xp database option values other than ON are no longer supported

The following values for the -xp database option, which were deprecated in previous versions, are no longer supported:

- partner-conn
- auth-str
- arbiter-conn
- mode
- autofailover
- pagetimeout
- preferred

See [-xp database option](#).

1.10 What's New in MobiLink Version 17.0

MobiLink version 17.0 introduces several new, changed, deprecated, or removed features.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

MobiLink Server

Support for the Native Component Supportability (NCS) library

The MobiLink server can now use `ncs.conf` files to connect to an SAP Diagnostics Agent. See [-ncs mlsvr17 option](#), [-ncsd mlsvr17 option](#), and [-ncsp mlsvr17 option](#).

Support for SAP Passports

The MobiLink server can now use SAP Passports for tracing. See [MobiLink server logging and SAP Passport](#).

Support for SAP IQ

The MobiLink server now supports consolidated databases running on SAP IQ 16.0 and higher servers.

i Note

This enhancement was first released in a version 16 Support Package.

Support for Oracle 12.1 consolidated databases

The MobiLink server now supports Oracle 12.1 consolidated databases.

To take advantage of Oracle 12.1 features with SQL Anywhere 17, the SQL Anywhere - Oracle ODBC driver must be greater than or equal to build number 2087 and the Oracle OCI library must be installed from the Oracle 12.1 installation image.

Support for SAP ASE 16.0 consolidated databases

The MobiLink server now supports SAP ASE 16.0 consolidated databases.

Support for Microsoft SQL Server 2014 consolidated databases

The MobiLink server now supports Microsoft SQL Server 2014 consolidated databases.

Support for MySQL 5.6.20 consolidated databases

The MobiLink server now supports MySQL 5.6.20 consolidated databases.

Support for IBM DB2 10.5 consolidated databases

The MobiLink server now supports IBM DB2 10.5 consolidated databases.

Changes to MobiLink support for LDAP authentication

The following changes have been made to the way MobiLink supports LDAP user authentication:

- Settings for the `ldap_failover_to_std` parameter of the `ml_add_user_auth_policy` system procedure have been extended to be 0, 1, or 2. See [ml_add_user_auth_policy system procedure](#).
- The `-zup` MobiLink server option allows you to specify a default authentication policy name for user authentication against an LDAP server. See [-zup option](#).
- The MobiLink user password is only hashed and stored in the `ml_user` table in the consolidated database if the `ldap_failover_to_std` parameter is configured with a value of 1 or 2. The password is not saved if this parameter is set to 0. See [ml_add_user_auth_policy system procedure](#).

MobiLink Clients

New options for restartable download

Use the `-kpd dbmsync` option or the `KeepPartialDownload` synchronization profile option to save information that is required to allow a failed download to be restarted. In previous releases, when a download failed, `dbmsync` always attempted to save the information required to restart a download. Now, when a download fails, `dbmsync` only attempts to save the information required to restart the download if either the `-kpd dbmsync` option or the `KeepPartialDownload` synchronization profile option is specified. See [-kpd dbmsync option](#), and [KeepPartialDownload synchronization profile option](#).

dbmsync provides the ability to restart downloads when no bytes of data have been received The `dbmsync` utility now allows you to restart a failed download even if no bytes of the download have been received. Previously a download could only be restarted if at least one byte had been received. See [MobiLink client utility \(dbmsync\) syntax](#).

i Note

This enhancement was first released in a version 16 Support Package.

dbmsync offline transaction log retrieval has been extended

The `dbmsync` utility can now retrieve offline transaction logs from the SQL Anywhere database server instead of accessing them directly. If offline transaction logs are required but the given offline transaction log directory cannot be opened or it does not contain offline transaction log files, then `dbmsync` retrieves the offline transaction logs through the database server. The following restrictions apply:

- The user ID that is used by `dbmsync` to connect to the synchronization database must have the `READ FILE` and `WRITE FILE` privileges and all the offline transaction log files must be in the online transaction log directory.
- The SQL Anywhere database server must have Support Package build number 1691 or later to support this feature.

There is a slight performance penalty when using this feature because the database server must do more work to retrieve the pages. If performance is critical, then using the `dbmsync OfflineDirectory` extended option may be best for your deployment. See [MobiLink client utility \(dbmsync\) syntax](#).

i Note

This enhancement was first released in a version 16 Support Package.

Improved authentication

When making a TLS or HTTPS connection, if none of the `certificate_name`, `certificate_company`, or `certificate_unit` protocol options are set, then the host name of the database server is verified against the host name of the database server certificate. To prevent this host name check, set the new `skip_certificate_name_check` protocol option to ON when connecting with TLS or HTTPS. See [skip_certificate_name_check protocol option](#).

New skip_certificate_name_check protocol option

Use the `skip_certificate_name_check` protocol option to control whether the client library skips the check of the server host name against the database server certificate host names. See [skip_certificate_name_check protocol option](#).

New allow_expired_certs MobiLink client network protocol option

Use the `allow_expired_certs` protocol option to accept a server certificate that has either expired or is not yet valid and continue with the synchronization.

MobiLink Plug-in for SQL Central

Test Window Configuration window is now configurable

Use the [Test Window Configuration](#) window to configure some of the options used for testing a synchronization model, enabling more complete testing of your synchronization logic and creating an environment closer to that of your production environment. See [Configuring synchronization model testing](#).

Assign a user authentication policy to multiple users

You can now assign a user authentication policy to multiple users at the same time. See [Assigning a user authentication policy to multiple users](#).

Synchronization models can be duplicated The [Synchronization Models](#) menu in the MobiLink plug-in has a new option to duplicate a synchronization model. See [Synchronization models](#).

i Note

This enhancement was first released in a version 16 Support Package.

Changes to MobiLink plug-in support for LDAP authentication

The following changes have been made to the way MobiLink plug-in supports LDAP user authentication:

- The MobiLink plug-in now supports calling the standard MobiLink authentication scripts for LDAP authentication. You can specify *Never*, *Always* or *If LDAP server unavailable* for LDAP authentication on the [LDAP Servers](#) page of the [New User Authentication Policy Wizard](#) and on the [General](#) tab of the [Authentication Policy Properties](#) window.
- The `-zup` MobiLink server option is supported. The `-zup` option allows you to specify a default authentication policy name for user authentication against an LDAP server. This option can be accessed from the [Advanced](#) tab of the [Server Command Line Properties](#) window. See [-zup option](#).

Enhancement to Oracle support

The `dbmlsrv17 -dt` option now supports Oracle consolidated databases. See [-dt mlsrv17 option](#).

MobiLink Behavior Changes, Deprecated Features, and Features That Are No Longer Supported

MobiLink support for AIX

MobiLink support for IBM AIX is deprecated. While you can continue to run MobiLink on IBM AIX, support for IBM AIX will be removed in a future release.

MobiLink support for DB2

MobiLink support for DB2 consolidated databases is deprecated. While you can continue to use DB2 as a MobiLink consolidated database, support for DB2 will be removed in a future release.

Support for `-cmax`, `-cmin`, and `-vk` options has been removed from the MobiLink plug-in

The *Server command-line properties* Line no longer supports the `-cmax`, `-cmin`, and `-vk` MobiLink server options. If an existing command line with one of these options is loaded, these options are removed.

Support for SAP HANA consolidated databases

Previous releases of MobiLink supported SAP HANA consolidated databases. Now, to synchronize SQL Anywhere clients with an SAP HANA consolidated database, use SAP HANA remote data sync. All features except for server-initiated synchronization and the arbiter are supported (the HANA Platform has its own tools for high availability).

MobiLink server API package names have changed

The package name for the MobiLink server Java API has been changed to:

```
com.sap.ml.script
```

The package name for the MobiLink server .NET API has been changed to:

```
Sap.MobiLink.Script
```

`mlrsa_tls17.dll` and `mlrsa_tls_fips17.dll` have been replaced

The MobiLink server no longer uses `mlrsa_tls17.dll` or `mlrsa_tls_fips17.dll`. Instead it uses `dbrsa17.dll` or `dbfips17.dll`. See [Windows 64-bit applications](#).

Support for `-cmax`, `-cmin`, `-cm`, and `-vk` MobiLink server options has been removed

The MobiLink server no longer supports the `-cmax`, `-cmin`, `-cm`, and `-vk` options. They are ignored if specified.

MobiLink and UltraLite no longer support the Microsoft ActiveSync provider

The Microsoft ActiveSync utility is not supported for MobiLink or UltraLite.

The `-oe` option has been removed

The `-oe` option of the `mlsrv17 -x` option is no longer supported.

Integrated Outbound Enabler support removed The Integrated Outbound Enabler for MobiLink has been removed. You can no longer use the `-x oe` option with `mlsrv17`. Use the normal, standalone Relay Server Outbound Enabler instead.

User authentication classes for POP3 and MAPI have been removed

The following convenience classes for MobiLink user authentication via POP3 and MAPI have been removed:

- `ianywhere.ml.authentication.IMAP`
`ianywhere.ml.authentication.POP3`

These classes appeared in `authenticate_user` event scripts that were generated by using the MobiLink plug-in in SQL Central. If your MobiLink model includes such generated `authenticate_user` event scripts, then you must replace them with manually created `authenticate_user` event scripts. The generated `authenticate_user` event scripts are automatically deleted when you open your MobiLink model in SQL Central.

Related Information

[SAP Supported Platforms and Engineering Support Status](#) 

[Recommended ODBC drivers for 17.0.0 MobiLink](#) 

1.11 What's New in UltraLite Version 17.0

UltraLite version 17.0 introduces new, changed, deprecated, or removed features.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

General Changes

PKCS #12 encoded identity support

The *identity-file* option for the `ulinit` and `ulload` utilities support PEM and PKCS12 identities.

Synchronization observer states have been added

The `UL_SYNC_STATE_RECEIVING_UPLOAD_ACK` state has been removed and several new states have been added.

UltraLite.NET

`ULIndexSchema.Close()` has been added to improve memory management

`ul_sync_state` This method closes a `ULIndexSchema` instance in an UltraLite.NET application.

UltraLiteJ

CustDB sample project for Android Studio IDE

Android Studio is the official IDE for Android devices. An Android Studio project has been added for the CustDB sample.

Java package name changes

The package name for the Java classes is now `com.sap.ultralitejni17`. The package name for the CustDB sample is now `com.sap.custdb`.

Platforms and Devices

Support for Windows Phone 8.1 and Windows 8.1 Store Apps

UltraLite for WinRT API supports Windows 8.1 Store Apps and Windows Phone 8.1 applications. An UltraLite CustDB sample for WinRT supports Microsoft Visual Studio 2013 and Windows Phone 8.1.

Support for x86 Linux

UltraLite supports the x86 Linux platform.

Support for Android on ARM64

UltraLite supports the Android platform running on the 64-bit ARM processor.

Behavior Changes, Deprecated Features, and Features That Are No Longer Supported

BlackBerry support has been dropped from UltraLiteJ BlackBerry and J2SE interfaces have been removed.

Microsoft ActiveSync provider support has been dropped

Microsoft ActiveSync provider support for UltraLite is no longer available.

Increase to the maximum publication article size

Publication articles up to 2048 bytes are now supported.

Create a new database or rebuild existing ones to access the larger publication article size. Older databases continue to use a maximum publication article size of 256 bytes.

See [sysarticle system table](#).

Increase to the maximum database file size

The maximum UltraLite database file size has been increased from 4 GB to 16 GB for page sizes of 4 KB, 8 KB, and 16 KB. For databases with a page size of 2 KB, the limit has only been increased to 8 GB, and for 1 KB database pages, the size limit has been decreased to 1 GB.

UltraLite Embedded SQL (ESQL) samples have been removed The CustDB for Embedded SQL, ESQLAuth, and ESQLSecurity samples have been removed.

UltraLite now supported on 32-bit Linux UltraLite now supports the 32-bit Linux platform.

New UltraLite kdf_iterations creation option `kdf_iterations` is the number of iterations, in thousands, of the key derivation function which converts the pass phrase provided by the `DBKEY` parameter into an actual encryption key. This function makes it more difficult to access an encrypted database by prolonging each attack attempt.

See [Ultralite kdf_iterations creation option](#).

1.12 What's New in Relay Server Version 17.0

Relay Server version 17.0 introduces several new, changed, deprecated, or removed features.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

New Features

Relay Server performance enhancements

Several enhancements have been made to improve performance:

- The Relay Server now supports the WebSocket protocol for Apache and Microsoft IIS 8 (earlier versions of IIS do not support WebSocket traffic). Applications that support the WebSocket protocol use it automatically. Otherwise, the Relay Server uses HTTP.
- The Outbound Enabler now uses junction pools to send requests to and from the Relay Server and for dedicated backend connections. The `rsoc2` utility supports the following options to control the size of the junction pool: `-jsl`, `-jsh`, and `-jl`. [Relay Server Outbound Enabler syntax](#)
- There is a new setup script for deploying Relay Server to Microsoft Windows (`iis7_plus_setup.bat`). [Deploying the Relay Server components to Microsoft Windows Server](#)

Enhancements to Microsoft IIS support

- You can now run Relay Server on Microsoft IIS 8.5.
- The thread pool for the Relay Server on Microsoft IIS is now self governed. You can use the default web garden size of 1 and use the `min_thread` and `max_thread` properties in the Options section of the Relay Server configuration file to control the Relay Server thread pool. [Configuring a Relay Server](#)

Relay Server on Apache supports dual Apache server instances

Relay Server supports configuring and running two Apache server instances. One instance of the Apache server is client facing and only services client requests. The other instance is server facing and only services Outbound Enabler requests. Running Apache Relay Server in this configuration provides better scalability in large deployments.

Relay Server log file size configuration

Limit the size of the Relay Server log file by using the `log_size_limit` option in the Options section of the Relay Server configuration file. [Configuring a Relay Server](#)

Configuration file monitoring on Microsoft IIS

On Microsoft IIS, the Relay Server now continuously monitors the configuration file and reports changes in status in the log file.

