What's New in SQL Anywhere® Studio

Part number: 38121-01-0900-01

Last modified: June 2003

Contents

About This Manual

Subject	This book describes new features in SQL Anywhere Studio version 9 and in previous releases of the software.
Audience	This manual is for users of previous versions who want to find out what is new and different in this release of the software.

SQL Anywhere Studio documentation

This book is part of the SQL Anywhere documentation set. This section describes the books in the documentation set and how you can use them.

The SQL Anywhere Studio documentation

The SQL Anywhere Studio documentation is available in a variety of forms: in an online form that combines all books in one large help file; as separate PDF files for each book; and as printed books that you can purchase. The documentation consists of the following books:

- **Introducing SQL Anywhere Studio** This book provides an overview of the SQL Anywhere Studio database management and synchronization technologies. It includes tutorials to introduce you to each of the pieces that make up SQL Anywhere Studio.
- **What's New in SQL Anywhere Studio** This book is for users of previous versions of the software. It lists new features in this and previous releases of the product and describes upgrade procedures.
- **Adaptive Server Anywhere Getting Started** This book is for people new to relational databases or new to Adaptive Server Anywhere. It provides a quick start to using the Adaptive Server Anywhere database-management system and introductory material on designing, building, and working with databases.
- **Adaptive Server Anywhere Database Administration Guide** This book covers material related to running, managing, and configuring databases and database servers.
- **Adaptive Server Anywhere SQL User's Guide** This book describes how to design and create databases; how to import, export, and modify data; how to retrieve data; and how to build stored procedures and triggers.
- **Adaptive Server Anywhere SQL Reference Manual** This book provides a complete reference for the SQL language used by Adaptive Server Anywhere. It also describes the Adaptive Server Anywhere system tables and procedures.
- **Adaptive Server Anywhere Programming Guide** This book describes how to build and deploy database applications using the C, C++, and Java programming languages. Users of tools such as Visual Basic and PowerBuilder can use the programming interfaces provided by those tools. It also describes the Adaptive Server Anywhere ADO.NET data provider.

- **Adaptive Server Anywhere Error Messages** This book provides a complete listing of Adaptive Server Anywhere error messages together with diagnostic information.
- **SQL Anywhere Studio Security Guide** This book provides information about security features in Adaptive Server Anywhere databases. Adaptive Server Anywhere 7.0 was awarded a TCSEC (Trusted Computer System Evaluation Criteria) C2 security rating from the U.S. Government. This book may be of interest to those who wish to run the current version of Adaptive Server Anywhere in a manner equivalent to the C2-certified environment.
- **MobiLink Synchronization User's Guide** This book describes how to use the MobiLink data synchronization system for mobile computing, which enables sharing of data between a single Oracle, Sybase, Microsoft or IBM database and many Adaptive Server Anywhere or UltraLite databases.
- **MobiLink Synchronization Reference** This book is a reference guide to MobiLink command line options, synchronization scripts, SQL statements, stored procedures, utilities, system tables, and error messages.
- **iAnywhere Solutions ODBC Drivers** This book describes how to set up ODBC drivers to access consolidated databases other than Adaptive Server Anywhere from the MobiLink synchronization server and from Adaptive Server Anywhere remote data access.
- **SQL Remote User's Guide** This book describes all aspects of the SQL Remote data replication system for mobile computing, which enables sharing of data between a single Adaptive Server Anywhere or Adaptive Server Enterprise database and many Adaptive Server Anywhere databases using an indirect link such as e-mail or file transfer.
- **SQL Anywhere Studio Help** This book includes the context-sensitive help for Sybase Central, Interactive SQL, and other graphical tools. It is not included in the printed documentation set.
- **UltraLite Database User's Guide** This book is intended for all UltraLite developers. It introduces the UltraLite database system and provides information common to all UltraLite programming interfaces.
- **UltraLite Interface Guides** A separate book is provided for each UltraLite programming interface. Some of these interfaces are provided as UltraLite components for rapid application development, and others are provided as static interfaces for C, C++, and Java development.

In addition to this documentation set, PowerDesigner and InfoMaker include their own online documentation.

Documentation formats

SQL Anywhere Studio provides documentation in the following formats:

- **Online documentation** The online documentation contains the complete SQL Anywhere Studio documentation, including both the books and the context-sensitive help for SQL Anywhere tools. The online documentation is updated with each maintenance release of the product, and is the most complete and up-to-date source of documentation.

  To access the online documentation on Windows operating systems, choose Start ➤ Programs ➤ SQL Anywhere 9 ➤ Online Books. You can navigate the online documentation using the HTML Help table of contents, index, and search facility in the left pane, as well as using the links and menus in the right pane.

  To access the online documentation on UNIX operating systems, see the HTML documentation under your SQL Anywhere installation.

- **Printable books** The SQL Anywhere books are provided as a set of PDF files, viewable with Adobe Acrobat Reader.

  The PDF files are available on the CD ROM in the *pdf_docs* directory. You can choose to install them when running the setup program.

- **Printed books** The complete set of books is available from Sybase sales or from eShop, the Sybase online store. You can access eShop by clicking How to Buy ➤ eShop at *http://www.ianywhere.com*.

Documentation conventions

This section lists the typographic and graphical conventions used in this documentation.

Syntax conventions

The following conventions are used in the SQL syntax descriptions:

- **Keywords** All SQL keywords appear in upper case, like the words ALTER TABLE in the following example:

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

- **Placeholders** Items that must be replaced with appropriate identifiers or expressions are shown like the words *owner* and *table-name* in the following example:

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

- **Repeating items** Lists of repeating items are shown with an element of the list followed by an ellipsis (three dots), like *column-constraint* in the following example:

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

  One or more list elements are allowed. In this example, if more than one is specified, they must be separated by commas.

- **Optional portions** Optional portions of a statement are enclosed by square brackets.

  RELEASE SAVEPOINT [*savepoint-name*]

  These square brackets indicate that the *savepoint-name* is optional. The square brackets should not be typed.

- **Options** When none or only one of a list of items can be chosen, vertical bars separate the items and the list is enclosed in square brackets.

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

  For example, you can choose one of ASC, DESC, or neither. The square brackets should not be typed.

- **Alternatives** When precisely one of the options must be chosen, the alternatives are enclosed in curly braces and a bar is used to separate the options.

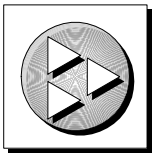
  [**QUOTES** { **ON** | **OFF** }]

  If the QUOTES option is used, one of ON or OFF must be provided. The brackets and braces should not be typed.

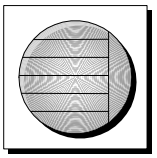
Graphic icons

The following icons are used in this documentation.

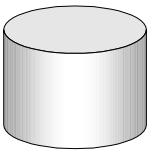
- A client application.



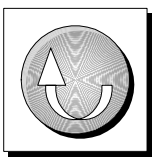
- A database server, such as Sybase Adaptive Server Anywhere.



- A database. In some high-level diagrams, the icon may be used to represent both the database and the database server that manages it.



- Replication or synchronization middleware. These assist in sharing data among databases. Examples are the MobiLink Synchronization Server and the SQL Remote Message Agent.



- A programming interface.

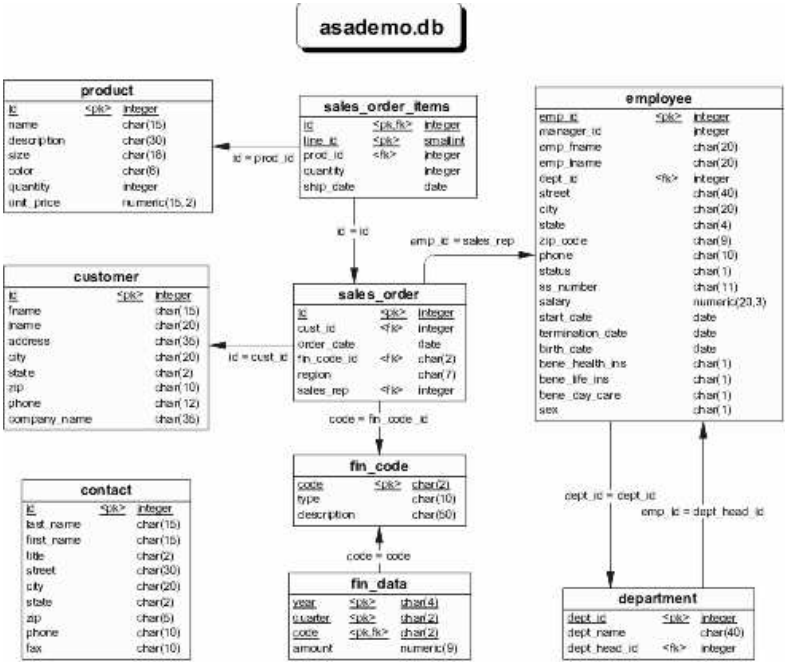
The Adaptive Server Anywhere sample database

Many of the examples throughout the documentation use the Adaptive Server Anywhere sample database.

The sample database is held in a file named *asademo.db*, and is located in your SQL Anywhere directory.

The sample database represents a small company. It contains internal information about the company (employees, departments, and finances) as well as product information and sales information (sales orders, customers, and contacts). All information in the database is fictional.

The following figure shows the tables in the sample database and how they relate to each other.



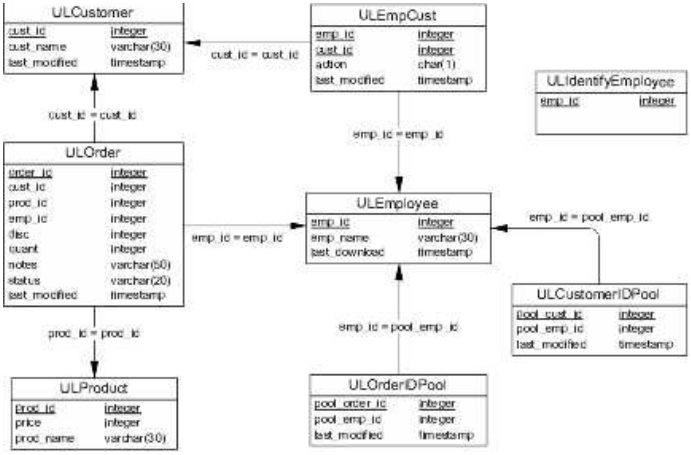
The CustDB sample database

Many of the examples in the MobiLink and UltraLite documentation use the UltraLite sample database.

The reference database for the UltraLite sample database is held in a file named *custdb.db*, and is located in the *Samples\UltraLite\CustDB* subdirectory of your SQL Anywhere directory. A complete application built on this database is also supplied.

The sample database is a sales-status database for a hardware supplier. It holds customer, product, and sales force information for the supplier.

The following figure shows the tables in the CustDB database and how they are related to each other.



Finding out more and providing feedback

We would like to receive your opinions, suggestions, and feedback on this documentation.

You can provide feedback on this documentation and on the software through newsgroups set up to discuss SQL Anywhere technologies. These newsgroups can be found on the *forums.sybase.com* news server.

The newsgroups include the following:

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

Newsgroup disclaimer

iAnywhere Solutions has no obligation to provide solutions, information or ideas on its newsgroups, nor is iAnywhere Solutions obliged to provide anything other than a systems operator to monitor the service and insure its operation and availability.

iAnywhere Solutions Technical Advisors as well as other staff assist on the newsgroup service when they have time available. They offer their help on a volunteer basis and may not be available on a regular basis to provide solutions and information. Their ability to help is based on their workload.

CHAPTER 1

Welcome to SQL Anywhere Studio Version 9

About this chapter

This chapter provides a high-level overview of the new features in this release of SQL Anywhere Studio.

☞ For a complete listing of new features, see "What's New in Version 9.0.0" on page 13.

Contents

Overview

SQL Anywhere Studio is a complete relational database management system. It includes the following major components:

- **Adaptive Server Anywhere** A self-managing relational database with high reliability, high performance out of the box, and a full range of SQL features. It is available for Windows, Windows CE, Linux, NetWare, and several flavors of UNIX.
- **UltraLite** A database system for small devices, including Palm OS, Pocket PC, and Java-based devices. Custom built for restricted environments, the UltraLite database management system can run in less than 100K while providing full transaction-processing support, a choice of development models, and synchronization with enterprise data stores.
- **MobiLink** A synchronization technology for sharing information among relational databases. It integrates physically separate databases into a single information system, maintaining the integrity of transactions across the entire system.
- **SQL Remote** A synchronization technology specifically for message-based synchronization. Dial-up links, e-mail, and ftp systems can be used to connect your databases.

SQL Anywhere Studio has many uses in the small and medium business (SMB) market, in mobile computing, and in embedded systems.

- **SMB server** As a self-tuning, easy to manage, scalable and secure database server, Adaptive Server Anywhere is an ideal choice for an SMB database server. It scales all the way from handheld devices to databases in the tens of gigabytes accessed by hundreds of concurrent connections.
- **Mobile computing** Adaptive Server Anywhere runs well in environments with restricted resources, making it ideal for desktop and laptop use, as well as on Windows CE devices. UltraLite provides a database with an even smaller footprint for Windows CE and Palm OS devices. MobiLink provides a scalable and secure synchronization system that ties many remote databases together into a single integrated system.
- **Embedded systems** Adaptive Server Anywhere and UltraLite have self-management features that make them ideal for use in zero-administration environments such as embedded systems. MobiLink's server-initiated synchronization (new in this release) makes it possible to synchronize embedded databases from a central location.

Version 9	SQL Anywhere Studio version 9 brings a host of new features to the product, while maintaining the ease of use and self-management features for which it is known.
Web services and .NET	With this release, SQL Anywhere Studio introduces XML and web services features. A web server is built directly in to the database. A flexible interface lets you retrieve data as XML and store XML data in the database. Integration with Microsoft Visual Studio .NET rounds out the web services offerings. Version 9 takes advantage of the Microsoft .NET architecture. With the XML and web services features, an ADO.NET provider for Adaptive Server Anywhere, a .NET scripting capability for MobiLink synchronization, and an UltraLite .NET component for development of handheld database applications, SQL Anywhere is built to work with .NET.
Rapid application development for small devices	The UltraLite .NET component is just one of a suite of UltraLite components that makes it easy to develop powerful database applications for small devices. Other components provide integration with rapid application development from a number of tools, including Pocket IE, eMbedded Visual Basic, AppForge MobileVB, and Borland JBuilder.
MobiLink synchronization	MobiLink synchronization ties UltraLite or Adaptive Server Anywhere databases into an enterprise system, using configurable two-way synchronization. In this release, MobiLink provides such enhancements as server-initiated synchronization and file-based downloads.
Performance	Version 9 continues to improve performance in the Adaptive Server Anywhere database. Improved cache management and other internal changes provide scalability to large numbers of users and more processors, and innovative algorithms in the optimizer provide better handling of complex queries. What's more, a 64-bit version of the software is available or planned for Linux, HP-UX, and for 64-bit Windows operating systems. For those who really need to get the most performance out of their database, a graphical Index Consultant makes performance tuning easy. This is just one feature included in a redesigned Sybase Central administration tool. The new look puts more information within easy reach than ever before, and provides a suite of powerful design and administration tools, including the query editor, and a redesigned stored procedure debugger.

Adaptive Server Anywhere highlights

Here are some of the highlights in this release of the Adaptive Server Anywhere database.

- **Scalability and performance** In this release, Adaptive Server Anywhere has taken more major strides in scalability and performance. A redesign of the cache management system makes for much-improved concurrency at high numbers of users. Customers already run Adaptive Server Anywhere with several hundred simultaneous users, and this release looks to raise the bar on the scale that can be handled.

  Query processing has also taken another big step forward. Innovative new algorithms make many queries run in a fraction of the time of previous releases.

  While many of the performance improvement changes are built into the internals of the database server, a new Index Consultant provides administrators with an easy-to-use way of optimizing performance by guiding them through the selection of appropriate indexes.

- **Web services** This release sees a web server built right into to the database. You can now access Adaptive Server Anywhere databases over HTTP as well as the usual client/server network protocols. Adaptive Server Anywhere now provides web services, including support for SOAP requests. Close integration into Microsoft Visual Studio .NET makes the development of web services applications straightforward.

- **XML in the database** Client applications can now get result sets as XML, or can take XML data and store it in the database in relational form. Developers have control over the format of the XML data, which eases integration into other systems.

- **.NET integration** Adaptive Server Anywhere version 9 is built to work with Microsoft's .NET architecture. In addition to the web services and XML features, an ADO .NET provider enables data access from Visual Studio .NET.

- **Even richer SQL** Adaptive Server Anywhere's SQL language has been extended to enable you to do more with your data. You can carry out simple OLAP features such as ROLLUP queries, execute recursive union queries to get at hierarchical "bill of materials" data, and carry out set operations such as INTERSECT and EXCEPT.

☞ For a complete list of new features in Adaptive Server Anywhere, see "Adaptive Server Anywhere new features" on page 14.

UltraLite highlights

UltraLite is a relational database for small devices running Palm OS or Windows CE. This version brings the benefits of relational data storage to a wider range of application developers than ever.

- **UltraLite components** Components provide integration with popular programming languages and tools. UltraLite now provides components for the following development platforms:
  - An ActiveX component brings UltraLite development to Microsoft eMbedded Visual Basic and Pocket IE (using JavaScript).
  - An AppForge MobileVB component brings UltraLite development to this powerful extension to Visual Basic. AppForge MobileVB brings rapid application development using Visual Basic to both Palm OS and Windows CE devices.
  - A .NET component brings UltraLite development in either C# or Visual Basic to the .NET Compact Framework.
  - A Native Java component provides a Java development model for small devices.
  - A C++ interface makes features of other UltraLite components accessible to C++ developers, including access to a version of the UltraLite runtime that supports access from multiple applications.
- **Dynamic SQL** UltraLite components provide a table-based API for straightforward data access, and also support dynamic SQL for more flexible operations, including multi-table joins.

For a complete list of new features in UltraLite, see "UltraLite new features" on page 34.

MobiLink highlights

- **Server-initiated synchronization** In previous releases of the software, synchronization had to be initiated by the client. It is now possible to initiate synchronization from the server, making it possible to push urgent changes to remote databases. Server-initiated synchronization is also ideal when the client database is running in an embedded system.
- **File-based download** Synchronization updates can now be distributed as files, enabling offline distribution of synchronization changes.
- **More selective synchronization** Adaptive Server Anywhere clients can now choose upload-only and download-only synchronization. In addition, new scripts provide additional control over the synchronization process.

☞ For a complete list of new features in MobiLink, see "MobiLink new features" on page 30.

Administration tools highlights

Sybase Central is the administration tool for Adaptive Server Anywhere databases and MobiLink consolidated databases. This release includes several new features:

- **Interface redesign** The interface has been redesigned to bring you more quickly to the information you need.
- **Stored procedure debugger** The stored procedure debugger has been integrated into Sybase Central and redesigned for easier use.
- **Faster start-up** Quicker start-up times make access to Sybase Central and Interactive SQL more convenient.

☞ For a complete list of new features in the administration tools, see "Adaptive Server Anywhere new features" on page 14.

New feature samples

These and other samples can be found at *http://ianywhere.codexchange.-sybase.com/servlets/ProjectDocumentList?folderID=0.*

UltraLite samples

The UltraLite samples are located at *http://ianywhere.codexchange.sybase.-com/servlets/ProjectDocumentList?folderID=283*

UL.NET sample	This sample demonstrates the new UltraLite.NET component. In this sample, you create an UltraLite database based on a .USM (UltraLite schema file). The schema file stores database information such as tables, columns, indexes, and so on. You do not alter the schema of an UltraLite database directly. Instead, you modify a schema file (which typically has the file extension .USM) and upgrade the database schema from that file using a built-in UltraLite function in your application. After the database is created, data is downloaded through MobiLink. Finally, a SQL query, which displays a resultset in a list box, is executed.
ULFoundation	ULFoundation was developed as a base platform on which to start the development of a new UltraLite application and as a learning tool. The goal of this project is to help developers understand the portions of an application that directly relate to UltraLite. Platform-specific functionality has been kept to a minimum. The code has been written to be platform-independent where possible. Additional modules are being developed that guide you through adding new features to the basic ULFoundation program. This is an effective tool for demonstrating how UltraLite works and can be used in a variety of applications. There are many ways to go about setting up ULFoundation. Following the instructions in the text files are recommended, but not necessary.
UltraLite Dynamic SQL sample	This sample illustrates the addition of Dynamic SQL queries within the UltraLite Component Suite. For this sample, an application was written using the UltraLite.NET API. During the sample, an application is deployed to a CE device along with a pre-created UltraLite database. From this sample, a user can type in any valid SQL SELECT query and the results appear in a list box.
UltraLite Engine sample	In this sample, two applications (ULapp1 and ULapp2) access the same database on your device using the new multi-process access feature. The first application creates a database. The UltraLite engine starts, allowing you to open the second application, which also connects to the newly created database. The engine remains open as long as at least one of the two applications is open. The engine shuts down automatically when both

applications are closed.

MobiLink samples

The MobiLink samples are located at *http://ianywhere.codexchange.sybase.-com/servlets/ProjectDocumentList?folderID=319*

.NET Remoting sample

.NET remoting enables you to build widely distributed applications easily, whether application components are all on one computer or spread out across the world. You can build client applications that use objects in other processes on the same computer or on any other computer that is reachable over its network. You can also use .NET remoting to communicate with other application domains in the same process.

Reference: *http://msdn.microsoft.com/library/default.asp?url=/library*

The following Visual Basic.NET and C# samples use MobiLink to invoke a .NET remoting service. The *server.cs/servclass.cs* files produce a simple remoting service. This service opens a local port on 1095 and listens to requests. When methods of the remoting service are invoked the remoting server posts messages to the console

ASP.NET sample

ASP.NET is a component of the .NET Framework for building, deploying, and running web applications and distributed applications. ASP.NET provides support for web services through the use of .ASMX files. A web service is a protocol that enables computers to work together by exchanging messages. Web services are based on the standard protocols of XML, SOAP, and WSDL, which allow them to interoperate across platforms and programming languages.

This sample provides integration between MobiLink and ASP.NET through the invocation of web services. In this sample, the file *sample.asmx* is a sample web service that is copied into IIS home/root directory (likely *c:\inetpup\wwwroot*). During synchronization, the MobiLink server attempts to invoke ASP.NET, which then attempts to write to a file name *data.bin* in the IIS home/root directory.

File Sync sample

This sample demonstrates the capabilities of the MobiLink server to send a file that contains a list of changes to be applied to a remote database. The remote database takes this file, applies it, and lets you verify that the changes were applied.

Adaptive Server Anywhere samples

The Adaptive Server Anywhere samples are located at *http://ianywhere.-codexchange.sybase.com/servlets/ProjectDocumentList?folderID=282*

ADO.NET - Simple XML sample	The Simple XML example provides sample code to obtain XML data from Adaptive Server Anywhere via ADO.NET.
ASP.NET tutorial	This tutorial is intended to give a brief understanding of how to use ASP.NET to access an Adaptive Server Anywhere 9.0 database using ADO.NET. For this example, you can use either Visual Studio.NET 2002 or 2003, with Visual Basic.NET as the development tool. In this tutorial, we walk through all of the steps required to build a sample ASP.NET application to query the Adaptive Server Anywhere database and return the results into a list box all within the browser.
HTTP Server - Show Table sample	This Show Table sample is an example of a web-based administration tool that displays database information, such as: ♦ Table list ♦ Table contents ♦ Web services available ♦ Database history This sample helps illustrate the capabilities of the HTTP server to serve HTML and XML data using XML style sheets. During the demonstration, two Adaptive Server Anywhere web services are created: 1. **show_html_table** 2. **show_xml_table**
HTTP Server - Simple Select sample	This sample shows how easy it is to create a web service within Adaptive Server Anywhere that takes any SQL query and returns a resultset in either HTML or XML format. The SQL commands executed to create this service include: ♦ CREATE SERVICE qhtml TYPE ‘html’ AUTHORIZATION ON USER DBA SECURE OFF ♦ CREATE SERVICE qxml TYPE ‘xml’ AUTHORIZATION ON USER DBA SECURE OFF When the user enters a URL (for example, *http://localhost/websample/qhtml?select * from sysusers*), the Adaptive Server Anywhere web server receives the URL and parses the text. This query is executed and as the qhtml service specifies, it returns the results in HTML format.
HTTP Server - SOAP Dataset sample	This sample illustrates how data returned in a SOAP request from Adaptive Server Anywhere can easily be displayed in a .NET DataGrid component.

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML-based protocol that consists of three parts:

- an envelope that defines a framework for describing what is in a message and how to process it
- a set of encoding rules for expressing instances of application-defined data types
- a convention for representing remote procedure calls and responses

SOAP can potentially be used in combination with a variety of other protocols; however, the only bindings defined in this document describe how to use SOAP in combination with HTTP and HTTP Extension Framework.

The DataGrid is a commonly used data bound list control available within Visual Studio.NET 2003. The DataGrid displays items from a data source in a table that lets you select, sort, and edit these items. The SOAP request from the application connects to the Adaptive Server Anywhere 9.0 Sample database and the results are returned as a .NET DataSet object. Next, the DataSet is used as the DataSource for the DataGrid component. To walk through the demonstration from scratch using the source code, see the file *Walkthrough.doc*.