Automatic configuration of the backend farm and backend server

The new `auto_config` option instructs the Relay Server to automatically configure the backend farm and backend server. This feature is only available on Windows. [Configuring the Relay Server automatically](#)

Behavior Changes, Deprecated Features, and Features That Are No Longer Supported

Compatibility with previous versions

The version 17 Relay Server does not work with version 16 and earlier of the Outbound Enabler. As well, version 17 of the Outbound Enabler does not work with version 16 and earlier of the Relay Server.

Changes to the `rsoe` utility

- For HTTPS connections, you must now specify at least one of `certificate_name`, `certificate_company`, or `certificate_unit` to ensure that the Outbound Enabler is connecting to the correct backend server. To prevent checking the certificate, specify the `skip_certificate_name_check` option.
- The `-cs status_url` option has been removed.
- The utility has been renamed `rsoe2`.
- When you connect the Outbound Enabler to the backend server using TLS or HTTPS, specify at least one of the following protocol options: `certificate_name`, `certificate_company`, or `certificate_unit`. The Outbound Enabler checks these values against the backend server certificate to ensure that it is connecting to the correct backend server. To prevent this checking, set the new `skip_certificate_name_check` protocol option to true. [Relay Server Outbound Enabler syntax](#)

AdminChannel tool changes

`ianywhere.ml.rs.AdminChannel` has been renamed `com.sap.relayserver.AdminChannel`.

See [Remote administration using AdminChannel \(Microsoft Windows\)](#).

SQL Anywhere Server Monitor support

The SQL Anywhere Server Monitor no longer supports Relay Server. Only the status page is supported.

File name changes

Old file name	New file name
<code>rs-setup.bat</code>	<code>iis7_plus_setup.bat</code>
<code>rsoesuppXX</code>	This file has been removed
<code>rsoe.exe</code>	<code>rsoe2.exe</code>

Old file name	New file name
ianywhere.ml.rs Java package qualifier	com.sap.relayserver
rs_admin.dll, rs_client.dll,rs_monitor.dll,and rs_server.dll	rs.dll (all combined into a single DLL)

Change to timeout behavior

The Relay Server and the Outbound Enabler no longer time out client traffic according to the IAS-RS-App-Timeout-Minute header. Instead, the timeout behavior is controlled by the client, other intermediaries, the backend server, and the web server that contains the Relay Server. The default 8 minute timeout for IAS-RS-App-Timeout-Minute no longer applies.

Change to the end of the Relay Server record

The end of the Relay Server record no longer ends with

```
<processId.threadId.F (BEFarmIdx) B (BEServerIdx) S (BESessionNum) R (RequestIdx) >
```

It now ends with

```
<label: J{relayserver-host#backend-farm-name#backend-server-name#junction-index} R{request-number}>
```

Changes to Microsoft IIS support

- Run Relay Server on Microsoft IIS version 7.0 or later. Support for version 6.0 has been removed.
- The Relay Server now starts on demand when the web service is running. Shutting down the web service also shuts down the Relay Server. As a result of this change, the Relay Server for Microsoft IIS no longer runs on shared memory, and it now uses a single application pool with a web garden size of one.

The rshost service is no longer available to administer Relay Server on Windows, and the dbsvc -t RSHOST option has been removed. On Windows, rshost is only available for internal state management. For Windows, use the AdminChannel class in rstool.jar to administer Relay Server locally and remotely. You can still use rshost on Linux.

As well, the Relay Server logs information to the following locations:

Log file	Location
Configuration file log	Relay-Server-DLL-directory/rs.config
Relay Server log	Relay-Server-DLL-directory/Log/rs.log

Relay Server plug-in removed

The Relay Server plug-in for SQL Central has been removed. Edit the Relay Server configuration file to change your Relay Server configuration.

Unsupported properties

Property type	Property	Notes
options section	shared_mem	This property is now ignored for Microsoft IIS. Supported for Apache.
	up_pad_size	Ignored by Microsoft IIS and Apache.
backend_farm section	client_security	The new behavior is the same as the old default (not restricted). Restrict via the web server configuration.
	backend_security	The new behavior is the same as the old default (not restricted). Restrict via the web server configuration.
	active_cookie	The new behavior is the same as the old default (yes to inject a standard cookie). The cookie is now server-level affinity only.
	active_header	Relay Server does not inject a proprietary header for affinity management.
	max_client_buffer	Flow and resource control is now managed via natural congestion on dedicated connections per client.
	renew_overlapped_cookie	This setting is no longer required because socket-level affinity is no longer available.
	socket_level_affinity	Socket-level affinity is no longer available. Relay Server transforms persistent HTTP to non-persistent HTTP downstream towards the back end. Relay Server client persistence over higher latency networks is still preserved. Server-level affinity via standard the cookie mechanism is the only supported behavior.

1.13 What's New in SQL Remote Version 17.0

SQL Remote version 17.0 includes enhancements and behavior changes.

Deprecated features will be removed in future versions. Alter your applications to use recommended feature replacements instead of relying on deprecated features.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

Enhancements

Enhancements to dbremote offline transaction log retrieval

The dbremote utility now retrieves offline transaction logs from the SQL Anywhere database server instead of accessing them directly. If offline transaction logs are required but the given offline transaction log directory cannot be opened or it does not contain offline transaction log files, then dbremote retrieves the offline transaction logs through the database server. The following restrictions apply:

- The user ID that is used by dbremote to connect to the replication database must have the READ FILE and WRITE FILE privileges and all the offline transaction log files must be in the online transaction log directory.

There is a slight performance penalty when using this feature because the database server performs extra work to retrieve the transaction logs. [SQL Remote Message Agent utility \(dbremote\)](#)

Behavior Changes

Changes to the SQL Remote Extract utility (dbxtract) and Extract Database Wizard

Previously, when extracting a database using the Extract utility (dbxtract) or the *Extract Database Wizard* in SQL Central, passwords were unloaded by default. Now, they are not. You must now specify whether to extract passwords (-up option in dbxtract, and a new UI control in the *Extract Database Wizard*), and you must have the ACCESS USER PASSWORD system privilege to extract them. See [Extraction utility \(dbxtract\)](#).

1.14 What's New in Previous Releases of SQL Anywhere

Use the following links to access previous releases of the SQL Anywhere documentation, where you can read about how and when features were added, enhanced, deprecated, and removed.

Version	Link to new features, behavior changes, and deprecated features
16.0.0	http://dcx.sap.com/sa160/en/sachanges/sa-16-nagano.html
12.0.1	http://dcx.sap.com/1201/en/sachanges/new1201.html
12.0.0	http://dcx.sap.com/1200/en/sachanges/newinnsbruck.html
11.0.1	http://dcx.sap.com/1101/en/sachanges_en11/new1101.html

Version	Link to new features, behavior changes, and deprecated features
11.0.0	http://dcx.sap.com/1101/en/sachanges_en11/newpanorama.html
6.0.1 to 10.0.1	http://dcx.sap.com/1001/en/dbwnen10/wn-new1001.html

Related Information

[Supported Platforms](#)

1.15 How to Upgrade to the Latest Version of SQL Anywhere

Major releases, service packs, and patch level updates include enhancements to the software that can require you to upgrade or rebuild your database.

Differences Between a Major Release, a Service Pack, and a Patch-Level Update

Major release

Major releases comprise all new features and fixes since the last major release. Major releases get a new major number (for example, 17.0 follows 16.0).

Service Pack (SP)

A service pack comprises bug fixes and new or changed features since the last service pack or major release. Service packs get a new minor number (for example 17.0.9 follows 17.0.8).

Patch-level Update (PL)

A patch-level update may contain bug fixes, performance improvements, and other revisions deemed important (for example, new security updates). The version number does not change but the build number does (for example 17.0.8.4087 follows 17.0.8.4075).

Compatibility with Existing Databases, Database Servers, and Administration Tools

For information, see the **SAP SQL Anywhere Supported Platforms and Engineering Support Status** page (<http://scn.sap.com/docs/DOC-35654>).

Compatibility with Existing Software

Version 17 SQL Anywhere database servers support connections from client applications using software from version 12 or later.

Version 17 SQL Anywhere clients can connect to version 12 and later SQL Anywhere database servers.

Version 17 SQL Anywhere database servers support connections from all supported TDS client versions; SAP Open Client version 15, 16, or later; and jConnect JDBC driver version 7 or later.

Version 17 MobiLink servers support connections from version 12 and later synchronization clients (UltraLite and MobiLink SQL Anywhere client utility (dbmlsync)). Version 17 synchronization clients can connect to version 16 and later MobiLink servers.

The `readme.txt` files for all of the SQL Anywhere support packages and patch-level upgrades can be found at <http://sqlasupport.sap.com/readme/index.html>.

In this section:

[SQL Anywhere Server Upgrades \[page 89\]](#)

Before using existing applications with this version of the software, review the list of behavior changes to determine whether your application is affected.

[MobiLink Upgrades \[page 125\]](#)

Before upgrading, check for behavior changes that may affect you and take standard upgrade precautions.

[UltraLite Upgrades \[page 138\]](#)

Before using existing UltraLite applications with this version of the software, be sure to review the list of new features and behavior changes to determine whether your application is affected.

[SQL Remote Upgrades \[page 142\]](#)

SQL Remote upgrades can be performed one site at a time and your upgrade must account for features that were available in older versions of SQL Remote that are not supported in the latest version.

[Upgrading the Monitor and Migrating Resources \[page 143\]](#)

Upgrade the monitor by installing the new software and migrating your resources. When upgrading the Monitor between major releases, use the software installer and follow the prompts to migrate your resources. When upgrading to a major release, install the software and run the Migrator utility to migrate the resources.

[Upgrading Version 16 Databases to Use the SQL Anywhere Cockpit \[page 145\]](#)

Create the COCKPIT_ROLE user-defined role that users require to connect to the Cockpit, and then grant users its exercise rights.

[Upgrading Version 17 Databases to Use the SQL Anywhere Cockpit \[page 146\]](#)

Grant the ACCESS DISK INFORMATION system privilege your version 17 users to allow them to have full functionality of the features in the version 17 Cockpit.

[Converting the OData Server from Version 16 \[page 148\]](#)

Use SQL statements to create ODATA PRODUCER definitions, and then use the `dbsrv17 -xs odata` protocol option to start the enabled and defined ODATA PRODUCER statements.

Related Information

[Supported Platforms](#)

[Unload Utility \(dbunload\)](#)

1.15.1 SQL Anywhere Server Upgrades

Before using existing applications with this version of the software, review the list of behavior changes to determine whether your application is affected.

i Note

Check the `readme.txt` file for additional information about the software, including feature information that was added after the documentation was completed.

In this section:

[Upgrades of Version 10 and Later Databases \[page 90\]](#)

If you are upgrading from version 10 or later, you can either upgrade or rebuild your database.

[Upgrades of Version 9 and Earlier Databases \[page 91\]](#)

If you are upgrading to version 17 from version 9 or earlier, you must rebuild the database, which consists of unloading the old database, and reloading it into a new version 17 database.

[User-defined Upgrades \[page 92\]](#)

In addition to performing system upgrades, you can define your own upgrade process by storing the SQL statements for a user-defined upgrade in a `.sql` script file.

[Upgrade and Rebuild Precautions \[page 93\]](#)

SQL Anywhere is used in many different configurations, and no upgrade guidelines can be guaranteed for all cases. There are several precautions that you must take before upgrading your software.

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

If you have multiple versions of SAP SQL Anywhere on your Windows computer, check your system path when using utilities. The software installation adds its executable directory to the end of your system path, so it is possible to install a new version and inadvertently run an older version.

[Rebuilding \(Unloading/Reloading\) a Database \(Command Line\) \[page 94\]](#)

Rebuild an older database to the latest version of the software.

[The Rebuild Process for Version 10 and Later Databases \[page 96\]](#)

Rebuilding a database consists of unloading and reloading the database to upgrade its file format. When you upgrade the file format, it changes the format used to store and access data on disk, letting you use all the new features and performance enhancements in the latest version of the software.

[The Rebuild Process for Version 9 and Earlier Databases \[page 100\]](#)

There are upgrade restrictions and considerations that are unique to rebuilding a version 9 or earlier database. Always back up a database before rebuilding it.

[The Upgrade Process for Version 10 and Later Databases \[page 107\]](#)

Upgrading a database adds and modifies system tables, system procedures, and database options to enable version 17 features. It does not change the file format used to store and access data on disk, and does not give access to all new features and performance enhancements in the latest software version.

[Upgrades and Rebuilds in a Database Mirroring System \[page 112\]](#)

All database servers in a database mirroring system must use the same minor release.

[Troubleshooting Database Upgrades: Ensure that JDBC Applications Are Not Running When Applying a Support Package \[page 121\]](#)

After applying a Support Package, you may find that your JDBC applications stop working.

[Troubleshooting Database Upgrades: Aggregate Functions and Outer References \[page 122\]](#)

SQL Anywhere follows ISO SQL standards for clarifying the use of aggregate functions when they appear in a subquery. These changes affect the behavior of statements written for previous versions of the software: previously valid queries may now produce error messages and result sets may change.

[Installing SQL Anywhere \(UNIX/Linux\) \[page 124\]](#)

Install SQL Anywhere using the command line.

[Installing SQL Anywhere \(Microsoft Windows\) \[page 125\]](#)

Install SQL Anywhere on Windows.

1.15.1.1 Upgrades of Version 10 and Later Databases

If you are upgrading from version 10 or later, you can either upgrade or rebuild your database.

Upgrading or rebuilding is an optional step because the version 17 software can be used with a version 10 or later database. However, to take advantage of all the new features in version 17, rebuild your database.

SQL Central will not support databases upgraded from version 11 or earlier. To use this feature, you must rebuild your database.

i Note

Refresh the materialized views in your database after upgrading your database server, or after rebuilding or upgrading your database to work with an upgraded database server.

Related Information

[The Upgrade Process for Version 10 and Later Databases \[page 107\]](#)

[The Rebuild Process for Version 10 and Later Databases \[page 96\]](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[Database Rebuilds](#)

[Refreshing a Materialized View Manually](#)

1.15.1.2 Upgrades of Version 9 and Earlier Databases

If you are upgrading to version 17 from version 9 or earlier, you must rebuild the database, which consists of unloading the old database, and reloading it into a new version 17 database.