CHAPTER 2

What's New in Version 9.0.0

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in SQL Anywhere Studio version 9.

Contents

New features in version 9.0.0

This section lists the new features introduced in components of SQL Anywhere Studio version 9.0.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 9.0. It provides an exhaustive listing of major and minor new features, with cross references to locations where each feature is discussed in detail.

Highlighted new features

- **XML support** Adaptive Server Anywhere 9.0 includes a broad range of support for XML, including storing XML documents, exporting relational data as XML, importing XML, and returning XML from queries on relational data.
  - **FOR XML clause** The SELECT statement supports a FOR XML clause with three modes, RAW, AUTO, and EXPLICIT, that allow you to obtain query results as an XML document. Each mode allows you a different level of control over the format of the XML that is generated.

    ☞ For more information, see "Obtaining query results as XML" [*ASA SQL User's Guide*, page 490] and "SELECT statement" [*ASA SQL Reference*, page 541].
  - **FOR_XML_NULL_TREATMENT option** You can use the FOR_XML_NULL_TREATMENT option to control how NULL values are returned by a query that includes the FOR XML clause.

    ☞ For more information, see "FOR_XML_NULL_TREATMENT option [database]" [*ASA Database Administration Guide*, page 594].
  - **OPENXML procedure** For more information, see "OPENXML function [String]" [*ASA SQL Reference*, page 165].
  - **SQL/XML support** SQL/XML is a draft standard that describes the ways SQL can be used in conjunction with XML. As part of its SQL/XML support, Adaptive Server Anywhere includes an XML data type that can be used to store XML documents in the database.

    ☞ For more information, see "XML data type [Character]" [*ASA SQL Reference*, page 55].

    Adaptive Server Anywhere also supports the following SQL/XML functions that provide an alternative method to the FOR XML clause for generating XML documents from your relational data:
    - **XMLAGG function** This aggregate function generates a forest of XML elements from a collection of XML elements.

      ☞ For more information, see "XMLAGG function [String]" [*ASA SQL Reference*, page 205].

- **XMLCONCAT function** This function generates a forest of XML elements by concatenating together the XML values that are passed in to it.
  ☞ For more information, see "XMLCONCAT function [String]" [*ASA SQL Reference*, page 206].
- **XMLELEMENT function** This function generates an XML element for which you can optionally specify element content, attributes, and attribute content.
  ☞ For more information, see "XMLELEMENT function [String]" [*ASA SQL Reference*, page 207].
- **XMLFOREST function** This function generates a forest of XML elements.
  ☞ For more information, see "XMLFOREST function [String]" [*ASA SQL Reference*, page 208].
- **XMLGEN function** This function generates an XML value based on an XQuery Constructor.
  ☞ For more information, see "XMLGEN function [String]" [*ASA SQL Reference*, page 209].

♦ **HTTP server in the database** Adaptive Server Anywhere database servers can now act as web servers, allowing you to write and run web-based applications using only an Adaptive Server Anywhere database and a web browser of your choice.

This feature allows the database server to handle standard HTTP and HTTPS requests, as well as standard SOAP requests. Service types available are HTTP, HTTPS, XML, RAW, SOAP, and DISH. DISH is a SOAP service handler.

To gain the benefits of this enhancement on databases created before this release, you must upgrade the database using the Upgrade utility.

☞ For more information, see "Using the Built-in Web Server" [*ASA Database Administration Guide*, page 219].

♦ **Index Consultant** The Index Consultant is a tool to assist you in proper selection of indexes. It analyzes either a single query or a set of operations, and recommends indexes to add to your database and to remove from the database.

To gain the benefits of this enhancement on databases created before this release, you must upgrade the database using the Upgrade utility.

☞ For more information, see "Index Consultant overview" [*ASA SQL User's Guide*, page 63].

♦ **64-bit version available** A full 64-bit version of the software is available for Windows Server 2003 on Itanium II chips. A deployment

release is available on 64-bit Linux and HP-UX operating systems.

For details of platform support, see "SQL Anywhere Studio Supported Platforms" [*Introducing SQL Anywhere Studio*, page 121].

SQL enhancements

- **The WITH clause can now be used before a select to specify common table expressions** Common table expressions are temporary view definitions that exist only within the scope of a SELECT statement. They can be recursive, or non-recursive. They sometimes let you write queries in a more elegant manner. They also permit you to perform multiple levels of aggregation within a single query. They can be used only within a top-level SELECT statement, within the top-level SELECT statement within a view definition, or within the top-level statement within an INSERT statement.

  For more information, see "Common Table Expressions" [*ASA SQL User's Guide*, page 305].

- **Recursive union can now be performed using a common table expression of a particular form** Recursive common table expressions allow you to write recursive queries. These are particularly useful when querying tables that represent hierarchical data structures or directed graphs. Each recursive common table expression contains an initial subquery, which is executed first, and a recursive subquery. The A reference to the view, which must appear within the FROM clause of the recursive subquery, references the rows added to the view during the previous iteration. You must be particularly careful to provide conditions that stop the recursion if the data structure you are querying may contain cycles.

  For more information, see "Recursive common table expressions" [*ASA SQL User's Guide*, page 314].

- **INTERSECT and EXCEPT operations are now supported** These operations compute the intersection and difference between two or more result sets. They complement the UNION operation.

  For more information, see the following:

  - "Using EXCEPT and INTERSECT" [*ASA SQL User's Guide*, page 252]
  - "Set operators and NULL" [*ASA SQL User's Guide*, page 254]
  - "EXCEPT operation" [*ASA SQL Reference*, page 423]
  - "INTERSECT operation" [*ASA SQL Reference*, page 482]

- **SELECT statements can operate on stored procedure result sets** In SELECT statements, a stored procedure call can now appear anywhere a base table or view is allowed.

If you want statistics on stored procedure calls to be stored, you must upgrade the database using the Upgrade utility. Without statistics, you may get bad plans if you try to join the result of a stored procedure call.

For more information, see "FROM clause" [*ASA SQL Reference*, page 445].

- **Online analytical processing features added** Several OLAP features have been added to the allowed SQL language:
  - **ROLLUP operation** For queries with a GROUP BY clause, the ROLLUP operation adds subtotal rows into the result set. Each subtotal row provides an aggregate over a set of rows in the GROUP BY result set.

    For more information, see "The ROLLUP operation: adding summary information to GROUP BY queries" [*ASA SQL User's Guide*, page 247]
  - **The LIST function can include ordered lists** The LIST function has been extended to provide sorted lists of items.

    For more information, see "LIST function [Aggregate]" [*ASA SQL Reference*, page 148].
  - **Additional aggregate functions** Functions have been added to compute sample-based and population-based standard deviations and variances.

    For more information, see "Aggregate functions" [*ASA SQL Reference*, page 84].
- **The CREATE INDEX statement permits an index to be created on a built-in function** This feature is a convenience method that adds a new computed column to a table, and creates an index on that column.

  For more information, see "CREATE INDEX statement" [*ASA SQL Reference*, page 319], and "Creating indexes" [*ASA SQL User's Guide*, page 60].
- **ORDER BY clause allowed in all contexts** In previous releases, many SELECT statements in view definitions, in subqueries, or in UNION operations were not allowed to use an ORDER BY clause. This restriction has now been removed.

  In some cases, particularly when combined with the FIRST or TOP clause, using a SELECT with an ORDER BY clause does affect the results of a view definition or a set operation. In other contexts, the ORDER BY clause is allowed but makes no difference to the operation.
- **SELECT statements can now include START AT as part of the TOP clause** START AT provides additional flexibility in queries that explicitly limit the result set.

For more information, see “SELECT statement” [*ASA SQL Reference,* page 541].

- **Constraints can now be named** Check constraints, unique constraints, and referential integrity constraints can now be assigned names. This permits modification of table and column constraints by changing individual constraints, rather than by modifying an entire table constraint.

  To gain the benefits of this enhancement on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

  For more information, see “ALTER TABLE statement” [*ASA SQL Reference,* page 250], “CREATE TABLE statement” [*ASA SQL Reference,* page 361], and “Using table and column constraints” [*ASA SQL User’s Guide,* page 85].

- **Lateral derived tables permit outer references in the FROM clause** Outer references can now be made from derived tables and from stored procedures in the FROM clause. To indicate that an outer reference is being made, the LATERAL keyword is used.

  For more information, see “FROM clause” [*ASA SQL Reference,* page 445].

- **EXECUTE IMMEDIATE allows more flexible escape character processing** A new option WITH ESCAPES OFF allows escape character processing to be suppressed. This feature makes it easier to construct dynamic statements that include file paths.

  For more information, see “EXECUTE IMMEDIATE statement [SP]” [*ASA SQL Reference,* page 429].

- **EXECUTE IMMEDIATE supports queries that return result sets** This new feature allows more dynamic construction of statements inside stored procedures.

  ☞ For more information, see “Using the EXECUTE IMMEDIATE statement in procedures” [*ASA SQL User’s Guide,* page 658], and “EXECUTE IMMEDIATE statement [SP]” [*ASA SQL Reference,* page 429].

- **CREATE FUNCTION and ALTER FUNCTION now permit Transact-SQL syntax** You can now create user-defined functions in the Transact-SQL dialect that return a scalar value to the calling environment.

  ☞ For more information, see “CREATE FUNCTION statement” [*ASA SQL Reference,* page 315].

- **Values of autoincrement columns are now available when inserting multiple rows** When inserting rows through a value-sensitive (keyset

driven) cursor, the newly inserted rows appear at the end of the cursor result set.

A consequence of this change is that the value of an autoincrement column for the most recent row inserted can be found by selecting the last row in the cursor. For example, in embedded SQL the value could be obtained using `FETCH ABSOLUTE -1 cursor-name`.

☞ For more information, see "Modifying rows through a cursor" [*ASA Programming Guide*, page 23].

- **Remote Data Access now handles UUID/GUID columns** Remote Data Access can now manage SQL Server unique identifier columns.

  For more information, see "Data type conversions: Microsoft SQL Server" [*ASA SQL User's Guide*, page 601], and "UNIQUEIDENTIFIERSTR data type [Character]" [*ASA SQL Reference*, page 55].

- **Remote Data Access now names remote connections** Remote Data Access connections made via ODBC are now given names, so that they can be dropped.

  ☞ For more information, see "Managing remote data access connections" [*ASA SQL User's Guide*, page 588].

- **New function returns data type of an expression** The EXPRTYPE function returns the data type of an expression.

  For more information, see "EXPRTYPE function [Miscellaneous]" [*ASA SQL Reference*, page 133].

- **EXIT statement enhanced** The Interactive SQL EXIT statement can now set an exit code for Interactive SQL.

  For more information, see "EXIT statement [Interactive SQL]" [*ASA SQL Reference*, page 432].

- **OUTPUT statement accepts ASIS keyword** When ASIS is specified, values are written to the file without any escaping.

  For more information, see "OUTPUT statement [Interactive SQL]" [*ASA SQL Reference*, page 501].

- **Indexes and foreign keys can be altered** The ALTER INDEX statement allows indexes and foreign keys to be renamed. It also allows an index type to be changed to clustered or nonclustered for user-created indexes as well as primary or foreign key indexes.

  To gain the benefits of clustered indexes on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

For more information, see “ALTER INDEX statement” [*ASA SQL Reference*, page 234].

- **Multiple distinct aggregates permitted in queries** Aggregate functions can take `DISTINCT` *`column-name`* as an argument. In previous versions of the software, only one aggregate function with a DISTINCT argument could be included in a query. Now, multiple such functions can be used. The following query is permitted in version 9, but not in earlier versions of the software:

```
SELECT count( DISTINCT first_name ),
       count( DISTINCT last_name )
FROM contact
```

- **Full length and abbreviated day names are recognized in all supported languages for event schedules** When creating events, the database server recognizes both full-length and abbreviated day names in any of the languages supported by Adaptive Server Anywhere. Previously, schedules in non-English languages required full day names.

  For more information, see “CREATE EVENT statement” [*ASA SQL Reference*, page 304].

- **Hide procedure text to keep your logic confidential** You can obscure the logic contained in stored procedures, functions, triggers and views using the SET HIDDEN option. This allows applications and databases to be distributed without revealing the logic in stored procedures, functions, triggers, and views.

  To gain the benefits of this enhancement on databases created before this release, you must upgrade the database using the Upgrade utility.

  ☞ For more information, see “Hiding the contents of procedures, functions, triggers and views” [*ASA SQL User’s Guide*, page 670].

Administration and scalability enhancements

- **The Validation utility gives more detailed return codes** The Validation utility (dbvalid) gives more specific return codes to indicate the reason a failure occurs.