Attempting to start version 9 or earlier databases results in an error on database startup. There are several approaches for rebuilding existing databases:

- Use the version 17 Unload utility (dbunload) with the `-an` (create a new database) or `-ar` (replace the old database) option.

i Note

The Unload utility (dbunload) has the same file name in all versions of SQL Anywhere. Ensure that you are using the correct version. Run the command `dbunload -?` to determine which version of the Unload utility you are using.

- Unload the database using the version 17 Unload utility, and then reload the database using the `reload.sql` file on the version 17 database server.
If you must make schema changes, this is the recommended way of upgrading. After you make the schema changes, you can create a database, and then apply the reload script to it.
- Use the [Unload Database Wizard](#) in SQL Central. You can choose to create a new database, replace an existing database with the new database, or unload the database to a file.
- Unload the database using an older version of dbunload, and then reload the database using the `reload.sql` file and the version 17 database server. Only use this approach if the other methods fail because deprecated or unsupported database option settings, objects, or SQL syntax could be unloaded into the `reload.sql` file. If problems occur during the reload, edit the file manually. The internal reload capabilities of version 17 take care of many of these problems.

i Note

SQL Anywhere 9.0.2 for Mac OS X was supported on PPC, while SQL Anywhere 10 and later for Mac OS X are supported on Intel. If you have a version 9.0.2 or earlier database on macOS, you have two options for unloading the database:

- Unload the database using the version 9.0.2 software.
- Copy the database to a different platform where SQL Anywhere 17 is installed, and then unload the database using the version 17 software.

Once the database is unloaded, you can perform the reload on macOS using the version 17 software.

Related Information

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[Database Rebuilds](#)

[Refreshing a Materialized View Manually](#)

1.15.1.3 User-defined Upgrades

In addition to performing system upgrades, you can define your own upgrade process by storing the SQL statements for a user-defined upgrade in a `.sql` script file.

User-defined upgrades allow you to perform an atomic upgrade of a database. The only changes made to the database are the changes listed in the script file. If an error occurs during the execution of the script, then the database is stopped (or restarted), and the upgrade is rolled back. All SQL statements in the script file execute normally as part of the script, but they do not produce transaction log entries.

The following statement restrictions and considerations apply to the contents of the upgrade script file:

- The ALTER DATABASE and BACKUP DATABASE statements are not permitted in the script file. If the script file contains one of these statements, the upgrade stops and the database is rolled back to the checkpoint that was automatically done at the start of the upgrade.
- CHECKPOINT statements are ignored.
- Any actions caused by CALL sa_server_option statements are not rolled back if the upgrade is rolled back and the database server is not shut down.
- SELECT and CALL statements that return a result set do not return the result set to the caller, but are still executed. This behavior can produce unintended side effects and affect the performance of the upgrade. As a workaround, these statements can be used with a cursor or FOR loop within the script if results need to be reported.
- An upgraded primary server in a mirrored configuration should not be used with mirror servers or copy nodes that have not been upgraded. All mirror and copy node databases must be replaced with copies of the upgraded database file prior to being started.
- An upgraded database can be used with an older database server, provided a system upgrade or unload/reload was not also performed.

Similar to system upgrades:

- Mirrored databases cannot be upgraded while they are running in a mirroring setup.
- Statements executed during the upgrade are not recorded in the transaction log.
- There must be no other connections to the database at upgrade time.
- The transaction log is renamed after a successful upgrade.
- Back up the database as soon as the upgrade is complete. Existing backups are no longer valid because they do not include the changes that were made as part of the upgrade.

Perform a user-defined upgrade by using the ALTER DATABASE statement or the [Upgrade Database Wizard](#) in SQL Central. The Upgrade utility (dbupgrad) does not support user-defined upgrades.

Related Information

[ALTER DATABASE Statement](#)

1.15.1.4 Upgrade and Rebuild Precautions

SQL Anywhere is used in many different configurations, and no upgrade guidelines can be guaranteed for all cases. There are several precautions that you must take before upgrading your software.

Back up your JRE directory before upgrading

Upgrading may overwrite the JRE directory (`%SQLANY17%\binXX\jre180`) and its subdirectories. In this case, if you are using certificates, your certificate store (`%SQLANY17%\binXX\jre180\lib\security\cacerts`) is overwritten, including your certificates. Similarly, fonts you added to the `%SQLANY17%\binXX\jre180\lib\fonts\fallback` directory to help display characters in the administration tools may be lost. To minimize upgrading steps with regards to the JRE change, create a backup copy of the JRE directory and all of its subdirectories before you upgrade so that you can refer to or restore files (such as `cacerts`) from the backup, as needed. To restore settings, use the `java_vm_options` option (SQL Anywhere), and/or the `-sl java` option (MobiLink) to optimize your JVM startup settings.

Check the behavior changes

Confirm that none of the documented behavior changes affect your application. If they do, update your application.

Test your application

Test your application thoroughly in a SQL Anywhere 17 environment before upgrading any applications in production use.

Use the correct version of the utilities

Make sure that you use the correct version of the database utilities with your new database.

Validate and back up the database

Before you begin an upgrade, validate your database, and back up your software and database. To ensure future recoverability, back up the database when you finish the upgrade.

Synchronize before upgrading

For databases involved in synchronization, such as UltraLite databases or SQL Anywhere remote databases in MobiLink installations, perform a successful synchronization before upgrading.

Test your upgrade procedure

Test your upgrade procedure carefully before carrying it out on a production system.

Related Information

[What's New in SQL Anywhere Version 17.0 \(Product-Wide\) \[page 29\]](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[-sl java mlsrv17 Option](#)
[java_vm_options Option](#)

1.15.1.5 How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed

If you have multiple versions of SAP SQL Anywhere on your Windows computer, check your system path when using utilities. The software installation adds its executable directory to the end of your system path, so it is possible to install a new version and inadvertently run an older version.

For example, if a version 12 executable directory is ahead of the 17 executable directory in your path and you use the dbinit command, you use the version 12 utility, and consequently create a version 12 database.

There are several ways you can ensure that you are using the version 17 utilities, including:

- Modify your system path so that the version 17 executable directory is before any previous version executable directory.
- Specify a fully qualified path name to the utility name that indicates the exact location of the utility you want to run.
- Create scripts to change your environment to use the correct version of the utilities.
- Uninstall the old software.

Related Information

[Environment Variables](#)

1.15.1.6 Rebuilding (Unloading/Reloading) a Database (Command Line)

Rebuild an older database to the latest version of the software.

Prerequisites

For version 17 databases, you must have the following system privileges:

- BACKUP DATABASE
- VALIDATE ANY OBJECT
- SERVER OPERATOR
- SELECT ANY TABLE

Follow the standard precautions for upgrading software.

Back up the database. For example:

```
dbbackup -c "DBF=mydb.db;UID=DBA;PWD=passwd" old-db-backup-dir
```

Procedure

1. If possible, defragment the drive where the new database will be stored because a fragmented drive can decrease database performance.
2. Shut down all SQL Anywhere database servers because the version 17 dbunload utility cannot be used against a database that is running on a previous version of the database server. For example:

```
dbstop -c "DBF=mydb.db;UID=DBA;PWD=passwd"
```

3. Unload and reload (rebuild) the old database into a new version 17 database. For example:

```
dbunload -c "DBF=mydb.db;UID=DBA;PWD=passwd" -an mydb17.db
```

4. Back up the new database before using it. For example:

```
dbbackup -c "DBF=mydb17.db;UID=DBA;PWD=passwd" new-db-backup-dir
```

5. Validate the new database before using it. For example:

```
dbvalid -c "DBF=mydb17.db;UID=DBA;PWD=passwd"
```

Results

The database is rebuilt to the latest version. By default, the database is stopped and restarted.

Next Steps

Test the rebuilt database with your application.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Rebuilding a Version 9 or Earlier Database with the Unload Utility \(dbunload\) \[page 106\]](#)

[Rebuilding a Version 9 or Earlier Database \(SQL Central\) \[page 104\]](#)

[Upgrading Authenticated Databases](#)

1.15.1.7 The Rebuild Process for Version 10 and Later Databases

Rebuilding a database consists of unloading and reloading the database to upgrade its file format. When you upgrade the file format, it changes the format used to store and access data on disk, letting you use all the new features and performance enhancements in the latest version of the software.

Caution

Unloading and reloading a large database can be time consuming and can require a large amount of disk space. The process may require disk space approximately twice the size of your database to hold the unloaded data and the new database file.

Because of index changes in SQL Anywhere, when you rebuild a database by unloading and reloading it, the rebuilt database may be smaller than the original database. This decrease in database size does not indicate a problem or a loss of data.

Note

Back up your database before you rebuild it.

In this section:

[Reloading Tables with AUTOINCREMENT Columns \[page 97\]](#)

You can retain the next available value for AUTOINCREMENT columns in the rebuilt database by specifying the `dbunload -l` option. This option adds calls to the `sa_reset_identity` system procedure to the generated `reload.sql` script for each table that contains an AUTOINCREMENT value, preserving the current value of `SYSTABCOL.max_identity`.

[Rebuilding a Database \(SQL Central\) \[page 97\]](#)

Use the *Unload Database Wizard* to upgrade a version 10 or later SQL Anywhere database to the latest version.

[Rebuilding a Database \(Command Line\) \[page 98\]](#)

Use the `dbunload` utility to upgrade a version 10 or later SQL Anywhere database to the latest version.

Related Information

[Improving Performance by Executing a List of CREATE INDEX or a List of LOAD TABLE Statements Concurrently](#)

[Improving Performance by Executing a List of CREATE INDEX or a List of LOAD TABLE Statements Concurrently](#)

[LOAD TABLE Statement](#)

[UNLOAD Statement](#)

[BEGIN PARALLEL WORK Statement](#)

1.15.1.7.1 Reloading Tables with AUTOINCREMENT Columns

You can retain the next available value for AUTOINCREMENT columns in the rebuilt database by specifying the `dbunload -I` option. This option adds calls to the `sa_reset_identity` system procedure to the generated `reload.sql` script for each table that contains an AUTOINCREMENT value, preserving the current value of `SYSTABCOL.max_identity`.

1.15.1.7.2 Rebuilding a Database (SQL Central)

Use the [Unload Database Wizard](#) to upgrade a version 10 or later SQL Anywhere database to the latest version.

Prerequisites

Follow the standard precautions for upgrading software.

To upgrade a version 10, 11, or 12 database, you do not require any system privileges.

To upgrade a version 16 or later database, you must have the SELECT ANY TABLE and the SERVER OPERATOR system privileges.

Ensure that the database is backed up before unloading and reloading it. For example:

```
dbbackup -c "DBF=mydb.db;UID=DBA;PWD=passwd" old-db-backup-dir
```

Procedure

1. Click **Start** > **Programs** > **SQL Anywhere 17** > **Administration Tools** > **SQL Central**.
2. Start a version 17 database server running the database you want to upgrade, and then connect to the database from SQL Central.
3. Click **Tools** > **SQL Anywhere 17** > **Unload Database**.
4. Read the text on the first page of the [Unload Database Wizard](#) and then click **Next**.
5. Click [Unload a database running on a current version of the server](#), and then select the database from the list. Click **Next**.
6. Choose to unload and reload into a new database. Click **Next**.
7. Specify a new file name for the database.
8. You can also specify the page size for the new database, but the page size you specify cannot be larger than the database server page size. The default page size is 4096 bytes. You can encrypt the database file if you want. If you choose strong encryption, you need the encryption key each time you want to start the database. Click **Next**.
9. Click [Unload structure and data](#). You can also select any other options you want for your database. Click **Next**.

10. Click [Unload all database objects](#). Click [Next](#).
11. Specify whether you want to connect to the new database when the unload/reload is complete.
12. Click [Finish](#) to start the process.

Results

The database is upgraded to the latest version. By default, the database is stopped and restarted.

Next Steps

Examine the new database to confirm that the rebuild completed properly and test the upgraded database with your application.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Tips on Exporting Data with the Unload Database Wizard](#)

[Database Rebuilds](#)

[Database Encryption and Decryption](#)

1.15.1.7.3 Rebuilding a Database (Command Line)

Use the `dbunload` utility to upgrade a version 10 or later SQL Anywhere database to the latest version.

Prerequisites

The database user specified in the `connection-string` must have the `SELECT ANY TABLE` and `SERVER OPERATOR` system privileges.

Follow the standard precautions for upgrading software.

Ensure that you have exclusive access to the database being upgraded and ensure that the path of the version 17 utilities is ahead of the path of the other utilities in your system path.

Ensure that the database is backed up before unloading and reloading. For example:

```
dbbackup -c "DBF=mydb.db;UID=DBA;PWD=passwd" old-db-backup-dir
```

Context

When using dbunload with a version 10 or later database, the version of dbunload used must match the version of the database server used to access the database. If an older version of dbunload is used with a newer database server, or vice versa, an error is returned.

If you are rebuilding a database that is a remote database in a MobiLink installation or that is involved in SQL Remote replication, and if you use the dbunload utility, you must be sure to use the `-ar` or `-an` option. These options ensure that the transaction log offsets for the new database are set to match those of the old database.

Procedure

1. Run the Unload utility (dbunload) and use the `-an` option to create a database.

```
dbunload -c "connection-string" -an new-db-file
```

This command creates a database. To replace the existing database with an upgraded database, use the `-ar` option instead of `-an`. To use the `-ar` option, connect to a personal database server, or to a network database server on the same computer as the Unload utility (dbunload).

2. Shut down the database and archive the transaction log before using the reloaded database.

To change the characteristics of the database during unload and reload (for example, change a case-sensitive database to a case-insensitive database), the procedure is more involved.

Results

The database is upgraded to the latest version. By default, the database is stopped and restarted.

Next Steps

Examine the new database to confirm that the rebuild completed properly and test the upgraded database with your application.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Database Rebuilds](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[Database Backup and Recovery](#)

1.15.1.8 The Rebuild Process for Version 9 and Earlier Databases

There are upgrade restrictions and considerations that are unique to rebuilding a version 9 or earlier database. Always back up a database before rebuilding it.

⚠ Caution

Unloading and reloading a large database can be time consuming and can require a large amount of disk space. The process requires access to disk space twice the size of your database to hold the unloaded data and the new database file.

Consideration	Details
macOS support	<p>SQL Anywhere 9.0.2 for Mac OS X was supported on PPC, while SQL Anywhere 10.0.0 and later for Mac OS X are supported on Intel. If you have a version 9.0.2 or earlier database on macOS, you have two options for unloading the database:</p> <ul style="list-style-type: none">• Unload the database using the version 9.0.2 software.• Copy the database to a different platform where SQL Anywhere 17 is installed, and then unload the database using the version 17 software. <p>Once the database is unloaded, you can perform the reload on macOS using the version 17 software.</p>

Consideration**Details**

Upgrade restrictions when rebuilding version 9 or earlier databases using the version 17 tools

- Disconnect the database from any earlier versions of the database server, and shut down any earlier database servers running on the computer. You must also shut down any version 17 database servers that are running on the computer. If dbunload cannot proceed because it detects any of these cases, it issues an error and fails.
- Do not include the ENG, START, or LINKS connection parameters in the dbunload connection string for the old database (specified in the -c option). If you specify these parameters, they are ignored and a warning appears. In the SQL Central *Connect* window, do not enter values in the *Server name* or *Start line* fields.
- Run dbunload on a computer with direct file system access to the old database (dbunload must be able to connect to the database using shared memory).
- Do not run a database server named dbunload_support_engine on the computer where the rebuild is taking place.
- Do not unload a version 9 or earlier database when the database file requires recovery. When you use the Unload utility (dbunload) on a database file that requires recovery, a message is returned indicating that the database could not be started. Use a version 9 database server to start the database and then stop the database before retrying dbunload. To unload a version 9 or later database, you must be able to start the database in read-only mode.

Password case sensitivity

In newly created SQL Anywhere 17 databases, all passwords are case sensitive, regardless of the case-sensitivity of the database.

When you rebuild an existing database, SQL Anywhere determines the case sensitivity of the password as follows:

- If the password was originally entered in a case-insensitive database, the password is case-insensitive.
 - If the password was originally entered in a case-sensitive database, uppercase and mixed case passwords remain case sensitive. However, if the password was entered in all lowercase, then the password becomes *case-insensitive*.
 - Changes to both existing passwords and new passwords are case sensitive.
-

Consideration	Details
Password length	The default minimum password length for new or upgraded databases is 6 characters. Old passwords are not modified during the unload/reload (so passwords less than the minimum are still accepted), but once the password is updated, the new password must meet the minimum password length requirement as specified by the <code>min_password_length</code> database option.
Page sizes	The default database page size for SQL Anywhere 17 databases is 4096 bytes. The supported page sizes in version 17 are 4096 bytes, 8192 bytes, 16384 bytes, and 32768 bytes. If your old database uses an unsupported page size, the new database has a page size of 4096 bytes by default. Use the <code>dbinit -p</code> option or the <code>dbunload -ap</code> option to specify a different page size.
Collations	<p>In version 9 and earlier, the database server supported one collation used with CHAR data types. This collation used the SQL Anywhere Collation Algorithm (SACA). In version 10 and later, the database server supports two collation algorithms, SACA and UCA (Unicode Collation Algorithm). Unless you specify a new or different collation for the rebuilt database, the SACA collation from the old database is unloaded and reused in the rebuilt database.</p> <p>If you are rebuilding a database with a custom collation, the collation is preserved only if you rebuild in a single step (internal unload). If you choose to unload the database, and then load the schema and data into a database that you create, then you must use one of the supplied collations.</p>
Database file size	Because of index changes in SQL Anywhere, when you rebuild a database by unloading and reloading it, the rebuilt database may be smaller than the original database. This decrease in database size does not indicate a problem or a loss of data.

Known issues

If the rebuild process fails when you run `dbunload` or the [Unload Database Wizard](#), then you can use the following steps to help diagnose the reason for the failure.

1. Create a new, empty version 17 database.

```
dbinit -dba DBA,passwd test.db
```

2. Apply the `reload.sql` file to the empty database.

```
dbisql -c "DBF=test.db;UID=DBA;pwd=passwd" reload.sql
```

3. Change the `reload.sql` file or the original database based on the messages you receive when applying the `reload.sql` file to the new database.

The following table lists issues that are known to cause a rebuild to fail, as well as their solutions.

Known problem	Solution
A DECLARE LOCAL TEMPORARY TABLE statement in a procedure or trigger causes a syntax error if the table name is prefixed with an owner name.	Remove the owner name.
If a CREATE TRIGGER statement does not include an owner name for the table on which the trigger is defined, and the table must be qualified with an owner when referenced by the user executing the <code>reload.sql</code> file, the statement fails and an error is returned indicating that the table could not be found.	Prefix the table name with the owner name.
If an object name (such as a table, column, variable, or parameter name) corresponds to a reserved word introduced in a later version of SQL Anywhere (such as NCHAR), then the reload fails. For example:	Change all references to the reserved word to use a different name. For variable names, prefixing the name with @ is a common convention that prevents naming conflicts.
<pre>CREATE PROCEDURE p () BEGIN DECLARE NCHAR INT; SET NCHAR = 1; END;</pre>	
If a database is unloaded with a version 9 or earlier copy of <code>dbunload</code> , the <code>reload.sql</code> file can contain calls to the <code>ml_add_property</code> system procedure, but this procedure is not present in a new version 17 database.	Unload the database with the version 17 <code>dbunload</code> utility.
If you unload a database using a version 9 or earlier version of <code>dbunload</code> , views that use Transact-SQL outer joins (by specifying <code>*=</code> or <code>=*</code>) may not be created properly when they are reloaded.	<p>Add the following line to the reload script:</p> <pre>SET TEMPORARY OPTION tsql_outer_joins='on';</pre> <p>Rewrite any views that use Transact-SQL outer joins.</p>
The [NOT] DETERMINISTIC clause is not supported in the CREATE PROCEDURE and ALTER PROCEDURE statements. If the clause is present, the reload fails and a syntax error is returned.	If you are upgrading a database that contains user-defined procedures that include the [NOT] DETERMINISTIC clause, you must remove the clause before you unload and reload the database.

In this section:

[Rebuilding a Version 9 or Earlier Database \(SQL Central\) \[page 104\]](#)

Use the *Unload Database Wizard* to rebuild a version 9 or earlier database. Rebuilt databases support all new features and performance enhancements in the latest software version.

[Rebuilding a Version 9 or Earlier Database with the Unload Utility \(dbunload\) \[page 106\]](#)

Use the Unload utility (dbunload) -an or -ar option to rebuild a version 9 or earlier database. Rebuilt databases support all new features and performance enhancements in the latest software version.

Related Information

[Reserved Words](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[-r Database Option](#)

[Initialization Utility \(dbinit\)](#)

[Unload Utility \(dbunload\)](#)

[Alternate Collations](#)

1.15.1.8.1 Rebuilding a Version 9 or Earlier Database (SQL Central)

Use the *Unload Database Wizard* to rebuild a version 9 or earlier database. Rebuilt databases support all new features and performance enhancements in the latest software version.

Prerequisites

Back up your database before rebuilding it.

The database file must be located on the same computer as the SQL Anywhere 17 installation.

You cannot unload a subset of tables from a database. Use the dbunload utility to do this.

If the *Unload Database Wizard* determines that the database file is already running, then the database is stopped before the unload proceeds.

Follow the standard precautions for upgrading software.

If possible, defragment the drive where the new database will be stored because a fragmented drive can decrease database performance.

Ensure that you have exclusive access to the database to be unloaded and reloaded. No other users can be connected.

Procedure

1. Click  *Start* > *Programs* > *SQL Anywhere 17* > *Administration Tools* > *SQL Central* .

2. Click **Tools** > **SQL Anywhere 17** > **Unload Database**.
3. Read the introductory page of the *Unload Database Wizard*, and click *Next*.
4. Click *Unload a database running on an earlier version of the server, or a database that is not running*. Specify the connection information for the database. Click *Next*.
5. Click *Unload and reload into a new database*. Click *Next*.
6. Specify a new file name for the database. Click *Next*.

You can specify the page size for the new database. In version 17, the default (and recommended) page size is 4096 bytes.

You can encrypt the database file. If you choose strong encryption, you need the encryption key each time you start the database.

7. Choose to unload structure and data. Click *Next*.
8. Specify whether you want to connect to the new database when the rebuild is complete.
9. Click *Finish*. Examine the new database to confirm that the rebuild completed properly.

Results

The database is upgraded to the latest version. By default, the database is stopped and restarted.

Next Steps

Examine the new database to confirm that the rebuild completed properly and test the rebuilt database with your application.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Database Encryption and Decryption](#)

[Database Backup and Recovery](#)

[Database Rebuilds](#)

[Unload Utility \(dbunload\)](#)

1.15.1.8.2 Rebuilding a Version 9 or Earlier Database with the Unload Utility (dbunload)

Use the Unload utility (dbunload) -an or -ar option to rebuild a version 9 or earlier database. Rebuilt databases support all new features and performance enhancements in the latest software version.

Prerequisites

The database user specified in the `connection-string` must have the SELECT ANY TABLE and SERVER OPERATOR system privileges.

Back up your database before rebuilding it.

The database file must be located on the same computer as the SQL Anywhere 17 installation.

Follow the standard precautions for upgrading software.

Ensure that you have exclusive access to the database to be unloaded and reloaded. No other users can be connected.

Ensure that the version 17 utilities are ahead of other utilities in your system path.

Context

The -an option is recommended because it creates a new database leaving the original database intact. The -ar option replaces your old database with a new version 17 database.

Procedure

1. Shut down all SQL Anywhere and Adaptive Server Anywhere database servers because the version 17 dbunload utility cannot be used against a database that is running on a previous version of the database server. For example:

```
dbstop -c "DBF=mydb.db;UID=DBA;PWD=sql"
```

2. If possible, defragment the drive where the new database will be stored because a fragmented drive can decrease database performance.
3. Back up the database. For example:

```
dbbackup -c "DBF=mydb.db;UID=DBA;PWD=sql" old-db-backup-dir
```

4. Run the Unload utility (dbunload) using the -an or -ar option to create a new database.

```
dbunload -c "connection-string" -an database-filename
```

For example:

```
dbunload -c "DBF=mydb.db;UID=DBA;PWD=sql" -an mydb17.db
```

This command creates a database (by specifying `-an`). If you specify the `-ar` option, the existing database is replaced with a rebuilt database. To use the `-ar` option, you must connect to a personal database server or to a network database server on the same computer as the Unload utility (`dbunload`).

Results

The database is upgraded to the latest version. By default, the database is stopped and restarted.

Next Steps

Examine the new database to confirm that the rebuild completed properly and test the rebuilt database with your application.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Database Backup and Recovery](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[Unload Utility \(dbunload\)](#)

1.15.1.9 The Upgrade Process for Version 10 and Later Databases

Upgrading a database adds and modifies system tables, system procedures, and database options to enable version 17 features. It does not change the file format used to store and access data on disk, and does not give access to all new features and performance enhancements in the latest software version.

In this section:

[Upgrading a Version 12 or Later Database \(SQL Central\) \[page 108\]](#)

Use the *Upgrade Database Wizard* to upgrade a version 12 or later database by adding and modifying system tables, system procedures, and database options to enable version 17 features.

[Upgrading a Version 10 or Later Database \(Command Line\) \[page 109\]](#)

Use the `dbupgrad` utility to upgrade a version 10 or later database by adding and modifying system tables, system procedures, and database options to enable version 17 features.

[Upgrading a Version 10 or Later Database \(SQL\) \[page 111\]](#)

Use the ALTER DATABASE statement to upgrade a version 10 or later database by adding and modifying system tables, system procedures, and database options to enable version 17 features.

1.15.1.9.1 Upgrading a Version 12 or Later Database (SQL Central)

Use the *Upgrade Database Wizard* to upgrade a version 12 or later database by adding and modifying system tables, system procedures, and database options to enable version 17 features.

Prerequisites

To upgrade a version 12 database, you do not require any system privileges.

To upgrade a version 16 or later database, you must have the ALTER DATABASE and the BACKUP DATABASE system privileges.

Follow the standard precautions for upgrading software.

⚠ Caution

Back up your database files before upgrading. If you apply the upgrade to the existing files, then these files become unusable if the upgrade fails. To back up your database, you must have the BACKUP DATABASE system privilege.

Context

The *Upgrade Database Wizard* does not upgrade a version 9.0.2 or earlier database to version 17. To upgrade an existing version 9.0.2 or earlier database to version 17, you must unload and reload the database using dbunload or the *Unload Database Wizard*.

Procedure

1. Follow the standard precautions for upgrading software.
2. Click **Start** > **Programs** > **SQL Anywhere 17** > **Administration Tools** > **SQL Central**.
3. From the *SQL Anywhere 17* plug-in, connect to the database you want to upgrade. The database must be running on a version 17 database server.
4. Click **Tools** > **SQL Anywhere 17** > **Upgrade Database**.
5. Follow the instructions in the *Upgrade Database Wizard*. By default, the database is stopped and restarted.

6. (Optional) Stop the database and archive the transaction log by making a copy of it before using the upgraded database if you did not choose to do so in the wizard.

Results

The database is upgraded and contains new and modified system tables, system procedures, and database options.

Next Steps

Examine the new database to confirm that the upgrade completed properly and test the upgraded database with your application.

Upgrading introduces new system privileges, which are automatically added to the UPGRADE_ROLE system privilege. You must distribute the privileges granted to the UPGRADE_ROLE system privilege to other roles and users, and then revoke them from the UPGRADE_ROLE system privilege.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Database Backup and Recovery](#)

[The Rebuild Process for Version 10 and Later Databases \[page 96\]](#)

[Upgrade and Rebuild Precautions \[page 93\]](#)

[System Privileges Introduced in Upgrades](#)

[Distributing Privileges Granted to the UPGRADE_ROLE System Privilege After an Upgrade](#)

[Upgrade Utility \(dbupgrad\)](#)

1.15.1.9.2 Upgrading a Version 10 or Later Database (Command Line)

Use the dbupgrad utility to upgrade a version 10 or later database by adding and modifying system tables, system procedures, and database options to enable version 17 features.

Prerequisites

The database user specified in the `connection-string` must have the ALTER DATABASE system privilege, and must be the only connection to the database.