  ☞ For more information, see “Validating a database using the dbvalid command-line utility” [*ASA Database Administration Guide*, page 548].

- **Two new server properties** Two new server properties have been added. CommandLine gives you the line that was used to start the server, and CompactPlatformVer gives a condensed version of the PlatformVer server property.

  ☞ For more information, see “Server-level properties” [*ASA Database Administration Guide*, page 657].

- **New sp_remote_primary_keys stored procedure** In order to obtain primary key information about remote tables using remote data access, a new stored procedure called sp_remote_primary_keys has been added.

  To gain the benefits of this enhancement on databases created before this release, you must upgrade the database using the Upgrade utility.

  ☞ For more information, see "sp_remote_primary_keys system procedure" [*ASA SQL Reference*, page 749].

- **New connection_property returns the name of the communication link for the connection** The new CommNetworkLink connection property returns the name of the communication link for the connection.

  ☞ For more information, see "Connection-level properties" [*ASA Database Administration Guide*, page 647].

- **NetWare now supports full character set conversion** In 8.x, NetWare supported single-byte-to-single-byte character set conversion, but in 9.0, all character sets supported by the other platforms are also supported on NetWare.

  ☞ For more information, see "-ct server option" [*ASA Database Administration Guide*, page 137].

- **Unload utility can unload column lists** The Unload utility (dbunload) can now unload the column list for the LOAD TABLE statements that it generates in the reload.sql file, facilitating easier reordering of the columns in a table

  ☞ For more information, see "The Unload utility" [*ASA Database Administration Guide*, page 533].

- **Database server registers with LDAP** The database server can now register itself with an LDAP server, so that clients and the Locate Utility [dblocate] can query the LDAP server to find it. This allows clients running over a WAN or through a firewall to find servers without specifying the IP address to find such servers. LDAP is only used with TCP/IP, and only on network servers.

  ☞ For more information, see "Connecting using an LDAP server" [*ASA Database Administration Guide*, page 97] or "LDAP communication parameter [LDAP]" [*ASA Database Administration Guide*, page 209].

- **Improved handling of a large number of connections** The liveness timeout value now increases automatically when there are more than 200 connections in an effort to better handle a large number of connections.

  ☞ For more information, see "-tl server option" [*ASA Database Administration Guide*, page 159] and "LivenessTimeout connection parameter [LTO]" [*ASA Database Administration Guide*, page 194].

- **Request log filtering, host variable support** Output to the request-level log can now be filtered to include only requests from a specific connection or for a specific database. As well, host variable values can now be output to a request log.

  ☞ For more information, see "sa_server_option system procedure" [*ASA SQL Reference,* page 739], "Monitoring and Improving Performance" [*ASA SQL User's Guide,* page 153], "sa_get_request_times system procedure" [*ASA SQL Reference,* page 718], and "-zr server option" [*ASA Database Administration Guide,* page 167].

- **BACKUP statement and DBBACKUP allow renaming of log copy** You can use the BACKUP statement and the Backup utility [dbbackup] to rename the log copy.

  ☞ For more information, see "The Backup utility" [*ASA Database Administration Guide,* page 458] and the "BACKUP statement" [*ASA SQL Reference,* page 263].

- **START DATABASE statement allows log truncation on checkpoint and read-only mode** The START DATABASE statement now allows a database to be started either with log truncation on checkpoint enabled, or in read-only mode.

  ☞ For more information, see "START DATABASE statement" [*ASA SQL Reference,* page 566].

- **Adaptive Server Anywhere supports different auditing options** In previous versions of Adaptive Server Anywhere, you could choose to turn auditing on or off. Now you can specify which options you want to audit.

  ☞ For more information, see "sa_disable_auditing_type" [*ASA SQL Reference,* page 713] or "sa_enable_auditing_type" [*ASA SQL Reference,* page 714].

- **Three new values can be passed to the event_parameter function** Three new values can be passed to the event_parameter function. ScheduleName returns the name of the schedule which fired the event. AppInfo returns the value of the connection_property('AppInfo') for the connection which caused the event. DisconnectReason returns a string indicating why the connection terminated.

  ☞ For more information, see "EVENT_PARAMETER function [System]" [*ASA SQL Reference,* page 128].

- **New server property specifies how many concurrent users are connected to the network server** The new LicensesInUse property determines the numbers of concurrent users currently connected to the network server. Each concurrent user is determined by the number of

unique client network addresses connected to the server, not the number of connections. For example, if three client machines are connected to a server, and each client machine has two connections, select property('LicensesInUse') is '3'.

☞ For more information, see "Server-level properties" [*ASA Database Administration Guide*, page 657].

- **The Service Creation [dbsvc] utility can now start and stop services** Two new options have been added to the Service Creation [dbsvc] utility. **Dbsvc -u** <*service_name*> starts the service named service_name, and *dbsvc -x* <*service_name*> stops the service named service_name.

  ☞ For more information, see "Managing services using the dbsvc command-line utility" [*ASA Database Administration Guide*, page 519].

- **The network server supports the LocalOnly communication parameter [LOCAL]** You can use the LocalOnly communication parameter [LOCAL] with the server. Running a server with the LocalOnly communication parameter set to YES allows the network server to run as a personal server without experiencing connection or CPU limits.

  ☞ For more information, see "LocalOnly communication parameter [LOCAL]" [*ASA Database Administration Guide*, page 209].

- **New minimum database server cache size when using Address Windowing Extensions** The minimum size of the database server cache when using Address Windowing Extensions (AWE) on Windows 2000, Windows XP, and Windows Server 2003 is now 2 Mb. In previous releases, the minimum cache size when using AWE was 3 Gb-256 Mb.

  ☞ For more information, see "-cw server option" [*ASA Database Administration Guide*, page 138].

- **New database property specifies drive type** The new DriveType database property provides information about the drive on which the database file is located.

  ☞ For more information, see "Database properties" [*ASA Database Administration Guide*, page 647].

- **Adaptive Server Anywhere NetWare now faster** The Adaptive Server Anywhere server for NetWare now uses LibC rather than CLIB. LibC is a C runtime library that allows better interaction with the new kernel of the NetWare operating system than the legacy CLIB library. All client-side software for NetWare (including dblib, dbisql, dbconsole, and dbremote) still uses CLIB. This has the benefit of increasing the maximum file size on NetWare to the same as NTFS, allowing multiple CPUs if available,

and allowing TCP and SPX to use Winsock, which is faster than previous versions.

☞ For more information, see "Physical Limitations" [*ASA Database Administration Guide*, page 673] and "Behavior changes in version 9.0" on page 37.

- **External function enhancements on NetWare** External functions or external stored procedures on NetWare can now use multiple NLMs without naming conflicts.

  ☞ For more information, see "External function prototypes" [*ASA SQL User's Guide*, page 666].

- **Connections can specify language of error messages** Each connection to the database server can now request the language in which the database server reports error messages and various other strings. The language used by the connection is independent of the language used by the server. The database server also uses the language requested by the connection to interpret date strings.

- **Two new server properties identify processor type** Two new server-level properties have been added. ProcessorArchitecture identifies the processor type, and on platforms where a processor can be emulated NativeProcessorArchitecture identifies the native processor type.

  For more information, see "Server-level properties" [*ASA Database Administration Guide*, page 657].

Performance enhancements (Query optimization)

The new features listed here are query optimization enhancements that require no user action to use. They take effect without user intervention. If you study query execution plans, you may see the effect of these optimizations.

The optimization enhancements do not require a database upgrade, but they do operate most effectively on a database created using version 9 software.

- **Cost-based subquery optimization** The optimizer has greatly extended the scope of optimizations that are available for subqueries. In previous releases, subqueries were either rewritten as joins during semantic query optimization or were optimized separately from the remainder of a query. Now subqueries that are too complex to be rewritten as joins can still be optimized as an integral part of the query.

- **Buffered row fetching improves performance of sequential scans** When reading rows from a database page for a sequential table scan, Adaptive Server Anywhere can now copy rows into a buffer before returning them to the consumer. Depending on the complexity of the query, this can provide significant time savings.

- **Top N queries executed more efficiently** A new algorithm for executing queries that use the TOP N clause permits faster execution.

  ☞ For more information, see "Sort Top N" [*ASA SQL User's Guide,* page 390].

- **New algorithm for determining which frequencies are kept in histograms** Previously, column histograms created singleton buckets for values with selectivity $> 1\%$. Now, the condition for singleton buckets is relaxed, and instead the histogram tries to keep a minimum number of singleton buckets.

  ☞ For more information, see "Optimizer estimates" [*ASA SQL User's Guide,* page 369].

- **Property QueryCachedPlans shows how many query plans are currently cached** The new property, QueryCachedPlans, shows how many query execution plans are currently cached for a given connection, or across all connections. It can be used in combination with QueryCachePages, QueryOptimized, QueryBypassed, and QueryReused to determine the best setting for the MAX_PLANS_CACHED option.

  ☞ For more information, see "Connection-level properties" [*ASA Database Administration Guide,* page 647] and "Database properties" [*ASA Database Administration Guide,* page 647].

- **Plans are cached faster for procedure statements** The scope of statements for which access plans are cached has been extended to include queries within stored procedures whose result sets are returned by the procedure to the calling environment. This enhancement eliminates the need to re-optimize some statements.

  ☞ For more information, see "Access plan caching" [*ASA SQL User's Guide,* page 375].

- **Index statistics maintained as each index is updated** Statistics are maintained for all indexes, including those on catalog tables, as each index is updated, providing accurate statistics to the optimizer at virtually no performance cost. Statistics persist in SYSATTRIBUTE in the form of one row for each statistic for an index.

  To gain the benefits of this enhancement on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

  ☞ For more information, see "SYSATTRIBUTE system table" [*ASA SQL Reference,* page 618].

Performance enhancements (Server operation)

- **New performance monitor statistics** Two new performance monitor statistics, Comm: Licenses in Use, and Connection Count, have been added to allow users to track the number of connections in use.

  ☞ For more information, see “Communications statistics” [*ASA Database Administration Guide*, page 640] and “Miscellaneous statistics” [*ASA Database Administration Guide*, page 645].

- **The option APPEND { ON | OFF } has been added to the UNLOAD and UNLOAD TABLE statements** A new APPEND option allows unloaded data to be appended to the end of the specified file.

  ☞ For more information, see the “UNLOAD statement” [*ASA SQL Reference*, page 588] or the “UNLOAD TABLE statement” [*ASA SQL Reference*, page 590].

- **Temporary tables can now be declared as NOT TRANSACTIONAL** When NOT TRANSACTIONAL is used, the table is not affected by COMMIT or ROLLBACK. This extension is useful when procedures that access the table are called repeatedly without a COMMIT.

  ☞ For more information, see “CREATE TABLE statement” [*ASA SQL Reference*, page 361], and “DECLARE LOCAL TEMPORARY TABLE statement” [*ASA SQL Reference*, page 397].

- **Persistent index statistics** Maintaining accurate statistics about the physical properties of candidate indexes facilitates the optimizer’s cost based decisions about which indexes to use. Statistics now persist in SYSATTRIBUTE, and are maintained as each index is updated. Additionally, the VALIDATE statement verifies that the statistics on the specified index(es) are accurate and generates an error if they are not. This provides accurate statistics to the optimizer at virtually no performance cost.

  To gain the benefits of this enhancement on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

  ☞ For more information, see “SYSATTRIBUTE system table” [*ASA SQL Reference*, page 618] and the “VALIDATE INDEX statement” [*ASA SQL Reference*, page 602]

- **New OPTIMISTIC_WAIT_FOR_COMMIT option added** This option is meant to mimic 5.x locking behavior when transactions add foreign rows before primary rows. While it is not intended for general use, it can be helpful when migrating 5.x applications to version 8.x.

  ☞ For more information, see “WAIT_FOR_COMMIT option [database]” [*ASA Database Administration Guide*, page 634].

- **New extended property function added** The new **db_extended_property()** is similar to **db_property()** except that it also allows an optional property-specific string parameter to be specified.

  ☞ For more information, see "DB_EXTENDED_PROPERTY function [System]" [*ASA SQL Reference*, page 121].

- **Two new properties added** Two new properties have been added: **FileSize** and **FreePages**. Each of these properties can take an optional argument which specifies the dbspace for which the property is being requested.

  ☞ For more information, see "Database properties" [*ASA Database Administration Guide*, page 647].

- **Server's quiet mode enhanced** The server's quiet mode and error logging switches have been enhanced to allow the server to suppress a variety of messages. Additionally, the -qw option has replaced the -q option, and the -qi option has replaced the -Q option.

Development and administration tools

- **Adaptive Server Anywhere plug-in changes** The Adaptive Server Anywhere plug-in for Sybase Central has been reorganized. Much of the information that was previously available in property sheets, dialog boxes, and folders in the left pane is now available on tabs in the right pane. For example, to view information about a foreign key, you now select the table that has the foreign key in the left pane and then select the Foreign Keys tab in the right pane. In previous versions, there was a separate Foreign Keys folder in the left pane.