Follow the standard precautions for upgrading software.

Ensure that you have exclusive access to the database to be upgraded and ensure that the version 17 utilities are ahead of other utilities in your system path.

⚠ Caution

Back up your database files before upgrading. If you apply the upgrade to the existing files, then these files become unusable if the upgrade fails.

Procedure

1. Run the Upgrade utility (dbupgrad) against the database:

```
dbupgrad -c "connection-string"
```

2. Shut down the database and archive the transaction log before using the upgraded database.

Results

The database is upgraded and contains new and modified system tables, system procedures, and database options.

Next Steps

Examine the new database to confirm that the upgrade completed properly and test the upgraded database with your application.

Upgrading may introduce new system privileges, which are automatically added to the UPGRADE ROLE system privilege. You must distribute the privileges granted to the UPGRADE ROLE system privilege to other roles and users, and then revoke them from the UPGRADE ROLE system privilege.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

[Database Backup and Recovery](#)

[System Privileges Introduced in Upgrades](#)

[Distributing Privileges Granted to the UPGRADE ROLE System Privilege After an Upgrade](#)

[Upgrade Utility \(dbupgrad\)](#)

1.15.1.9.3 Upgrading a Version 10 or Later Database (SQL)

Use the ALTER DATABASE statement to upgrade a version 10 or later database by adding and modifying system tables, system procedures, and database options to enable version 17 features.

Prerequisites

Follow the standard precautions for upgrading software.

You must be the owner of the database or have ALTER DATABASE system privilege, and must be the only connection to the database.

⚠ Caution

Back up your database files before upgrading. If you apply the upgrade to the existing files, then these files become unusable if the upgrade fails.

Procedure

1. Connect to the database from Interactive SQL or another application that can execute SQL statements. No other connection can be using the database at the same time.
2. Execute an ALTER DATABASE statement.

For example, the following statement upgrades a database:

```
ALTER DATABASE UPGRADE;
```

3. Shut down the database and archive the transaction log before using the upgraded database.

Results

The database is upgraded and contains new and modified system tables, system procedures, and database options.

Next Steps

Examine the new database to confirm that the upgrade completed properly and test the upgraded database with your application.

Upgrading may introduce new system privileges, which are automatically added to the UPGRADE ROLE system privilege. You must distribute the privileges granted to the UPGRADE ROLE system privilege to other roles and users, and then revoke them from the UPGRADE ROLE system privilege.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Database Backup and Recovery](#)

[System Privileges Introduced in Upgrades](#)

[Distributing Privileges Granted to the UPGRADE ROLE System Privilege After an Upgrade](#)

[ALTER DATABASE Statement](#)

1.15.1.10 Upgrades and Rebuilds in a Database Mirroring System

All database servers in a database mirroring system must use the same minor release.

In this section:

[Updating the Database Server Software to a Major Version for a Mirroring System Without Rebuilding the Database \[page 113\]](#)

Update the software of the database servers in a mirroring system by stopping each database server, installing the software, and starting the databases on servers running the new software. The database itself is not upgraded or rebuilt.

[Updating the Database Server Software to a Different Minor Release for a Mirroring System Without Rebuilding the Database \[page 115\]](#)

Update the software of the database servers in a mirroring system by stopping each server, installing the software, and starting the database on the new server. All database servers in a database mirroring system must use the same minor release.

[Updating the Database Server Software to a Support Package for Database Mirroring System Without Rebuilding the Database \[page 117\]](#)

Update the software of the database servers in a mirroring system by stopping each server, installing the software, and then restarting the database on the new server.

[Upgrading or Rebuilding \(Unloading/Reloading\) the Databases in a Database Mirroring System \[page 119\]](#)

Upgrade the primary database, and then copy the upgraded database and transaction log to the mirror. The mirroring system is temporarily stopped. Upgrading databases is generally performed when updating to a major version or minor release of the software.

1.15.1.10.1 Updating the Database Server Software to a Major Version for a Mirroring System Without Rebuilding the Database

Update the software of the database servers in a mirroring system by stopping each database server, installing the software, and starting the databases on servers running the new software. The database itself is not upgraded or rebuilt.

Prerequisites

You must have the BACKUP DATABASE system privilege or have the VALIDATE ANY OBJECT system privilege.

By default, you must have the SERVER OPERATOR system privilege to stop network database servers.

Context

Test the following steps in non-production environment with your application before performing the steps in a production environment.

All database servers in a database mirroring system must use the same minor release of SQL Anywhere. Because of this requirement, you must stop the mirroring system temporarily during the update.

Procedure

1. Make a backup of the primary database, copy the backup, and validate the copy of the backup.

For example, the following command backs up a database named `mydb.db`:

```
dbbackup -c "DBN=mydb;SERVER=myserver;HOST=primaryhost;UID=DBA;PWD=sql"  
backup-dir
```

To create a copy of the backup, run the following commands:

```
xcopy backup-dir\mydb.db validatebackup-dir\
```

Validate the backup copy.

```
dbvalid -c "DBF=validatebackup-dir\mydb.db;UID=DBA;PWD=sql"
```

If the copy of the backup does not validate, then fix the problems that cause the validation to fail before proceeding. Otherwise, you risk losing data.

2. Install the new software on the primary, mirror, and arbiter servers.
3. If the mirroring system is involved in read-only scale-out, install the software on the copy nodes.

4. Stop the servers in the following order:

- a. copy nodes
- b. mirror server
- c. primary server
- d. arbiter server

For example, run the Stop utility (dbstop):

```
dbstop -y -c "UID=DBA;PWD=sql;Server=myserver"
```

5. (Optional) Upgrade or rebuild the databases.

6. Start the databases on the new servers in the following order:

- a. arbiter server, primary server, and mirror server
- b. copy nodes

Results

The databases in the mirroring system run on the new version of the software.

Next Steps

Examine the database mirroring system to confirm that the update completed properly, and test the database mirroring system with your application.

Related Information

[User-initiated Role Switches \(Failovers\)](#)

[Stopping a Database Server in a Mirroring System \(dbstop Utility\)](#)

[Stop Server Utility \(dbstop\)](#)

1.15.1.10.2 Updating the Database Server Software to a Different Minor Release for a Mirroring System Without Rebuilding the Database

Update the software of the database servers in a mirroring system by stopping each server, installing the software, and starting the database on the new server. All database servers in a database mirroring system must use the same minor release.

Prerequisites

You must have the BACKUP DATABASE system privilege and the VALIDATE ANY OBJECT system privilege.

By default, you must have the SERVER OPERATOR system privilege to stop network database servers.

Test the following steps in non-production environment with your application before performing the steps in a production environment.

Context

The mirroring system stops temporarily during the update.

Procedure

1. Make a backup of the primary database, copy the backup, and validate the copy of the backup.

For example, run the following command to back up a database named `mydb.db`:

```
dbbackup -c "DBN=mydb;SERVER=myserver;HOST=primaryhost;UID=DBA;PWD=passwd"
backup-dir
```

Create a copy of the backup:

```
xcopy backup-dir\ validatebackup-dir\
```

Validate the backup copy:

```
dbvalid -c "DBF=validatebackup-dir\backupmydb.db;UID=DBA;PWD=passwd"
```

If the copy of the backup does not validate, then fix the problems that cause the validation to fail before proceeding. Otherwise, you risk losing data.

2. Stop the servers in the following order:
 - a. copy nodes
 - b. mirror server

- c. primary server
- d. arbiter server

For example, run the Stop utility (dbstop):

```
dbstop -y -c "UID=DBA;PWD=passwd;Server=myserver"
```

3. Install the new software on the primary, mirror, and arbiter servers.
4. Start the servers in the following order:
 - a. arbiter server
 - b. primary server
 - c. mirror server
5. If the mirroring system is involved in read-only scale-out, install the software on the copy nodes, and then restart the copy nodes

Results

The databases in the mirroring system run on the new version of the software.

Next Steps

Examine the database mirroring system to confirm that the upgrade completed properly, and test the database mirroring system with your application.

Related Information

[User-initiated Role Switches \(Failovers\)](#)

[Stopping a Database Server in a Mirroring System \(dbstop Utility\)](#)

[Stop Server Utility \(dbstop\)](#)

1.15.1.10.3 Updating the Database Server Software to a Support Package for Database Mirroring System Without Rebuilding the Database

Update the software of the database servers in a mirroring system by stopping each server, installing the software, and then restarting the database on the new server.

Prerequisites

You must have the BACKUP DATABASE system privilege.

You must have the VALIDATE ANY OBJECT system privilege.

By default, you must have the SERVER OPERATOR system privilege to stop network database servers.

Context

Because you can stop servers in the system one at a time, the mirroring system can continue to run during a Support Package update of the server software. A failover from the primary server to the mirror server occurs at least once during the update process. Connections to the primary and mirror server drop during any failover.

Test the following steps in non-production environment with your applications before performing the steps in a production environment.

Procedure

1. Make a backup of the primary database, copy the backup, and validate the copy of the backup.

For example, run the following command to back up a database named `mydb.db`:

```
dbbackup -c "DBN=mydb;SERVER=myserver;HOST=primaryhost;UID=DBA;PWD=passwd"
backup-dir
```

Create a copy of the backup:

```
xcopy backup-dir\ validatebackup-dir\
```

Validate the backup copy:

```
dbvalid -c "DBF=validatebackup-dir\mydb.db;UID=DBA;PWD=passwd"
```

If the copy of the backup does not validate, then fix the problems that cause the validation to fail before proceeding. Otherwise, you risk losing data.

2. If the mirroring system is part of a read-only scale-out system, then for each copy node:

- a. Stop the copy node server.
 - b. Install the software.
 - c. Start the copy node.
3. Stop the mirror server.
 4. Install the software on the mirror server.
 5. Start the mirror server and ensure that it is in a synchronized state.
 6. Stop the arbiter server.
 7. Install the software on the arbiter.
 8. Start the arbiter server.
 9. Initiate a fail over by connecting to the primary database and executing the following statement:

```
ALTER DATABASE SET PARTNER FAILOVER;
```

Connections to the primary and mirror drop during the failover. The current primary becomes the mirror.

10. Stop the mirror server.
11. Install the software on the mirror server.
12. Start the mirror server and ensure that it is synchronized.

Results

The databases in the mirroring system run on the new version of the software.

Next Steps

Examine the database mirroring system to confirm that the upgrade completed properly, and test the database mirroring system with your application.

Related Information

[User-initiated Role Switches \(Failovers\)](#)

[Stopping a Database Server in a Mirroring System \(dbstop Utility\)](#)

[Stop Server Utility \(dbstop\)](#)

1.15.1.10.4 Upgrading or Rebuilding (Unloading/Reloading) the Databases in a Database Mirroring System

Upgrade the primary database, and then copy the upgraded database and transaction log to the mirror. The mirroring system is temporarily stopped. Upgrading databases is generally performed when updating to a major version or minor release of the software.

Prerequisites

You must have the BACKUP DATABASE system privilege.

You must have the VALIDATE ANY OBJECT system privilege.

By default, you must have the SERVER OPERATOR system privilege to start or stop network database servers.

To upgrade a database, you must have the ALTER DATABASE system privilege, and must be the only connection to the database

To rebuild (unload/reload) a database using the Unload utility (dbunload), you must have the SELECT ANY TABLE system privilege. For an unload with a reload, you must also have the SERVER OPERATOR system privilege.

Test the following steps in non-production environment with your application before performing the steps in a production environment.

Procedure

1. (Optional) Install the new version of the software on each computer in the system. All database servers in a database mirroring system must use the same maintenance version of SQL Anywhere. If you are upgrading to a new minor version, you must install the software while the server is stopped.
2. Make a backup of the primary database, copy the backup, and validate the copy of the backup.

For example, run the following command to back up a database named `mydb.db`:

```
dbbackup -c "DBN=mydb;SERVER=myserver;HOST=primaryhost;UID=DBA;PWD=passwd"  
backup-dir
```

Create a copy of the backup:

```
xcopy backup-dir\ validatebackup-dir\
```

Validate the backup copy:

```
dbvalid -c "DBF=validatebackup-dir\mydb.db;UID=DBA;PWD=passwd"
```

If the copy of the backup does not validate, then fix the problems that cause the validation to fail before proceeding. Otherwise, you risk losing data.

3. Stop the servers in the following order:

- a. copy nodes
 - b. mirror
 - c. primary
4. Back up the primary database and transaction log to another backup directory. This step is required because changes could have been made to the primary database since the previous backup.

```
xcopy /y c:\primary-database.db backup_dir\  
xcopy /y c:\primary-database.log backup_dir\  

```

5. Choose one of the following options to upgrade or rebuild the primary database:

Option	Action
Upgrade the database	<p>Run the Upgrade utility (dbupgrad) on the primary database. For example:</p> <pre>dbupgrad -c "UID=DBA;PWD=passwd;DBF=C:\primary-database.db"</pre> <p>The database is upgraded, a new transaction log is created, and the database is stopped. You can delete the old transaction logs.</p>
Rebuild (unload/reload) the database	<pre>dbunload -c "UID=DBA;PWD=passwd;DBF=C:\primary-database.db" -ar</pre>

6. Start and stop the upgraded or rebuilt database to create a transaction log for the database. For example, run the following command:

```
dbping -d -c "UID=DBA;PWD=passwd;DBF=C:\primary-database.db"
```

7. Copy the upgraded or rebuilt database and its new transaction log to the mirror server and any scale-out copy nodes.
8. Start the servers in the following order:
 - a. primary
 - b. mirror
 - c. copy nodes

Results

The databases in the mirroring system are upgraded or rebuilt and the mirroring system is running.

Next Steps

Examine the database mirroring system to confirm that the upgrade completed properly, and test the database mirroring system with your application.

Related Information

[User-initiated Role Switches \(Failovers\)](#)

[Stopping a Database Server in a Mirroring System \(dbstop Utility\)](#)

[Stop Server Utility \(dbstop\)](#)

1.15.1.11 Troubleshooting Database Upgrades: Ensure that JDBC Applications Are Not Running When Applying a Support Package

After applying a Support Package, you may find that your JDBC applications stop working.