  Several other changes have been made to the plug-in, including the following:

  - The Code Editor and Table Editor are no longer separate windows. Now you can edit stored procedures, functions, triggers, and events, as well as edit tables, in the right pane of Sybase Central.
  - The toolbar buttons now change to include options specific to the object selected.
  - The SQL Statements log and server messages (the same information that appears in the Server Messages window) can now be viewed directly in the Sybase Central main window. To view this information, in Sybase Central choose File ➤ Design Details. The Design Details pane appears at the bottom of the main Sybase Central window.
  - The Adaptive Server Anywhere plug-in provides several new wizards to guide you through tasks, including creating tables, unique constraints, and web services.

- **Enhanced clipboard support in the Adaptive Server Anywhere plug-in** Clipboard support has been enhanced in the Adaptive Server Anywhere plug-in so you can copy and paste most objects within Sybase Central into other applications, such as Interactive SQL or a text editor. When you copy objects into other applications, depending on the object you select, either the object name or the SQL for the object appears. For example, if you copy an index in Sybase Central and paste it into a text editor, the CREATE INDEX statement for that index appears.

  ☞ For more information, see "Copying database objects in Sybase Central" [*ASA SQL User's Guide*, page 73].

- **Debugger changes** The database object debugger that lets you debug both stored procedures and Java classes has been integrated into Sybase Central. The user interface has been redesigned.

  ☞ For more information, see "Debugging Logic in the Database" [*ASA SQL User's Guide*, page 673].

- **Sybase Central, Interactive SQL, and the Console utility include an option to automatically check for software updates** Sybase Central, Interactive SQL, and the Console utility can be configured to automatically check for software updates. This option can be set from the Options dialog in Interactive SQL and the Console utility, and can be set from the Help menu in Sybase Central when the Adaptive Server Anywhere plug-in is loaded. In previous releases, you had to go to a web site to obtain this information.

  For more information, see "Options dialog: Check for updates tab" [*SQL Anywhere Studio Help*, page 152].

- **Enhancements made to the Console utility** There have been a number of enhancements to the Console utility, including changes to the interface, support for multiple connections, sorting, and drag and drop.

- **Fast launching of Sybase Central and Interactive SQL** On Windows, Sybase Central and Interactive SQL include a fast launcher that is designed to reduce application startup time when you start Sybase Central or Interactive SQL. Running Adaptive Server Anywhere 9.0 starts two background processes, an instance of *dbisqlg.exe* and an instance of *scjview.exe*, which are the fast launcher processes for Interactive SQL and Sybase Central, respectively. Both of these executables are started when the user logs in.

  ☞ For more information, see "Options dialog: General tab" [*SQL Anywhere Studio Help*, page 144].

- **Syntax highlighting editor in Interactive SQL** You can configure the appearance of syntax typed in the SQL Statements pane of Interactive SQL using the Interactive SQL Options dialog.

  ☞ For more information, see "Options dialog: Editor tab" [*SQL Anywhere Studio Help*, page 149].

- **Printing from Interactive SQL** You can print the contents of the SQL Statements pane and of the graphical plan in Interactive SQL.

  ☞ For more information, see "Interactive SQL main window description" [*ASA Getting Started*, page 70].

- **Graphical plan enhancements** The graphical query access plan display has been enhanced in several ways:
  - The number of rows that passes from one operator to another is indicated by varying line thickness.
  - Slow operations are highlighted by a red border.
  - The statistics display has been extended and reorganized.
  - You can now print the access plan.

- **Database Utilities accept @filename parameters** All of the database administration utilities except Interactive SQL [dbisql], the Debugger utility [dbprdbg], the Language utility [dblang], and the Console utility [dbconsole] now accept parameters contained within a file using the @file syntax. The filename can occur at any point in the command line, and parameters contained in the file are inserted at that point. Multiple files can be specified, and the file specifier can be used with command line switches. Note that the @file syntax is not recursive.

  ☞ For more information, see the "@filename server option" [*ASA Database Administration Guide*, page 131].

- **Row numbers can appear beside results in Interactive SQL** Interactive SQL has an option to display row numbers beside results. This option can be set on the Results tab of the Interactive SQL options dialog.

  ☞ For more information, see "Options dialog: Results tab" [*SQL Anywhere Studio Help*, page 145].

- **Interactive SQL can be set as the default editor for .SQL files** On Windows platforms, you can create a file association for *.SQL* files so that when you double-click the file, Interactive SQL is used to open the file.

  ☞ For more information, see "The Interactive SQL utility" [*ASA Database Administration Guide*, page 492].

- **Interactive SQL Command History dialog enhancements** You can now copy and delete commands from the Command History dialog in Interactive SQL, as well as select multiple commands in the window. The command history now persists between Interactive SQL sessions.

  ☞ For more information, see "Printing SQL statements" [*ASA Getting Started*, page 83].

- **Warning messages now have W prefix** Prior to version 9.0, all warning and error messages had a prefix of I or E. Warning messages now have a prefix of W. This change affects dbmlsrv9, dbmlsync, dbremote, ssremote, dbltm, and ssqueue.

MobiLink new features

Following is a list of changes and additions to the software introduced in version 9.0.0.

- **Server-initiated synchronization** Server-initiated synchronization allows you to initiate MobiLink synchronization from the consolidated database. This means you can push data updates to remote databases. The MobiLink component (the Notifier) provides programmable options for determining what changes in the consolidated database will initiate synchronization and how remotes are chosen to receive update messages. The remote component (the Listener) determines how remotes respond.

  ☞ For more information, see "Server-Initiated Synchronization" [*MobiLink Synchronization User's Guide*, page 137].

- **File-based downloads** Downloads can now be processed as a file that can be distributed in any way that files are distributed, such as e-mail, ftp, disk, or multicast file distribution. For this release, this feature can be used only with Adaptive Server Anywhere remote databases.

  ☞ For more information, see "File-Based Downloads" [*MobiLink Synchronization User's Guide*, page 117].

Enhancements to the MobiLink synchronization server

- **New connection scripts begin_publication and end_publication** Two new scripts have been added. One of their uses is implementing file-based downloads.

  ☞ For more information, see "begin_publication connection event" [*MobiLink Synchronization Reference*, page 118] and "end_publication connection event" [*MobiLink Synchronization Reference*, page 155].

- **New connection script authenticate_parameters** A new script has been added that allows custom authentication. The new script is invoked during authentication, before the begin_synchronization script.

☞ For more information, see "authenticate_parameters connection event" [*MobiLink Synchronization Reference*, page 98].

- **New option removes blank padding of strings** For columns of type VARCHAR or LONG VARCHAR, the dbmlsrv9 -b option removes trailing blanks from strings during synchronization.

  ☞ For more information, see "-b option" [*MobiLink Synchronization Reference*, page 8].

- **Option starts new log file with .old extension** The dbmlsrv9 -on option allows you to set a hard limit on the amount of disk space used by the MobiLink server log.

  ☞ For more information, see "-on option" [*MobiLink Synchronization Reference*, page 14].

- **Log progress offsets** The MobiLink synchronization server can now report progress offsets, last upload time, and last download time. To obtain this information, use the dbmlsrv9 options -vp or -v+.

  ☞ For more information, see "-v option" [*MobiLink Synchronization Reference*, page 21].

- **Handling errors and warnings in .NET and Java synchronization logic** You can now add logic to to deal with errors and warnings at the MobiLink synchronization server.

  ☞ For more information, see "Handling MobiLink server errors in Java" [*MobiLink Synchronization User's Guide*, page 236] and "Handling MobiLink server errors with .NET" [*MobiLink Synchronization User's Guide*, page 257].

- **Additions to MobiLink system tables** Two new columns have been added to both the ml_user and ml_subscription tables. They are last_upload_time and last_download_time. The default is NOT NULL with a default time of January 1, 1900 00:00:00.

  In addition, a subscription_id column has been added to ml_subscription. The publication_name column now contains the publication name.

  ☞ For more information, see "ml_user" [*MobiLink Synchronization Reference*, page 320] and "ml_subscription" [*MobiLink Synchronization Reference*, page 318].

Enhancements for Adaptive Server Anywhere clients

- **Upload-only synchronization** You can now choose to perform an upload-only synchronization.

  ☞ For more information, see the dbmlsync "-uo option" [*MobiLink Synchronization Reference*, page 80].

- **Download-only synchronization** You can now choose to perform a download-only synchronization.

  ☞ For more information, see "-ds option" [*MobiLink Synchronization Reference,* page 43] and "DownloadOnly (ds) extended option" [*MobiLink Synchronization Reference,* page 49].

- **Window messages can initiate synchronization** You can now wake up dbmlsync and perform a synchronization by registering a window message as dbas_synchronize and sending it to the dbmlsync top level window.

- **Load dlls on startup (for Windows CE)** The new dbmlsync option -pd specifies DLLs that should be loaded on startup. This option should be used by everyone using dbmlsync on Windows CE.

  ☞ For more information, see "-pd option" [*MobiLink Synchronization Reference,* page 75].

- **New way to upgrade or revise schema** The hook sp_hook_dbmlsync_schema_upgrade stored procedure has been added to replace the dbmlsync option -i and extended option SiteScriptName (sn).

  ☞ For more information, see "sp_hook_dbmlsync_schema_upgrade" [*MobiLink Synchronization Reference,* page 291].

- **MobiLink return codes** To help you track and log the success and failure of your synchronizations, especially when you have multiple synchronizations in a dbmlsync session, there is a new client event hook procedure, sp_hook_dbmlsync_process_return_code. In addition, a new value, return code, is set in the #hook_dict table for the sp_hook_dbmlsync_abort hook.

  ☞ For more information, see "sp_hook_dbmlsync_process_return_code" [*MobiLink Synchronization Reference,* page 289] and "sp_hook_dbmlsync_abort" [*MobiLink Synchronization Reference,* page 270].

- **Enhancements to scheduling** When scheduling is specified, you can reduce the amount of time spent scanning the log by using the new extended option HoverRescanThreshold (hrt) or the new hook sp_hook_dbmlsync_log_rescan.

  ☞ For more information, see "HoverRescanThreshold (hrt) extended option" [*MobiLink Synchronization Reference,* page 52] and "sp_hook_dbmlsync_log_rescan" [*MobiLink Synchronization Reference,* page 286].

Languages other than English now support the use of abbreviated day names in schedules. Previously, schedules in non-English languages required full day names.

Two new keywords have been added to scheduling syntax: **INFINITE** instructs dbmlsync to wait indefinitely to be signalled for the next synchronization, and **0** as a day of the month specifies the last day of the month.

☞ For more information, see "Schedule (sch) extended option" [*MobiLink Synchronization Reference*, page 59].

Enhancements for UltraLite clients

- **Additional troubleshooting assistance for HotSync conduit** You can configure the HotSync conduit to record troubleshooting information in the HotSync log.

  ☞ For more information, see "HotSync log files" [*MobiLink Synchronization User's Guide*, page 215].

Performance and monitoring enhancements

- **Better dbmlsync performance when there are no schema changes** Dbmlsync no longer loads schema information before every synchronization by default. This typically speeds up synchronization on slower handheld devices by 20 seconds.

  ☞ For more information, see "-sc option" [*MobiLink Synchronization Reference*, page 79].

- **Better dbmlsync performance on Windows CE** Dbmlsync no longer uses dbtool9.dll on Windows CE. This means that it uses less memory.

- **MobiLink Monitor command line options** The MobiLink Monitor can now be started from the command line with a variety of options.

  ☞ For more information, see "Starting the MobiLink Monitor" [*MobiLink Synchronization User's Guide*, page 299].

- **Enhancements to Redirector** A new parameter, LOG_LEVEL, has been added to allow you to control the verbosity level.

  ☞ For more information, see "Configuring Redirector properties (all versions)" [*MobiLink Synchronization User's Guide*, page 318].

- **Improved liveness** When connecting over TCP/IP, dropped connections are detected more quickly. This frees up MobiLink worker threads more quickly when a connection is dropped, improving throughput.

Miscellaneous

- **Warning messages now have W prefix** Prior to version 9.0, all warning and error messages had a prefix of I or E. Warning messages now have a prefix of W. This change affects dbmlsrv9, dbmlsync, dbremote, ssremote, dbltm, and ssqueue.

SQL Remote new features

SQL Remote version 9.0 includes the following new features.

- **Warning messages now have W prefix** Prior to version 9.0, all warning and error messages had a prefix of I or E. Warning messages now have a prefix of W. This change affects dbmlsrv9, dbmlsync, dbremote, ssremote, dbltm, and ssqueue.