You receive a message similar to the following: `The sajdbc4.jar build does not match the shared object build.`

This message can be returned because either Interactive SQL, SQL Central, the fast launchers, or your own JDBC applications were running when the Support Package was applied. In these cases, the Java VM locks any DLLs or shared objects that are loaded by it, but does not lock the JAR files. As a result, applying the Support Package updates the sajdbc4 and jodbc4 JAR files, but not the accompanying dbjdbc17 and/or dbjodbc17 DLL or shared object. When the JDBC application is restarted, the JDBC JAR file does not match the build of the accompanying DLL or shared object, and a message like the one above is returned.

First, try shutting down all JDBC-based applications and then reapplying the Support Package. If reapplying the Support Package does not work, try resolving the problem through the following methods:

Check that the Support Package installer properly updated DLLs and shared objects

- On Microsoft Windows, check that the `dbjdbc17.dll` and `dbjodbc17.dll` files were properly updated when applying the Support Package.
- On UNIX and Linux (excluding macOS), check that the `libdbjdbc17.so.1` and `libdbjodbc17.so.1` shared objects were properly updated when applying the Support Package.
- On macOS, check that the `libdbjdbc17.dylib` and `libdbjodbc17.dylib` shared objects were properly updated when applying the Support Package.

i Note

There may be both 32-bit and 64-bit versions of the DLLs or shared objects installed on your system. When checking the DLLs or shared objects, you must check those that match the bitness of the JAVA VM, not the bitness of the database server.

Check that there are not multiple copies of the DLLs and shared objects on your system

If the DLLs or shared objects have been properly updated, make sure that you do not have multiple copies of the DLLs or shared objects with the same bitness. You have multiple copies of DLLs if you have copied the DLLs or shared objects to the extensions folders of the Java VM to bypass the Java restriction that does not allow DLLs and shared objects to be loaded within multiple class loaders.

Check that the JAR files were properly updated

If your DLLs or shared objects were properly updated, and there are not multiple copies on your system, then make sure that the various JAR files were properly updated. To check each JAR file, run the following

commands, and ensure that the SQL Anywhere version and build number reported by the JAR file matches the SQL Anywhere version and build number of the Support Package you installed.

To check `sajdbc4.jar`, run the following command (replacing `path` with the path to the JAR file):

```
java -cp path\sajdbc4.jar sap.jdbc4.sqlanywhere.IBuildNum
```

To check `jodbc4.jar`, run the following command (replacing `path` with the path to the JAR file):

```
java -cp path\jodbc4.jar ianywhere.ml.jdbcodbc.jdbc4.IBuildNum
```

Once you determine which JAR file, DLL, or shared object does not match the Support Package build number, make sure that the file is not locked by an application, and then reapply the Support Package.

1.15.1.12 Troubleshooting Database Upgrades: Aggregate Functions and Outer References

SQL Anywhere follows ISO SQL standards for clarifying the use of aggregate functions when they appear in a subquery. These changes affect the behavior of statements written for previous versions of the software: previously valid queries may now produce error messages and result sets may change.

When an aggregate function appears in a subquery, and the column referenced by the aggregate function is an outer reference, the entire aggregate function itself is treated as an outer reference. The aggregate function is computed in the outer query block, not in the subquery, and becomes a constant within the subquery.

The following restrictions apply to the use of outer reference aggregate functions in subqueries:

- The outer reference aggregate function can only appear in subqueries that are in the SELECT list or HAVING clause, and these clauses must be in the immediate outer block.
- Outer reference aggregate functions can only contain one outer column reference.
- Local column references and outer column references cannot be mixed in the same aggregate function.

Some problems related to the new standards can be circumvented by rewriting the aggregate function so that it only includes local references. For example, the subquery (`SELECT MAX(S.y + R.y) FROM S`) contains both a local column reference (`S.y`) and an outer column reference (`R.y`), which is now illegal. It can be rewritten as (`SELECT MAX(S.y) + R.y FROM S`). In the rewrite, the aggregate function has only a local column reference. The same sort of rewrite can be used when an outer reference aggregate function appears in clauses other than SELECT or HAVING.

Example

Example 1

The following query produced valid results in SQL Anywhere 7 and earlier versions:

```
SELECT Name,  
       ( SELECT SUM( p.Quantity )  
         FROM SalesOrderItems )
```

```
FROM Products p WHERE p.ID = 300;
```

Name	SUM(p.Quantity)
Tee shirt	30,716

In SQL Anywhere 8 and later versions, the same query produces an error message stating that the function or column reference must appear in a GROUP BY clause. The statement is no longer valid because `SUM(p.Quantity)` is treated as an outer reference and computed in the outer query block. The above query is equivalent to the following query:

```
SELECT Name,  
       SUM( p.Quantity ) AS Z,  
       ( SELECT Z  
         FROM SalesOrderItems )  
FROM Products p WHERE p.ID = 300;
```

However, since the aggregate function is now computed in the outer query block, the outer query block is treated as a grouped query and the column name must appear in a GROUP BY clause in the SELECT list. Therefore, this query is also invalid and produces the same error message. To return the same result set in SQL Anywhere 8 and later versions as the first query did in SQL Anywhere 7 and previous versions, you must use the following query:

```
SELECT Name,  
       p.Quantity * ( SELECT COUNT( * ) FROM SalesOrderItems )  
FROM Products p WHERE p.ID = 300;
```

Example 2

In SQL Anywhere 7 and earlier versions, the following query produced the result 30,716, because `SUM(p.Quantity)` was computed inside the nested SELECT query block and only `p.Quantity` was treated as an outer reference.

```
SELECT ( SELECT FIRST SUM( p.Quantity ) FROM SalesOrderItems ) AS ss  
FROM Products p WHERE p.ID = 300;
```

In SQL Anywhere 8 and later versions, the same query returns the result 28, because `SUM(p.Quantity)` is treated as an outer reference and is computed in the outer query block. In other words, the above query is equivalent to the following query:

```
SELECT DT.ss  
FROM ( SELECT SUM( p.Quantity ) AS asum,  
       ( SELECT FIRST asum FROM SalesOrderItems ) AS ss  
      FROM Products p WHERE p.ID = 300 ) AS DT;
```

To return the same results in SQL Anywhere 8 and later versions as the first query did in SQL Anywhere 7 and previous versions, you must use the following query:

```
SELECT p.Quantity * ( SELECT COUNT( * ) FROM SalesOrderItems ) AS ss  
FROM Products p WHERE p.ID = 300;
```

Related Information

[The HAVING Clause: Selecting Groups of Data](#)

1.15.1.13 Installing SQL Anywhere (UNIX/Linux)

Install SQL Anywhere using the command line.

Prerequisites

You have copied the installation files to your computer.

Procedure

1. Open a shell and navigate to the root of the installation image.
2. Run the following command to install the software:

```
./setup
```

(Optional) To view the usage, run the following command:

```
./setup -h
```

3. Follow the on-screen instructions.

Results

SQL Anywhere installs.

Related Information

[UNIX and Linux Environment Variables](#)

[File Locations and Installation Settings](#)

[Sourcing sa_config.sh, sample_config32.sh, and sample_config64.sh \[UNIX/Linux\]](#)

1.15.1.14 Installing SQL Anywhere (Microsoft Windows)

Install SQL Anywhere on Windows.

Prerequisites

You have the SQL Anywhere installation CD, or have the installation files on your computer.

Procedure

1. Navigate to the root of the installation CD, or the directory containing the SQL Anywhere installation and run `setup.exe`.
2. Follow the on-screen instructions.

Results

SQL Anywhere installs.

Related Information

[SQL Anywhere on Windows](#)
[File Locations and Installation Settings](#)
[Tutorial: Connecting to the Sample Database Editions and Licensing](#)
[Separately Licensed Components](#)

1.15.2 MobiLink Upgrades

Before upgrading, check for behavior changes that may affect you and take standard upgrade precautions.

Compatibility With Existing Software

- Version 17 MobiLink clients are compatible with version 16 or later MobiLink servers. However, in some cases functionality may be limited if you are not using a version 17 MobiLink server.

- The version 17 MobiLink server can be used with clients that are version 12 or later. If you must support earlier clients, keep an earlier version of the MobiLink server to support them.
- Confirm that none of the documented behavior changes affect your application. If they do, update your application.

i Note

There are no changes to the MobiLink server system objects between versions 16 and 17, so there are no upgrade scripts for you to run if you are using a version 16 MobiLink server.

Upgrade Order

If you are upgrading an existing MobiLink installation, upgrade the components in the following order:

1. Shut down the MobiLink servers.
2. Upgrade the consolidated database.
3. Upgrade the MobiLink servers.
4. Start the MobiLink servers.
5. Upgrade the MobiLink clients.

The version 17 MobiLink server can only be used with clients that are version 11 or later. MobiLink clients before version 11 must be upgraded to work with version 17 MobiLink server.

In this section:

[Consolidated Database Upgrades \[page 127\]](#)

Before you can use the new MobiLink server with an existing consolidated database, you must run upgrade scripts that install new system objects. The upgrade scripts must be run by the owner of the currently installed MobiLink system tables.

[MobiLink Server Upgrades \[page 135\]](#)

Upgrade the MobiLink server to version 17 if you are synchronizing version 17 remote databases.

[SQL Anywhere MobiLink Client Upgrades \[page 136\]](#)

In a production environment, only upgrade SQL Anywhere remote databases after you have upgraded both the consolidated database and the MobiLink server. You only need to upgrade the MobiLink server if it is version 12 or earlier because version 16 and 17 clients can synchronize with a version 17 MobiLink server.

Related Information

[What's New in MobiLink Version 17.0 \[page 76\]](#)

[Upgrade and Rebuild Precautions \[page 93\]](#)

[UltraLite Upgrades \[page 138\]](#)

1.15.2.1 Consolidated Database Upgrades

Before you can use the new MobiLink server with an existing consolidated database, you must run upgrade scripts that install new system objects. The upgrade scripts must be run by the owner of the currently installed MobiLink system tables.

You can also use the following methods to update the MobiLink system setup:

- In the MobiLink plug-in for SQL Central, click **► MobiLink 17 ► Project ► Consolidated Databases ►** and right-click the database name and click *Check MobiLink System Setup*. If your database requires setup or upgrading, you are prompted to continue.
- When you use the *Deploy Synchronization Model Wizard*, system setup is checked when you connect to your consolidated database. If your database requires setup or upgrading, you are prompted to continue.

The MobiLink upgrade scripts for 6.0.x have been removed. If you require this upgrade, contact Technical Support.

Notes

- Use the `ml_add_missing_dnl_d_scripts` stored procedure to fix missing `download_cursor` and/or `download_delete_cursor` scripts. Invoking this procedure with a script version name defines the missing `download_cursor` and/or `download_delete_cursor` scripts as ignored scripts for every synchronization table used by the given script version.
- If you have `authenticate_user_hashed` scripts that were created earlier than version 10.0.0, change them to accept `VARBINARY(32)` instead of `VARBINARY(20)`, using the binary equivalent type of your RDBMS.

In this section:

[Upgrading a Consolidated Database \(SQL Anywhere 10.0.0 and Later\) \[page 128\]](#)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing SQL Anywhere consolidated database that is version 10.0.0 or later.

[Upgrading a Consolidated Database \(Adaptive Server Enterprise, SAP IQ, Oracle, MySQL, Microsoft SQL Server, or SAP HANA\) \[page 129\]](#)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing consolidated database.

[Upgrading an IBM DB2 LUW Consolidated Database \(Deprecated\) \[page 131\]](#)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing IBM DB2 LUW consolidated database.

[Upgrading a Consolidated Database \(SQL Anywhere Earlier Than 10.0.0\) \[page 132\]](#)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing SQL Anywhere consolidated database that is earlier than version 10.0.0.

Related Information

1.15.2.1.1 Upgrading a Consolidated Database (SQL Anywhere 10.0.0 and Later)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing SQL Anywhere consolidated database that is version 10.0.0 or later.

Prerequisites

i Note

Adobe will stop updating and distributing the Flash Player at the end of 2020. Because the SQL Anywhere Monitor is based on Flash, you cannot use it once Flash support ends. In many cases, tasks that were previously performed in the Monitor can be performed in the SQL Anywhere Cockpit. See [SQL Anywhere Monitor Non-GUI User Guide](#).

You must be the owner of the currently installed MobiLink system tables to run the setup scripts.

The MONITOR system privilege is required to invoke the locking/blocking detection logic.

Context

You can also use the following methods to update the MobiLink system setup:

- In the MobiLink plug-in for SQL Central, click **► MobiLink 17 ► Project ► Consolidated Databases ►** and right-click the database name and click *Check MobiLink System Setup*. If your database requires setup or upgrading, you are prompted to continue.
- When you use the *Deploy Synchronization Model Wizard*, system setup is checked when you connect to your consolidated database. If your database requires setup or upgrading, you are prompted to continue.

Procedure

1. Upgrade the SQL Anywhere software.
2. Upgrade the MobiLink system setup by running the appropriate upgrade script for the version you are upgrading from.

The upgrade script is called `upgrade_sa.sql`. It is located under your SQL Anywhere installation in `MobiLink\upgrade\version`, where `version` is the SQL Anywhere version you are upgrading from.

For example, connect to the database in Interactive SQL and execute the following statement:

```
READ "C:\Program Files\SQL Anywhere 17\MobiLink\upgrade\10.0.x\upgrade_sa.sql"
```

Results

The consolidated database can now be used with the new MobiLink server.

Related Information

[Synchronization Models](#)

[Upgrades of Version 10 and Later Databases \[page 90\]](#)

1.15.2.1.2 Upgrading a Consolidated Database (Adaptive Server Enterprise, SAP IQ, Oracle, MySQL, Microsoft SQL Server, or SAP HANA)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing consolidated database.

Prerequisites

You must be the owner of the currently installed MobiLink system tables to run the setup scripts.

ASE The MobiLink server login ID must have a SELECT privilege on MASTER..SYSTRANSACTIONS and MASTER..SYSPROCESSES, and also needs to have the dtm_tm_role role, if the MobiLink server command line option -cs is used.

SAP IQ The EXECUTE permission on SP_IQTRANSACTION is required by MobiLink server to use snapshot isolation for download, and the MONITOR system privilege is required in order to invoke the locking/blocking detection logic.