UltraLite new features

UltraLite development is possible using two kinds of programming interface:

- **UltraLite components** UltraLite components bring UltraLite database and synchronization features to users of rapid application development tools. They provide a familiar interface for each supported development tool. UltraLite components provide a simple table-based data access interface and also dynamic SQL for more complex queries.

  The UltraLite components were introduced in version 8.0.2.

  ☞ For an overview of the components, see "Introduction" [*UltraLite Database User's Guide*, page 4].

- **Static development models** Embedded SQL, the static C++ API, and the static Java API are still available. These are now referred to in the documentation as static interfaces to distinguish them from the components.

In particular, note the following:

- **Native UltraLite for Java** is an UltraLite component, which uses a C/C++ UltraLite runtime. The UltraLite **static Java API** is a pure Java solution, available in previous releases, in which queries must be specified at compile time.
- **UltraLite for C++** is a component interface. The UltraLite **static C++ API** is a static interface, available in previous releases, in which queries must be specified at compile time.
- **Embedded SQL** is a static interface, in which queries must be specified at compile time.

Following is a list of changes and additions to the software introduced in version 9.0.

- **New components** In addition to the components for AppForge MobileVB, eMbedded Visual Basic, and Java, the following components have been introduced:

- **UltraLite .NET** A component for development using the Visual Studio .NET environment. Applications built with this component can be deployed to devices that support the .NET Compact Framework (version 1.05.0000 or later).

  ☞ For more information, see "Introduction to UltraLite.NET" [*UltraLite.NET User's Guide*, page 1].

- **C++ component** A component for development using C++ compilers.

  ☞ For more information, see "Introduction to UltraLite for C++" [*UltraLite C++ User's Guide*, page 1].

♦ **Pocket IE support** The eMbedded Visual Basic component has been upgraded to an ActiveX component. Support has been added for development using JScript, for applications that run from Pocket IE on Windows CE devices.

☞ For more information, see "Introduction" [*UltraLite ActiveX User's Guide*, page 1].

♦ **Dynamic SQL** In addition to the table-based data access interface provided in version 8.0.2, the UltraLite components can now use Dynamic SQL for more complex queries, including multi-table joins.

☞ For more information, see "Dynamic SQL" [*UltraLite Database User's Guide*, page 125].

♦ **Connection parameters** Connection parameters for the UltraLite components (except C++) are now exposed as individual properties rather than as a single string. This design makes debugging connection issues easier and makes connection management more straightforward.

☞ For more information, see "Connection Parameters" [*UltraLite Database User's Guide*, page 49].

♦ **Drag and drop MobileVB component** The MobileVB component can now be dragged on to a form. The properties of the component can be set in the design environment as well as in code.

☞ For more information, see "UltraLite for MobileVB architecture" [*UltraLite for MobileVB User's Guide*, page 4].

♦ **Multi-process access** The C++ component supports access from more than one process. To develop an application using this model, a separate UltraLite database engine and the application must be linked against a different UltraLite runtime library.

☞ For more information, see "Compiling and linking your application" [*UltraLite C++ User's Guide*, page 38].

- **Concurrent synchronization** In previous releases, all access to data was prevented during synchronization. Full access to the data is now provided during the download phase of synchronization. Read-only access is provided during the upload phase.

  ☞ For more information, see "Threading in UltraLite applications" [*UltraLite Database User's Guide*, page 47].

- **Palm OS enhancements** On the Palm OS the structure of the UltraLite code has been reorganized to make better use of Palm database segments.

- **Extended error information** More error information is available to applications built using the UltraLite components.

- **Unicode library available on Windows NT/2000/XP** A Unicode version of the UltraLite runtime library is provided for embedded SQL and static C++ API applications. This version is used by the UltraLite components. When using this library, UltraLite database files are compatible between Windows CE and desktop operating systems.

  ☞ For more information, see "Character sets in UltraLite" [*UltraLite Database User's Guide*, page 40].

- **Windows XP supported as a deployment platform** UltraLite application deployment is now supported on Windows XP.

Behavior changes in version 9.0

This section lists the behavior changes introduced in components of SQL Anywhere Studio version 9.0.

Adaptive Server Anywhere behavior changes

The following is a list of behavior changes from previous versions of the software.

- **Java objects in the database not supported** Support has been removed for storing data as Java objects. Support is maintained for Java stored procedures.

  ☞ For a description of the current support for Java in the database, see "Introduction to Java in the Database" [*ASA Programming Guide*, page 51].

- **New Greek collation for Windows environment** Greek collations for OEM/DOS character sets existed in previous versions, however, a new Greek collation, 1253ELL, has been added for Windows. When creating a new database in a Greek Windows environment, 1253ELL will be selected automatically if a collation is not specified.

  ☞ For more information, see "Supplied and recommended collations" [*ASA Database Administration Guide*, page 308].

- **New connection limit** The database server now allows one extra DBA connection above the connection limit, to allow a DBA to connect and drop other connections in case of an intentional or accidental denial-of-service.

  ☞ For more information, see "-gm server option" [*ASA Database Administration Guide*, page 147].

- **References to table expressions preceding in the FROM clause may now be used in ON clauses of nested outer joins.** In previous releases, outer references in the ON phrase were permitted. Such outer references must now be indicated by use of the LATERAL keyword. The restriction enforces clarity and conforms to the SQL/99 standard.

  The following query is an example of one that is no longer valid, as it contains an outer reference (highlighted) without use of the LATERAL keyword:

```
SELECT *
FROM T1,
    T2 LEFT OUTER JOIN
                ( T3 LEFT OUTER JOIN T4 ON T1.col1 =
        T2.col2 )
    ON T1.col2 = T2.col2
```

For more information, see “FROM clause” [*ASA SQL Reference*, page 445].

- **Unqualified table references with multiple matches are reported as syntax errors** In previous releases, if a query contained a reference to a table without an owner name specified (an unqualified table reference) and if more than one match was possible on that table, the first match found was used. Unqualified table references now cause an error. See “Table name ‘%1’ is ambiguous” [*ASA Error Messages*, page 326].

- **LIKE operator with NULL escape character now evaluates to NULL** LIKE predicates containing a NULL escape character now evaluate to NULL. Previously, a LIKE predicate with a NULL escape character was evaluated as if there were no escape character. The new behavior matches the ISO/ANSI specification.

- **Properties and statistics removed** The ServerIdleWaits database property, and the TaskSwitch and CurrTaskSwitch connection properties have been removed, along with their corresponding performance monitor statistics: Context Switches, Server Idle Waits/sec, Request Queue Waits/sec.

- **Column statistics are updated on INSERT/UPDATE/DELETE** Statistics are now updated when executing an INSERT, UPDATE, or DELETE statement results in changing a significant amount of data.

- **Statistics no longer updated during recovery** The server no longer updates statistics during recovery or when executing simple DELETE and UPDATE statements. Simple statements are those that are not optimized and executed directly by the server.

- **Histogram ranges displayed as the correct data type** The sa_get_histogram() system procedure and the histogram [dbhist] utility previously displayed outputted ranges in hash values. Now, outputted histogram ranges match the data in the corresponding column, and are displayed as the correct data type.

  ☞ For more information, see “The Histogram utility” [*ASA Database Administration Guide*, page 481] and the “sa_get_histogram system procedure” [*ASA SQL Reference*, page 717].

- **Only one consolidated user permitted per remote database** It is no longer possible to define multiple consolidated users on the same remote database.

  ☞ For more information, see “GRANT CONSOLIDATE statement [SQL Remote]” [*ASA SQL Reference*, page 460] or “REVOKE CONSOLIDATE statement [SQL Remote]” [*ASA SQL Reference*, page 532].

- **CommLinks connection parameter uses shared memory if not explicitly specified** Now, connections that do not specify a CommLinks connection parameter always attempt to connect over shared memory.

  ☞ For more information, see "CommLinks connection parameter [LINKS]" [*ASA Database Administration Guide,* page 179].

- **CommLinks connection parameter always attempts shared memory protocol first** When you specify CommLinks=all, Adaptive Server Anywhere always attempts to connect using the shared memory protocol before attempting to connect using other protocols.

  ☞ For more information, see "CommLinks connection parameter [LINKS]" [*ASA Database Administration Guide,* page 179].

- **Connection errors abort process** Previously, connection protocols listed in the CommLinks connection parameter were attempted one by one until a connection occurred. Now, if a connection error occurs during the process, it aborts the connection process immediately, regardless of whether or not all the listed protocols were tried.

  ☞ For more information, see "CommLinks connection parameter [LINKS]" [*ASA Database Administration Guide,* page 179].

- **Default value for PREVENT_ARTICLE_PKEY_UPDATE changed** The default value for the PREVENT_ARTICLE_PKEY_UPDATE database option has been changed to ON to reflect the fact that updating primary key values should be avoided. The new default setting disallows primary key updates on primary keys that are part of a publication. You can override this feature by setting the value to OFF.

  ☞ For more information, see "PREVENT_ARTICLE_PKEY_UPDATE option [database]" [*ASA Database Administration Guide,* page 619].

- **Some functions treated as non-deterministic** The RAND, NEWID, and GET_IDENTITY functions are treated as non-deterministic. A consequence is that these functions are not cached during query execution.

  ☞ For more information, see "Function caching" [*ASA SQL User's Guide,* page 418].

- **Performance messages now display database name** The engine performance advice messages now display the database name. This is especially helpful when running more than one database. As well, messages starting with the word *Note* indicate that they are advice messages.

- **NetWare clients using Adaptive Server Anywhere versions prior to 9.0 require upgrade** As a result of enhancements to NetWare support in Adaptive Server Anywhere, NetWare clients using Adaptive Server Anywhere versions prior to 9.0 cannot connect to 9.0 servers using shared memory unless they have a specific EBF installed. The build numbers are: 7.0.4.3400, 8.0.0.2358, 8.0.1.3088, and 8.0.2.4095. Clients with build numbers before these will simply not find the 9.x server.

- **Change in syntax for ALTER DATABASE CALIBRATE** The syntax for ALTER DATABASE CALIBRATE TEMPORARY DBSPACE has been changed to ALTER DATABASE CALIBRATE DBSPACE TEMPORARY to make the syntax consistent with other, similar statements.

  ☞ For more information, see the "ALTER DATABASE statement" [*ASA SQL Reference*, page 225].

- **Dynamic cache sizing more aggressive** Dynamic cache sizing is now more aggressive at resizing the cache after a new database is started or when a file grows significantly. Prior to this change, statistics were sampled and the cache was resized at most once per minute. Now, after a database is started or a file grows significantly, statistics are sampled and the cache may be resized every five seconds for thirty seconds.

  ☞ For more information, see "Using the cache to improve performance" [*ASA SQL User's Guide*, page 176].

- **Determining the language for interfaces and messages** Two new environment variables, ASLANG and ASCHARSET, control languages used in interfaces (such as Sybase Central or Interactive SQL) and messages. ASLANG specifies the language, and ASCHARSET specifies the character set.

  For more information, see the "ASLANG environment variable" [*ASA Database Administration Guide*, page 245] or the "ASCHARSET environment variable" [*ASA Database Administration Guide*, page 245].

- **Rowcount setting now limits the rows returned** The rowcount setting now limits the rows returned by a cursor from the top. It is no longer possible to position to the beginning of the results using an absolute fetch.

  You can use the new feature, TOP N / START AT to emulate this behaviour if it is needed.

  For more information see "Sort Top N" [*ASA SQL User's Guide*, page 390].

Deprecated and unsupported features

Adaptive Server Anywhere -d server option no longer supported As a result of enhancements to NetWare support in Adaptive Server Anywhere, the `-d` server option is no longer supported.

NetWare 4.x no longer supported As a result of enhancements to NetWare support in Adaptive Server Anywhere, Adaptive Server Anywhere will only run on NetWare version 5.1 SP6 or higher, or version 6.0 SP3 or higher. The correct service packs must be installed or the Adaptive Server Anywhere server will display an error message.

SQLLOCALE environment variable no longer supported SQLLOCALE environment variable has been replaced by two new environment variables, ASLANG and ASCHARSET.

For more information, see "Behavior changes in version 9.0" on page 37.

MobiLink behavior changes

The following is a list of behavior changes from previous versions of the software.

- **dbmlsync option -i and extended option SiteScriptName are no longer supported** dbmlsync -i and dbmlsync -e sc are no longer supported. They are replaced with a new hook called sp_hook_dbmlsync_schema_upgrade.

  ☞ For more information, see "sp_hook_dbmlsync_schema_upgrade" [*MobiLink Synchronization Reference*, page 291].

- **Download acknowledgement is now OFF by default** For Adaptive Server Anywhere remotes, the SendDownloadAck extended option now defaults to OFF. For UltraLite remotes, the send_download_ack field of the ul_synch_info struct now defaults to ul_false.

  When you upgrade to version 9, you must explicitly set this option ON if the application depends on knowing that the remote has applied a download before the commit of the download transaction.

  ☞ For more information, see "SendDownloadACK (sa) extended option" [*MobiLink Synchronization Reference*, page 62] and "Send Download Acknowledgement synchronization parameter" [*UltraLite Database User's Guide*, page 172].

- **Some dbmlsync hooks may not work by default on Windows CE devices** The dbmlsync extended option LockTables has been modified to allow you to specify whether tables are locked in shared mode or exclusive mode. The default setting for LockTables, ON, continues to lock tables in shared mode for all platforms other than Windows CE. However, on Windows CE devices, ON now means that tables are locked in exclusive mode. This change provides significant performance enhancements for Windows CE applications.

The dbmlsync event hooks sp_hook_dbmlsync_download_com_error, sp_hook_dbmlsync_download_fatal_sql_error, and sp_hook_dbmlsync_download_log_ri_violation are all executed on separate connections. They will not be able to execute correctly if they attempt to access any synchronization tables that are locked in exclusive mode. If your deployment uses any of these hooks on Windows CE, you may need to set LockTables to SHARE.

☞ For more information, see "LockTables (lt) extended option" [*MobiLink Synchronization Reference,* page 55], "sp_hook_dbmlsync_download_com_error" [*MobiLink Synchronization Reference,* page 275], "sp_hook_dbmlsync_download_fatal_sql_error" [*MobiLink Synchronization Reference,* page 277], and "sp_hook_dbmlsync_download_log_ri_violation" [*MobiLink Synchronization Reference,* page 279].

- **MobiLink synchronization server error codes** The MobiLink synchronization server now provides more information about errors. All MobiLink server error codes are less than -10000, starting at -10001. For dbmlsync, the error appears in the GUI and the output file. For UltraLite, the error is available as a string in the ul_synch_info struct.

  ☞ For more information, see "MobiLink Synchronization Server Error Messages" [*MobiLink Synchronization Reference,* page 399].

- **Upload cursors deprecated** The following scripts are deprecated: upload_cursor, new_row_cursor, and old_row_cursor. You should use statement-based scripts for the upload stream.

  ☞ For more information, see "Writing scripts to upload rows" [*MobiLink Synchronization User's Guide,* page 54].

- **-zac and -zec deprecated** The MobiLink synchronization server options for generating cursor-based scripts, -zac and -zec, have been deprecated. You can continue to use -za and -ze to automatically generate statement-based scripts.

  ☞ For more information, see "-za option" [*MobiLink Synchronization Reference,* page 28] and "-ze option" [*MobiLink Synchronization Reference,* page 29].

- **-zd removed** The MobiLink synchronization server option -zd, which caused the last_download timestamp to be passed last, has been removed. This parameter is now always passed first.

- **mlxtract deprecated** The mlxtract utility is deprecated.

  ☞ For information, see "Creating a remote database" [*MobiLink Synchronization User's Guide,* page 168].

- **end_synchronization scripts always called** Prior to version 9.0, the end_synchronization script might not be called if synchronization failed. Now, the script is always called if a begin_synchronization script is called. This means that any cleanup activities you have placed in the end_synchronization script will be performed regardless of whether the synchronization was successful.

  In addition, end_synchronization scripts have a new parameter, sync_ok, that indicates whether the synchronization was successful (1), or failed (0).

  ☞ For more information, see "end_synchronization connection event" [*MobiLink Synchronization Reference,* page 158]and "end_synchronization table event" [*MobiLink Synchronization Reference,* page 160].

- **Stream dlls and shared objects renamed** The names of stream dlls and shared objects have been changed to improve consistency with Adaptive Server Anywhere. The following table details the changes:

Old name	New name
dbhttp9	dbmlhttp9
dbhttps9	dbmlhttps9
dbjrsa9	dbmljrsa9
dbjtls9	dbmljtls9
dbrsa9	dbmlrsa9
dbsock9	dbmlsock9
dbtls9	dbmltls9

  For more information, see "Deploying MobiLink Applications" [*MobiLink Synchronization Reference,* page 337].

- **ScoutSync no longer supported** ScoutSync is no longer supported.

- **Schema information no longer reloaded at each synchronization** Prior to version 9.0, dbmlsync reloaded schema information from the database before each synchronization. It now reloads schema information only at dbmlsync startup. You can revert to the old behavior using the dbmlsync -sc option. If you do not use -sc, dbmlsync should be shut down before any schema changes are made to remote databases. Making schema changes without shutting down dbmlsync could lead to synchronization errors or other unexpected behavior.

  ☞ For more information, see "-sc option" [*MobiLink Synchronization Reference,* page 79].

- **Synchronization now aborts if key scripts are missing** Prior to version 9.0, synchronization would continue even if certain scripts were missing that might result in the loss of data. MobiLink now aborts in this instance. You can use the dbmlsrv9 -fr option to cause an error to be generated instead of failure.

  ☞ For more information, see "-fr option" [*MobiLink Synchronization Reference*, page 13].

- **keep_alive synchronization parameter is always ON** The keep_alive synchronization parameter for TCP/IP and HTTP protocols is now ignored and is always set to ON. This was previously the default setting. To control liveness for TCP/IP connections, you can use the liveness_timeout parameter.

  ☞ For more information, see the liveness_timeout parameter in "CREATE SYNCHRONIZATION USER statement [MobiLink]" [*ASA SQL Reference*, page 351] or "-x option" [*MobiLink Synchronization Reference*, page 24].

UltraLite behavior changes

The following is a list of behavior changes from previous versions of the software.

- **Supported platform changes** The following changes have been made to the supported UltraLite deployment platforms.
  - **ScoutSync no longer supported** Support has been dropped for ScoutSync synchronization software.
  - **VxWorks no longer supported** The VxWorks operating system is no longer supported.
  - **JDK 1.1.8 required for pure Java UltraLite** The pure Java static development model requires JDK 1.1.8 or later, rather than JDK 1.1.4 or later.
  - **Palm OS changes** Changes to the UltraLite architecture for Palm OS provides better performance on newer devices. A consequence is that UltraLite requires more dynamic memory than in previous releases. For anything other than very small databases, it is recommended that Palm OS version 3.5 or later be used, with 4 MB or more of memory.
  - **MobileBuilder and PRC Tools no longer supported** UltraLite development is no longer supported on the PenRight! MobileBuilder platform. Development using the GNU PRC Tool chain is also no longer supported.

- **Development platform changes** Application development for UltraLite components is now supported on Windows NT/2000/XP only. Development using the static interfaces is also supported on Windows 98 SE. Other members of the Windows 95/98/Me family are not supported for development purposes.

  The supported Metrowerks CodeWarrior versions are now 8 and 9.

- **Documentation terminology change** The introduction of the UltraLite components requires new names in order to distinguish the different interfaces. The older UltraLite interfaces (Embedded SQL, the C++ API, and the Java API) are now named **static interfaces**, as the queries they use must be specified at compile time. The components provide access to **dynamic SQL**.

- **UltraLite runtime library on Windows NT/2000/XP** The ActiveX and MobileVB components now use a Unicode runtime library on Windows. This runtime library is compatible with version 8.0.2 UltraLite database (*.udb*) files for Windows, but not with version 8.0.2 UltraLite database files built on other Windows operating systems.

- **file_name parameter** In previous versions of the software, the file_name parameter used to specify the UltraLite database file name on the desktop would also be used to specify the file name on a device if no platform-specific parameter was supplied. The file_name parameter is now ignored except for on desktop operating systems.

  ☞ For more information, see "Database identification parameters" [*UltraLite Database User's Guide,* page 54].

- **Static Java API changes** The static Java API has changed. The following methods that were on the JdbcDatabase object have been moved to the JdbcConnection object:
  - countUploadRows
  - getLastDownloadTimeDate
  - getLastDownloadTimeLong

  The grant and revoke methods have been added to JdbcConnection for use by applications that do not have an explicit JdbcManager object.

  ☞ For more information, see "Class JdbcConnection" [*UltraLite Static Java User's Guide,* page 59].

- **Error code changes** Some UltraLite error codes have changed to more specific and useful values. If you test for individual error codes in your application, check the new codes after upgrade.

  For example, if you check for SQLE_DATABASE_NOT_FOUND (or the equivalent in one of the UltraLite interfaces) when connecting to a

database, you should change this to SQLE_ULTRALITE_DATABASE_NOT_FOUND.

For a list of error codes, see the SQL error object in the interface you are using.

- **UL_STORE_PARMS change for embedded SQL** The UL_STORE_PARMS macro is now evaluated during the EXEC SQL CONNECT statement. The database is no longer started during the dbinit call, but rather on connect. This means that UL_STORE_PARMS could be evaluated a different number of times if you use multiple connections. It also means that UL_STORE_PARMS must be defined before any EXEC SQL CONNECT statements.

CHAPTER 3

What's New in Version 8.0.2

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in SQL Anywhere Studio version 8.0.2.

Contents

New features in version 8.0.2

This section lists the new features introduced in components of SQL Anywhere Studio version 8.0.2.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 8.0.2. It provides an exhaustive listing of major and minor new features, with cross references to locations where each feature is discussed in detail.

Highlighted new features

- **Clustered index support** Creating a clustered index on a table causes the rows in that table to be stored in approximately the same order as they appear in the index. You can use the LOAD TABLE statement to load a table with information in the clustered order. As you insert information into the table, the clustering characteristics of the table degrade. You can use the REORGANIZE TABLE statement to reestablish the clustering order. Clustered indexes can improve performance.

  To use clustered indexes on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

  ☞ For more information, see "Using clustered indexes" [*ASA SQL User's Guide,* page 59].

- **Unique identifier support** Adaptive Server Anywhere supports unique identifiers (UUIDs and GUIDs). UUIDs (universally unique identifiers) and GUIDs (globally unique identifiers) are a mechanism for uniquely identifying rows, even across distinct databases in a synchronization environment.

  ☞ For more information, see "The NEWID default" [*ASA SQL User's Guide,* page 82].

- **Update existing rows with ON EXISTING clause** You can use the ON EXISTING clause of the INSERT statement to update existing rows with new values, as long as the table has a primary key.

  ☞ For more information, see "Changing data using INSERT" [*ASA SQL User's Guide,* page 364], or the "INSERT statement" [*ASA SQL Reference,* page 476].

- **BACKUP statement supported on Windows CE** Adaptive Server Anywhere allows you to create image backups of databases operating on the Windows CE platform, or to rename or truncate the database's transaction log.

☞ For more information, see "Types of backup" [*ASA Database Administration Guide*, page 346], or the "BACKUP statement" [*ASA SQL Reference*, page 263].

- **Graphical plan enhancements** The graphical plan has been enhanced to include more information, resulting in a new look.

  ☞ For more information, see "Graphical plans" [*ASA SQL User's Guide*, page 429].

- **Use of work tables is now explicit** The use of work tables is now postponed until as late as possible in the plan. When work tables are used, they now appear explicitly in the graphical plan.

  ☞ For more information, see "Graphical plans" [*ASA SQL User's Guide*, page 429] or "Use of work tables in query processing" [*ASA SQL User's Guide*, page 185].

- **New joins added** New joins added to this release include the nested loops semijoin, the nested loops anti-semijoin, the hash semijoin and the hash anti-semijoin.

  ☞ For more information, see "Join algorithms" [*ASA SQL User's Guide*, page 381].

Function enhancements

- **Obtain plan for SQL queries of a specific cursor-type** You can now obtain plans for SQL queries based on their cursor type, using the PLAN, EXPLANATION, GRAPHICAL_PLAN functions.

  ☞ For more information, see "GRAPHICAL_PLAN function [Miscellaneous]" [*ASA SQL Reference*, page 135], "EXPLANATION function [Miscellaneous]" [*ASA SQL Reference*, page 132], or "PLAN function [Miscellaneous]" [*ASA SQL Reference*, page 170].

  ☞ For information about setting these plan options in Interactive SQL, see "Options dialog: Plan tab" [*SQL Anywhere Studio Help*, page 148].

- **Character set conversion function** A new function CSCONVERT is available to convert strings between character sets.

  ☞ For more information, see "CSCONVERT function [STRING]" [*ASA SQL Reference*, page 111].

- **Variable test function** A new function VAREXISTS is available to test whether a user-defined variable has been created or declared with a given name. After this test, the variable can be created if necessary, and then used safely.

  ☞ For more information, see "VAREXISTS function [Miscellaneous]" [*ASA SQL Reference*, page 203].

Statement enhancements

- **Hide procedure text to keep your logic confidential** You can obscure the logic contained in stored procedures, functions, triggers and views using the SET HIDDEN option. This is allows applications and databases to be distributed without revealing the logic in stored procedures, functions, triggers, and views.