Oracle The RDBMS user that the MobiLink server uses to connect to the consolidated database must be able to use the MobiLink system tables, procedures, and so on, without any qualifiers (for example, SELECT * from m_user). The RDBMS user must also have SELECT privilege on GV\$TRANSACTION, GV\$SESSION, GV\$LOCK, and DBA_OBJECTS, and EXECUTE privileges on DBMS_UTILITY. You cannot grant permission directly for the GV\$TRANSACTION, GV\$SESSION and GV\$LOCK synonyms; you must instead grant permission on the underlying GV_\$TRANSACTION, GV_\$SESSION, and GV_\$LOCK dynamic performance views. You must connect as SYS to grant this access.

Microsoft SQL Server The RDBMS user that the MobiLink server uses to connect to the consolidated database must have permission to VIEW SERVER STATE, permission to SELECT from SYS.DATABASES, and permission to SELECT from SYS.DM_TRAN_LOCKS and SYS.PARTITIONS, SYS.SYSPROCESSES.

SAP HANA The HANA database user used by MobiLink server to connect to the consolidated database must have the CATALOG READ system permission.

Context

Only upgrade the MobiLink system objects in your SAP HANA consolidated database if your previous version of the MobiLink server is 17.0.0 or earlier.

You can also use the following methods to update the MobiLink system setup:

- In the MobiLink plug-in for SQL Central, click **MobiLink 17 > Project > Consolidated Databases** and right-click the database name and click *Check MobiLink System Setup*. If your database requires setup or upgrading, you are prompted to continue.
- When you use the *Deploy Synchronization Model Wizard*, system setup is checked when you connect to your consolidated database. If your database requires setup or upgrading, you are prompted to continue.

Procedure

1. For Adaptive Server Enterprise databases, set the SELECT INTO database option. Execute the following statement in Interactive SQL:

```
USE master
go
sp_dboption your-database-name, "SELECT INTO", true
go
USE your-database-name
go
checkpoint
go
```

2. Run the appropriate upgrade script for the version you are upgrading from.

The upgrade scripts are called `upgrade_XXX.sql`, where `XXX` indicates the RDBMS of your consolidated database. They are located under your SQL Anywhere installation in `MobiLink\upgrade\version`, where `version` is the MobiLink version you are upgrading from.

For example, to upgrade a Microsoft SQL Server database on which the MobiLink system tables from version 9.0.2 have been applied, run the following command:

```
osql -S server_name -U user_name -P password -i
"C:\Program Files\SQL Anywhere 17\MobiLink\upgrade\9.0.2\upgrade_mss.sql"
```

Results

The consolidated database can now be used with the new MobiLink server.

Related Information

[Synchronization Models](#)

1.15.2.1.3 Upgrading an IBM DB2 LUW Consolidated Database (Deprecated)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing IBM DB2 LUW consolidated database.

Prerequisites

You must be the owner of the currently installed MobiLink system tables to run the setup scripts.

For DB2 LUW, SELECT permission on SYSIBMADM.MON_LOCKWAITS and MON_GET_CONNECTION are required by the MobiLink server locking/blocking detection logic.

Context

You can also use the following methods to update the MobiLink system setup:

- In the MobiLink plug-in for SQL Central, click **► *MobiLink 17* ► *Project* ► *Consolidated Databases* ►** and right-click the database name and click *Check MobiLink System Setup*. If your database requires setup or upgrading, you are prompted to continue.
- When you use the *Deploy Synchronization Model Wizard*, system setup is checked when you connect to your consolidated database. If your database requires setup or upgrading, you are prompted to continue.

Procedure

1. Locate the IBM DB2 LUW upgrade script.

The upgrade script is called `upgrade_db2.sql` and is held in the `MobiLink/upgrade/` `version` subdirectory of your SQL Anywhere installation. The `version` directory refers to the version of MobiLink from which you are upgrading.

2. Copy `upgrade_db2.sql` and modify the copy. Change the CONNECT statement at the start of the script so it works with the instance you want to connect to. Apply the copied SQL script to the consolidated database.

Results

The consolidated database can now be used with the new MobiLink server.

Related Information

[IBM DB2 LUW Consolidated Database Synchronization Models](#)

1.15.2.1.4 Upgrading a Consolidated Database (SQL Anywhere Earlier Than 10.0.0)

Run upgrade scripts to install new system objects before using the new MobiLink server with an existing SQL Anywhere consolidated database that is earlier than version 10.0.0.

Prerequisites

If you have set up a SQL Anywhere consolidated database but never synchronized with it, then you must run the setup script (not the upgrade script). This step only applies to SQL Anywhere consolidated databases.

Context

- Before SQL Anywhere version 10.0.0, MobiLink system tables were owned by dbo. To run the setup scripts for a SQL Anywhere database, you must be logged in to the consolidated database as the owner of the MobiLink system tables. It is not enough to run these scripts as a user with permission to change the tables. To run the upgrade scripts, you can use the SETUSER SQL statement to impersonate dbo. For example:

```
SETUSER "dbo";
```

To upgrade a consolidated database in SQL Central, use the GRANT CONNECT statement to create a password for dbo and then connect as dbo. For example:

```
GRANT CONNECT TO dbo IDENTIFIED BY password;
```

In the latter case, after you have upgraded, use ALTER USER to remove the dbo password. For example:

```
ALTER USER TO dbo IDENTIFIED BY "";
```

Procedure

1. If you are upgrading a SQL Anywhere consolidated database that is earlier than version 10.0.0, you must first upgrade the database to version 17:
 - a. Shut down the database server.
 - b. Upgrade the database to version 17.
 - c. Start the database server, logging in as DBA.

Log in as DBA to upgrade.

2. Run the appropriate upgrade script for the version you are upgrading from.

The upgrade script is called `upgrade_asa.sql`. It is located under your SQL Anywhere installation in `MobiLink\upgrade\version`, where `version` is the SQL Anywhere version you are upgrading from.

To run the upgrade scripts, impersonate the `dbo` user by using the `SETUSER` SQL statement.

For example, to upgrade a SQL Anywhere version 9.0.2 consolidated database, connect to the database in Interactive SQL and execute the following statement:

```
SETUSER "dbo";
READ 'C:\Program Files\SQL Anywhere 17\MobiLink\upgrade\9.0.2\upgrade_asa.sql'
```

3. Remove the `dbo` password. For example:

```
GRANT CONNECT TO "dbo";
```

4. If you are running the MobiLink server as a user other than DBA, grant execute permission for that user on the new MobiLink system objects. Which system objects are new depends on which version you are upgrading from. The following code grants the necessary permissions to all MobiLink system objects. Before executing the code, change the user name `my_user` to the name of the user who is running the MobiLink server.

```
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;
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_dnl_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;

```

Results

The consolidated database can now be used with the new MobiLink server.

Related Information

[SQL Anywhere Consolidated Database](#)

[The Rebuild Process for Version 9 and Earlier Databases \[page 100\]](#)

1.15.2.2 MobiLink Server Upgrades

Upgrade the MobiLink server to version 17 if you are synchronizing version 17 remote databases.

Before using a version 17 MobiLink server, check the behavior changes to see if any affect you.

Version 17 of the MobiLink server only supports version 12 or later SQL Anywhere and UltraLite clients. If you must support earlier clients, keep an earlier version of the MobiLink server for supporting them.

1.15.2.3 SQL Anywhere MobiLink Client Upgrades

In a production environment, only upgrade SQL Anywhere remote databases after you have upgraded both the consolidated database and the MobiLink server. You only need to upgrade the MobiLink server if it is version 12 or earlier because version 16 and 17 clients can synchronize with a version 17 MobiLink server.

There are several kinds of upgrade to consider:

- Upgrading the software.
- Upgrading the remote database itself.
- Upgrading the whole application.

Upgrading the Software

Upgrade dbmlsync and the SQL Anywhere database server at the same time. Version 17 MobiLink clients can only synchronize version 17 databases running on version 17 database servers.

Version 17 MobiLink clients require a version 16 or later MobiLink server for synchronization. Version 17 MobiLink clients do not synchronize with a MobiLink server earlier than version 16.

Upgrading SQL Anywhere Remotes

You can upgrade MobiLink SQL Anywhere remote databases by running the Unload utility (dbunload) with the `-ar` option set.

When there is a schema change or other significant database change, you may need to perform a manual unload and reload.

Upgrading Applications

When deploying a new version of a MobiLink application, use a new script version for the synchronization scripts. For example, if the existing application uses a script version called v1, then the upgraded application could use a script version called v2. Both script versions can be in use at the same time, which makes it easier to upgrade the remote databases incrementally, rather than all at once.

For version 9.0.0 and later, the MobiLink server `-zd` option has been removed. If your deployment uses the `-zd` option and you want to upgrade, change your download scripts to accept the last download timestamp as the first parameter. Alternatively, you can upgrade your client and start using named parameters, which enable you to put script parameters in any order.

In this section:

[Unloading/Reloading a Remote SQL Anywhere Database Manually \[page 137\]](#)

When there is a schema change or other significant database change, you may need to perform a manual unload and reload.

Related Information

[SQL Anywhere Server Upgrades \[page 89\]](#)

[Unload Utility \(dbunload\)](#)

1.15.2.3.1 Unloading/Reloading a Remote SQL Anywhere Database Manually

When there is a schema change or other significant database change, you may need to perform a manual unload and reload.

Prerequisites

For version 16 and higher databases, you must have the following system privileges:

- BACKUP DATABASE
- VALIDATE ANY OBJECT
- SERVER OPERATOR
- SELECT ANY TABLE

Follow the standard precautions for upgrading software.

Procedure

1. Stop all database activity.
2. Perform a successful synchronization and validate and back up the remote database.
3. Run the dbtran utility to display the starting offset and ending offset of the database transaction log. Make note of the ending offset.
4. Rename the transaction log to ensure that it is not modified during the unload process. Move the renamed log file to a secure location, such as an offline directory.
5. Unload the database, without using any of the dbunload -a switches to automatically reload the database into another database.
6. Initialize a new database.
7. Reload the data into the new database using dbisql to read the `reload.sql` file generated by dbunload.
8. Shut down the new database.
9. Erase the new database's transaction log.
10. Run dblog on the new database, using the following options:
 - Use -z to specify the ending offset that you noted earlier.
 - Use -x to set the relative offset to zero.

For example:

```
dblog -x 0 -z 137829 database-name.db
```

11. Start dbmsync, specifying the location of the original log file that you moved earlier.
12. When you no longer need the old transaction log file, set the database option delete_old_logs.

Results

The remote SQL Anywhere database is unloaded and reloaded.

Related Information

[Upgrade and Rebuild Precautions \[page 93\]](#)

[Log Translation Utility \(dbtran\)](#)

[Unload Utility \(dbunload\)](#)

[Initialization Utility \(dbinit\)](#)

[Interactive SQL Utility \(dbisql\)](#)

[Transaction Log Utility \(dblog\)](#)

[MobiLink SQL Anywhere Client Utility \(dbmsync\) Syntax](#)

[delete_old_logs Option \[MobiLink\]\[SQL Remote\]](#)

1.15.3 UltraLite Upgrades

Before using existing UltraLite applications with this version of the software, be sure to review the list of new features and behavior changes to determine whether your application is affected.

Consider the following notes prior to upgrading to an UltraLite version 17 database:

- UltraLite version 17 cannot read UltraLite databases that were created using previous versions of the software.
- You cannot upgrade databases on devices.
- Once the database is upgraded, prior versions of applications, the utilities, and the software cannot connect to the database.
- If you have multiple versions of SQL Anywhere on your computer, pay attention to your system path to ensure that you are using the appropriate utilities. UltraLite utility names are the same between versions, and only the directory path name sets them apart from each other.

Upgrading Version 16 and Earlier Databases

UltraLite version 16 and earlier databases must be unloaded into a SQL or XML file using the version of the software that was used to create the database. Use the ulunload utility provided with that software.

If the older version of the software is not installed and you are using a Windows desktop operating system, you can use one of the older ulunload utilities that are included in SQL Anywhere version 17. Select the ulunload utility that is appropriate for your version of the UltraLite database.

Once the database is unloaded, use the version 17 ulload utility to load the XML file into a version 17 database, or the dbisql utility if you unloaded the database to a SQL file.

An alternative to creating a new database and using the ulload utility to populate it is to create a new database and synchronize it.

Compatibility with Existing Software

- UltraLite 17 database files only support connections from version 17 client applications or the version 17 UltraLite engine.
- The UltraLite version 17 runtime and the UltraLite version 17 engine do not work with database files created with UltraLite version 16 and earlier. To use application code created with version 16 and earlier, recompile it.

In this section:

[Upgrading an UltraLite Database \(Windows\) \[page 140\]](#)

Rebuild your UltraLite database for version 17 on Windows if you are upgrading from a previous version of the software.

[Upgrading an UltraLite Database \(Linux\) \[page 141\]](#)

Rebuild your UltraLite database for version 17 on Linux if you are upgrading from a previous version of the software.

Related Information

[What's New in UltraLite Version 17.0 \[page 80\]](#)

[How to Ensure That You Are Running the Correct Version of the Utilities When You Have Multiple Versions Installed \[page 94\]](#)

1.15.3.1 Upgrading an UltraLite Database (Windows)

Rebuild your UltraLite database for version 17 on Windows if you are upgrading from a previous version of the software.

Prerequisites

- If you are upgrading from an UltraLite version 9 or earlier database, you must have that version of SQL Anywhere installed.
- Make a backup copy of your existing UltraLite database.
- Synchronize your database if it is a production database that may contain unsynchronized changes.

Context

UltraLite version 17 cannot read UltraLite databases created using any prior version of the software.

Procedure

1. Create an XML file (or files) with the contents of the database.

Open a command prompt, navigate to the directory of the earlier SQL Anywhere install, and then use the `ulunload` utility on the database.

If the UltraLite database was created with version 11 or 12 of SQL Anywhere, you can, alternatively, open a command prompt, go to the `%SQLANY17%\UltraLite\Unload\` directory, choose the directory that applies to your version of the UltraLite database, such as `v11` or `v12`, and then use the `ulunload` utility from that directory.

2. Use the UltraLite version 17 ulload utility, or use the *Load Database Wizard* in the *UltraLite 17* plug-in for SQL Central to load the schema and data into a new version 17 database.

If you create an empty database and then load the XML by using the ulload utility, then verify the collation and encoding options for the database. UltraLite databases now default to the UTF8 encoding, and when you run ulload, the new database uses the encoding of the existing database.

3. Check the generated XML file to verify the setting of the UTF-8 encoding.

Results

The UltraLite database is upgraded to the latest version.

Related Information

[UltraLite utf8_encoding Creation Option](#)
[UltraLite Database Unload Utility \(ulunload\)](#)
[UltraLite Load XML to Database Utility \(ulload\)](#)

1.15.3.2 Upgrading an UltraLite Database (Linux)

Rebuild your UltraLite database for version 17 on Linux if you are upgrading from a previous version of the software.

Prerequisites

- Install the version of SQL Anywhere that created the database.
- Make a backup copy of your existing UltraLite database.
- Synchronize your database if it is a production database that may contain unsynchronized changes.

Context

UltraLite version 17 cannot read UltraLite databases created using any prior version of the software.

Procedure

1. Create an XML file (or files) with the contents of the database.

At a command prompt, navigate to the location of your earlier SQL Anywhere install, and then use the ulunload utility on the databases.

2. Use the UltraLite version 17 ulload utility, or use the [Load Database Wizard](#) in the *UltraLite 17* plug-in for SQL Central to load the schema and data into a new version 17 database.

UltraLite databases now default to the UTF8 encoding. If this encoding does not suit your needs, explicitly set the utf8_encoding parameter to off.

3. Check the generated XML file to verify the setting of the UTF-8 encoding.

Results

The UltraLite database is upgraded to the latest version.

Related Information

[UltraLite utf8_encoding Creation Option](#)
[UltraLite Database Unload Utility \(ulunload\)](#)
[UltraLite Load XML to Database Utility \(ulload\)](#)

1.15.4 SQL Remote Upgrades

SQL Remote upgrades can be performed one site at a time and your upgrade must account for features that were available in older versions of SQL Remote that are not supported in the latest version.

The upgrade requirements for SQL Remote are as follows:

Software upgrades can be performed one site at a time

Older Message Agents (dbremote) can exchange messages with version 17 Message Agents. For version 5 of SQL Remote, the version 5 Message Agents can exchange messages with version 17 Message Agents, as long as the compression database option is set to a value of -1. There is no need to upgrade software throughout the installation simultaneously.

Upgrade databases

If you are upgrading a remote or consolidated database that used SQL Anywhere version 9 or lower, you must upgrade the database file format by unloading and reloading your database. There is no need for all databases to be upgraded at the same time.

Upgrading Adaptive Server Enterprise consolidated databases

SQL Remote no longer supports Adaptive Server Enterprise consolidated databases. To synchronize Adaptive Server Enterprise databases, use MobilLink.

Upgrading with SQL Remote

Support for the VIM and MAPI message systems for SQL Remote was removed in version 11.0.0. When you upgrade a database that uses VIM or MAPI to SQL Anywhere version 17, you must change the message type to File, FTP, or SMTP. If the message type is MAPI or VIM, `dbremote.exe` does not start.

Example

One approach to upgrading version 5 of SQL Remote is as follows:

1. Upgrade the consolidated database server and SQL Remote Message Agent, and then upgrade the database file by unloading and reloading the consolidated database. Set the compression database option to -1, so that all messages are compatible with the version 5 software at remote sites.
2. One at a time, upgrade the remote database servers and Message Agents, and then upgrade the database file format by unloading and reloading the remote databases. You can set the compression database option to a value other than -1 to take advantage of compression and encoding on messages being sent to the consolidated database server.
3. When all remote database servers and Message Agents are upgraded, set the compression database option at the consolidated site to a value other than -1.

Related Information

[The Rebuild Process for Version 9 and Earlier Databases \[page 100\]](#)
[Rebuilding Databases Involved in Synchronization or Replication \(dbunload\) compression Option \[SQL Remote\]](#)

1.15.5 Upgrading the Monitor and Migrating Resources

Upgrade the monitor by installing the new software and migrating your resources. When upgrading the Monitor between major releases, use the software installer and follow the prompts to migrate your resources. When upgrading to a major release, install the software and run the Migrator utility to migrate the resources.

Prerequisites

i Note

Adobe will stop updating and distributing the Flash Player at the end of 2020. Because the SQL Anywhere Monitor is based on Flash, you cannot use it once Flash support ends. In many cases, tasks that were previously performed in the Monitor can be performed in the SQL Anywhere Cockpit. See [SQL Anywhere Monitor Non-GUI User Guide](#).

To avoid unwanted alerts, schedule a blackout period in the Monitor for your resources before you shut them down.

Context

Only one version of the Monitor can run on a computer at a time.

The default locations of the version 12, 16, and 17 Monitor database files are listed in the following table:

Monitor Production Edition

Operating system	Version 12 and 16 directories	Version 17 directories
Microsoft Windows	<code>C:\Users\Public\Documents\SQL Anywhere Monitor version-number\samonitor.db</code>	<code>C:\Users\Public\Documents\SQL Anywhere Monitor 17\samonitor.db</code>
Linux	<code>/opt/samonitorversion-number/samonitor.db</code>	<code>/opt/samonitor17/samonitor.db</code>

Monitor Developer Edition

Operating system	Version 12 and 16 directories	Version 17 directories
Microsoft Windows	<code>C:\Users\Public\Documents\SQL Anywhere version-number\Monitor\samonitor.db</code>	<code>C:\Users\Public\Documents\SQL Anywhere 17\Monitor\samonitor.db</code>
Linux	<code>/opt/sqlanywhereversion-number/samonitor.db</code>	<code>/opt/sqlanywhere17/samonitor.db</code>

Procedure

1. Create a back up copy of the existing Monitor database file, `samonitor.db`.
2. Install the new Monitor. Run the `setup.exe` file from the `Monitor` directory on your installation media, and follow the instructions provided. When the installation finishes, stop the new Monitor (if it is running).
3. At a command prompt, run the Migrator utility (`run_migrator`) with the following options.

-t temporary-directory

Specifies the directory for temporary files. By default, the temporary files are created in the directory where the Migrator utility exists.

i Note

The Monitor Migrator creates temporary files that are deleted at the end of the migration process. Use the `-t` option to specify a directory for these temporary files. The temporary files take up a similar amount of space as the old Monitor database file. Ensure that the specified directory has sufficient space.

source-filename

Specifies the path and file name to the old Monitor database file. For example, the path to the version 16 `samonitor.db` file.

destination-filename

Specifies the path and file name to the new Monitor file where the resources and configuration settings are loaded. For example, the path to the version 17 `samonitor.db` file.

For example:

```
C:\Program Files\SQL Anywhere 17\Bin64\run_migrator.cmd -t c:\monitorbackup C:\Users\Public\Documents\SQL Anywhere 16\Monitor\samonitor.db C:\Users\Public\Documents\SQL Anywhere 17\Monitor\samonitor.db
```

Use the following table to determine the location of the Migrator utility (`run_migrator`).

Operating system	Monitor type
Microsoft Windows	<code>C:\Program Files\SQL Anywhere 17\Bin??\run_migrator.cmd</code>

Operating system	Monitor type
Linux	<code>/opt/sqlanywhere17/bin??/run_migrator.sh</code>

Results

The Monitor is upgraded.

Next Steps

Related Information

[Starting the Monitor](#)

1.15.6 Upgrading Version 16 Databases to Use the SQL Anywhere Cockpit

Create the COCKPIT_ROLE user-defined role that users require to connect to the Cockpit, and then grant users its exercise rights.

Prerequisites

You must have the MANAGE ROLES system privilege to create a new role and grant it to users.

Context

The Cockpit monitors the database server and consequently captures and displays information about all databases running on that server. To connect to the Cockpit you must have the user ID and password of a user from one of the databases running on the server. This user must have the exercise rights to the COCKPIT_ROLE user-defined role. By default, this role is not defined in version 16 databases, so you must create the COCKPIT_ROLE user-defined role and then grant its exercise rights to your users.

The Cockpit does not support databases that use the legacy definer security model. If your database uses this security model, you cannot connect to the Cockpit even if you have the user-defined COCKPIT_ROLE role.

Procedure

1. Start the version 16 database on a version 17 database server.
2. Start Interactive SQL and connect to the database.
3. Create the COCKPIT_ROLE user-defined role by executing the following statements.

```
CREATE ROLE COCKPIT_ROLE;  
GRANT MONITOR, DROP CONNECTION, BACKUP DATABASE, SERVER OPERATOR TO  
COCKPIT_ROLE;
```

The COCKPIT_ROLE is created. The MANAGE ROLES system privilege is automatically granted administrative rights (only) to this new role. This way, any user with the MANAGE ROLES system privilege can administer the role.

4. Grant exercise rights for the COCKPIT_ROLE to the users who need to use the Cockpit

For example, execute the following statement to grant the COCKPIT_ROLE to the user JohnDoe:

```
GRANT ROLE COCKPIT_ROLE TO JohnDoe;
```

5. Start the Cockpit on the database server that runs the version 16 database.

Related Information

[SQL Anywhere Cockpit](#)

[Starting and Connecting to the Cockpit\(SQL Central\)](#)

[COCKPIT_ROLE User-defined Role](#)

1.15.7 Upgrading Version 17 Databases to Use the SQL Anywhere Cockpit

Grant the ACCESS DISK INFORMATION system privilege your version 17 users to allow them to have full functionality of the features in the version 17 Cockpit.

Prerequisites

You must have the MANAGE ROLES system privilege to add a role to the COCKPIT_ROLE user defined role.

Context

The Cockpit monitors the database server and consequently captures and displays information about all databases running on that server. To connect to the Cockpit you must have the user ID and password of a user

from one of the databases running on the server. This user must have the exercise rights to the COCKPIT_ROLE user-defined role.

To have access to the full functionality in the Cockpit, the user must also have the following system privileges:

- MONITOR system privilege
- DROP CONNECTION system privilege
- BACKUP DATABASE system privilege
- SERVER OPERATOR system privilege
- ACCESS DISK INFORMATION system privilege

In version 17.0 build 2000 and later, the COCKPIT_ROLE user-defined role is by default populated with these system privileges. If you are using an earlier version of the software, then your COCKPIT_ROLE user-defined role could be missing the ACCESS DISK INFORMATION system privilege. This system privilege is needed to view information about low disk space alert.

To allow your users full functionality in the Cockpit, grant them exercise rights to the ACCESS DISK INFORMATION system privilege.

Procedure

1. Start the version 17 database on a version 17 database server.
2. Start Interactive SQL and connect to the database.
3. Grant the ACCESS DISK INFORMATION system privilege to your users:

Option	Action
Add the ACCESS DISK INFORMATION system privilege to the COCKPIT_ROLE user-defined role, so that all users with this role have the ACCESS DISK INFORMATION system privilege.	Execute the following: <pre>GRANT ACCESS DISK INFORMATION to COCKPIT_ROLE;</pre>
Grant individual Cockpit users, the ACCESS DISK INFORMATION system privilege.	For example: <pre>GRANT ACCESS DISK INFORMATION to userA;</pre>

Related Information

[SQL Anywhere Cockpit](#)
[Starting and Connecting to the Cockpit\(SQL Central\)](#)
[COCKPIT_ROLE User-defined Role](#)

1.15.8 Converting the OData Server from Version 16

Use SQL statements to create ODATA PRODUCER definitions, and then use the `dbsrv17 -xs odata` protocol option to start the enabled and defined ODATA PRODUCER statements.

Procedure

1. Start the OData server by running the database server with the `-xs odata` option, and passing the version 16 embedded HTTP server options as network protocol options for the OData protocol.

For example, assume that the following options are defined in the version 16 OData Server configuration file:

```
# Embedded HTTP server options
# -----
LogFile = ../../odata.log
LogVerbosity = 1
ServerPort = 8000
ShutdownListenerPort = 8083
SSLKeyStore = ../../samplekeystore.jks
SSLKeyStorePassword = pwd
Producers = OrderEntryProducer, StaffingProducer
```

Run the following command to replicate the behavior:

```
dbsrv17 -xs odata(SecureServerPort=8000;SSLKeyStore=../../
samplekeystore.jks;ServerPort=0;SSLKeyStorePassword=pwd;LogFile=../../
odata.log;LogVerbosity=1)
```

i Note

Unless HTTP listening is turned off via the `SecureOnly` protocol, the OData server listens on both HTTP and HTTPS ports when key store information is provided.

The *Producers* and *ShutdownListenerPort* options are no longer required. Specified *Producers* are enabled with the `CREATE ODATA PRODUCER ENABLED` clause.

2. Start Interactive SQL and connect to the database.
3. Replicate the OData Producers defined in the version 16 OData Server configuration file with the `CREATE ODATA PRODUCER` statement.

For example, assume that the following options are defined in the version 16 OData Server configuration file:

```
# Shared OData Producer options
# -----
Authentication = none
ConnectionPoolMaximum = 10
DbProduct = sqlanywhere
PageSize = 100
ReadOnly = false
# OrderEntryProducer options
# -----
OrderEntryProducer.Model = ../../ordermodel.osdl
OrderEntryProducer.ModelConnectionString =
uid=dba;pwd=sql;ServerName=orderentry;dbf=orderentry.db
```

```
OrderEntryProducer.ServiceRoot = /orders/  
OrderEntryProducer.DbConnectionString =  
uid=dba;pwd=sql;ServerName=orderentry;dbf=orderentry.db
```

Create the following statement to define an OData Producer that replicates its properties:

```
CREATE ODATA PRODUCER OrderEntryProducer  
ENABLED  
MODEL FILE '../..//ordermodel.osdl'  
SERVICE ROOT '/orders/'  
USING 'ConnectionPoolMaximum=10;PageSize=100;ReadOnly=false;
```

The *ModelConnectionString*, *DbProduct*, and *DbConnectionString* options are no longer required.

Related Information



[-xs Database Server Option](#)
[Network Protocol Options](#)
[CREATE ODATA PRODUCER Statement](#)

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
 - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
 - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

© 2022 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.