  ☞ For more information, see "Hiding the contents of procedures, functions, triggers and views" [*ASA SQL User's Guide*, page 670].

- **LOAD TABLE now accepts delimiters of more than 1 byte** The LOAD TABLE statement now supports delimiters that are up to 255 bytes.

  ☞ For more information, see the "LOAD TABLE statement" [*ASA SQL Reference*, page 486].

- **New statement provides compatibility for Adaptive Server Enterprise and Microsoft SQL Server** You can use the DEALLOCATE statement to release resources associated with a cursor. This statement is provided for Adaptive Server Enterprise and Microsoft SQL Server compatibility.

  ☞ For more information, see the "DEALLOCATE statement" [*ASA SQL Reference*, page 386].

- **ALTER DATABASE statement behaves like dblog utility** You can use the ALTER DATABASE statement to change the transaction log and mirror log names associated with a database file. Previously, you could only do this using the Transaction Log (dblog) utility.

  ☞ For more information, see the "ALTER DATABASE statement" [*ASA SQL Reference*, page 225].

- **LOAD TABLE can be used for both global and local temporary tables** Adaptive Server Anywhere now supports the LOAD TABLE statement on declared local temporary tables. Previously, only global temporary tables were supported.

  ☞ For more information, see the "LOAD TABLE statement" [*ASA SQL Reference*, page 486].

- **SET statement can be used to assign variable values** You can now assign values to variables using the SET statement in Transact-SQL procedures.

- **INSERT statement now supports WITH AUTO NAME** If you specify WITH AUTO NAME in an INSERT statement, the names of the items in the SELECT list determine the associations of values to destination columns.

  ☞ For more information, see "INSERT statement" [*ASA SQL Reference*, page 476].

- **EXIT statement enhanced** The Interactive SQL EXIT statement can now set an exit code for Interactive SQL.

  ☞ For more information, see "EXIT statement [Interactive SQL]" [*ASA SQL Reference,* page 432].

- **Specify the optimization goal for a query in the FROM clause** You can use the FASTFIRSTROW table hint to set the optimization goal for the query without setting the OPTIMIZATION_GOAL option to first-row.

  ☞ For more information, see "FROM clause" [*ASA SQL Reference,* page 445].

Security Enhancements

- **New utility allows you to hide the contents of files** Configuration files, also known as command files, sometimes contain passwords. As an enhanced security feature, Adaptive Server Anywhere has a new utility, called the File Hiding utility, that allows you to hide the contents of configuration files using simple encryption.

  ☞ For more information, see "The File Hiding utility" [*ASA Database Administration Guide,* page 466]

- **Certicom encryption changes** Security has been enhanced to support two types of Certicom encryption, ECC_TLS and RSA_TLS. The encryption known in previous versions of Adaptive Server Anywhere as Certicom encryption has been renamed to ECC_TLS encryption. The Certicom parameter is still accepted and is equivalent to ECC_TLS encryption. Adaptive Server Anywhere now also supports RSA_TLS encryption.

  ☞ For more information, see the "-ec server option" [*ASA Database Administration Guide,* page 141] or the "Encryption connection parameter [ENC]" [*ASA Database Administration Guide,* page 188].

Performance Enhancements

- **New communication parameters can improve network responsiveness** The LazyClose and PrefetchOnOpen network communication parameters can improve performance on networks with poor latency or with applications that process many requests.

  ☞ For information about these parameters, see the "LazyClose connection parameter [LCLOSE]" [*ASA Database Administration Guide,* page 193] and the "PreFetchOnOpen communication parameter" [*ASA Database Administration Guide,* page 214]

- **Scattered reads now used on Windows NT/2000/XP** Previously, sequential scans of large tables copied pages to a 64K buffer and then into the cache. Now, providing you are running in a Windows NT Service Patch 2 or higher environment, or in a Windows 2000/XP environment,

and provided your page size is at least 4K, scattered reads copy the pages directly to the cache, thus saving time and improving performance.

☞ For more information, see "Use an appropriate page size" [*ASA SQL User's Guide*, page 170].

- **Improved time resolution in request-level logging** The times obtained using procedure profiling or request-level logging now have a resolution of 1 millisecond. This change primarily affects servers running on Windows operating systems.

- **Running multiple versions of the Performance Monitor** If you run multiple versions of Adaptive Server Anywhere simultaneously, you can also run multiple versions of the Windows Performance Monitor simultaneously.

  ☞ For more information about the Windows Performance Monitor, see "Monitoring database statistics from Windows Performance Monitor" [*ASA SQL User's Guide*, page 190].

Miscellaneous Enhancements

- **Changing server's temp folder via a registry setting** On Windows CE platforms, you can use the registry to specify which temporary directory the server uses.

  ☞ For more information, see "Registry settings on Windows CE" [*ASA Database Administration Guide*, page 250].

- **New iAnywhere JDBC driver** This robust and high-performance JDBC driver enjoys the benefits of ODBC data sources and the Command Sequence client/server protocol. It is an alternative to the jConnect JDBC driver.

  ☞ For information on the iAnywhere JDBC driver, see "Using the iAnywhere JDBC driver" [*ASA Programming Guide*, page 115].

  ☞ For information on choosing a JDBC driver, see "Choosing a JDBC driver" [*ASA Programming Guide*, page 104].

- **Triggers can discriminate among the actions that caused a trigger to fire** You can now carry out different actions depending on whether the trigger was fired by an UPDATE, INSERT, or DELETE operation. This feature enables you to share logic among the different events within a single trigger, and yet carry out some actions in an action-dependent manner.

  ☞ For more information, see "Trigger operation conditions" [*ASA SQL Reference*, page 28].

- **RETURN_DATE_TIME_AS_STRING is no longer TDS specific** All connections can now use the RETURN_DATE_TIME_AS_STRING option.

  ☞ For more information about this option, see "RETURN_DATE_TIME_AS_STRING option [database]" [*ASA Database Administration Guide,* page 622].

- **Units can be specified when adding space to a dbspace** You can extend database files by a specific size, in units of pages, kilobytes, megabytes, gigabytes, or terabytes.

  ☞ For more information, see the "ALTER DBSPACE statement" [*ASA SQL Reference,* page 229]

- **sa_make_object system procedure** This system procedure can be used in a SQL script to ensure that a skeletal instance of an object exists before executing an ALTER statement which provides the actual definition.

  ☞ For more information, see "sa_make_object system procedure" [*ASA SQL Reference,* page 723].

- **New global variable compatible with Microsoft SQL Server** A new global variable has been introduced to allow for Microsoft SQL Server compatibility. The @@fetch_status global variable is the same as the @@sqlstatus global variable, except that it returns the status of the most recent fetch in different values.

  ☞ For more information, see "Global variables" [*ASA SQL Reference,* page 39].

- **Character set conversion supported on NetWare** NetWare now supports character set translation.

  ☞ For more information, see "-ct server option" [*ASA Database Administration Guide,* page 137].

- **Information utility reports the version of installed Java classes** The dbinfo utility and a_db_info structure now report the version of the Java classes installed in a database.

  ☞ For more information, see "The Information utility" [*ASA Database Administration Guide,* page 483] and "a_db_info structure" [*ASA Programming Guide,* page 288].

- **Suppress warnings on fetch operations** Versions 8.0 and later of the database server return a wider range of fetch warnings than earlier versions of the software. The ODBC Configuration for Adaptive Server Anywhere dialog allows you to suppress warning messages returned from

the database server to ensure that they are handled properly for applications that are deployed with earlier versions of the software.

☞ For more information, see "ODBC tab" [*ASA Database Administration Guide*, page 56].

- **Controlling updates to primary key columns** Setting the new prevent_article_pkey_update option to ON disallows updates to the primary key columns of tables that are part of a publication. This option helps ensure data integrity, especially in a replication and synchronization environment.

  ☞ For more information, see the "PREVENT_ARTICLE_PKEY_UPDATE option [database]" [*ASA Database Administration Guide*, page 619].

MobiLink new features

Following is a list of changes and additions to the software introduced in version 8.0.2.

- **Support for .NET** MobiLink now supports Visual Studio .NET programming languages for writing synchronization scripts.

  ☞ For more information, see "Writing Synchronization Scripts in .NET" [*MobiLink Synchronization User's Guide*, page 251], "-sl dnet option" [*MobiLink Synchronization Reference*, page 17], "ml_add_dnet_connection_script" [*MobiLink Synchronization Reference*, page 264], and "ml_add_dnet_table_script" [*MobiLink Synchronization Reference*, page 265].

- **Start classes** You can now write Java and .NET code that executes at the time the MobiLink server starts the JVM or CLR, before the first synchronization.

  ☞ For more information, see "User-defined start classes" [*MobiLink Synchronization User's Guide*, page 237].

- **Maintain unique primary keys using UUIDs** A new way to maintain unique primary keys on remote databases is introduced with Universally Unique IDs (UUIDs, also known as GUIDs).

  ☞ For more information, see "Maintaining unique primary keys using UUIDs" [*MobiLink Synchronization User's Guide*, page 81].

- **New way to handle referential integrity violations** Two new client stored procedures, sp_hook_dbmlsync_download_ri_conflict and sp_hook_dbmlsync_download_log_ri_contlict, are introduced to help you manage referential integrity violations during download.

☞ For more information, see "sp_hook_dbmlsync_download_ri_violation" [*MobiLink Synchronization Reference,* page 280] and "sp_hook_dbmlsync_download_log_ri_violation" [*MobiLink Synchronization Reference,* page 279].

- **Simpler way to delete all rows in a remote table** You can now delete all the data in a remote table by including one row in the download_delete_cursor that has NULL in every primary-key column.

  ☞ For more information, see "Writing download_delete_cursor scripts" [*MobiLink Synchronization User's Guide,* page 58].

Performance and monitoring enhancements

- **MobiLink Monitor** A graphical tool, the MobiLink Monitor, has been introduced to allow you to see the time taken by every aspect of the synchronization, sorted by MobiLink user or by worker thread.

  ☞ For more information, see "MobiLink Monitor" [*MobiLink Synchronization User's Guide,* page 297].

- **Users can estimate number of upload rows to dbmlsync** A new dbmlsync command line option has been created, -urc, which allows you to improve synchronization performance by providing an estimate of the number of rows that will be uploaded.

  ☞ For more information, see "-urc option" [*MobiLink Synchronization Reference,* page 80].

- **Users can specify persistent HTTP/HTTPS connections** You can use the **persistent** option to tell MobiLink to attempt to use the same connection for all HTTP requests in a synchronization. This setting may improve performance. It should only be used when you are connecting directly to MobiLink, and not through an intermediate agent such as a proxy or redirector.

  ☞ For more information, see "CREATE SYNCHRONIZATION USER statement [MobiLink]" [*ASA SQL Reference,* page 351].

- **New ways to control warning messages** Three new dbmlsrv9 command line options have been created: -zw, -zwd, and -zwe. With -zw, you can control which levels of warning message you want reported. With -zwd, you can disable specific warning codes. With -zwe, you can enable specific that are disabled with -zw.

  ☞ For more information, see "-zw option" [*MobiLink Synchronization Reference,* page 31], "-zwd option" [*MobiLink Synchronization Reference,* page 32] and "-zwe option" [*MobiLink Synchronization Reference,* page 33].

- **New verbose logging options** The dbmlsync -v command line option has been altered and expanded. Now, using -v alone causes minimum verbosity. To get maximum verbosity, use -v+. There are also several new levels that can be specified to fine tune the information that is logged. These options are also available as extended options.

  ☞ For more information, see "-v option" [*MobiLink Synchronization Reference*, page 80].

Connection enhancements

- **Ping support** The remote database can now ping the MobiLink synchronization server.

  ☞ For more information, see "-pi option" [*MobiLink Synchronization Reference*, page 76] and "Ping synchronization parameter" [*UltraLite Database User's Guide*, page 168].

- **New synchronization stream** MobiLink now supports the HTTPS protocol. This new stream implements HTTP over SSL/TLS using RSA encryption, and is compatible with any other HTTPS server.

  ☞ For more information, see "-x option" [*MobiLink Synchronization Reference*, page 24] and "CREATE SYNCHRONIZATION USER statement [MobiLink]" [*ASA SQL Reference*, page 351].

- **New buffer_size option** You can now specify a maximum buffer size for a fixed length HTTP message with the **buffer_size** option.

  ☞ For more information, see "CREATE SYNCHRONIZATION USER statement [MobiLink]" [*ASA SQL Reference*, page 351].

- **Auto-dial for MobiLink clients** MobiLink clients running on Pocket PC 2002 or Windows desktop computers can now connect through dial-up network connections. Using scheduling, your remote can synchronize unattended. The new synchronization stream parameters are **network_name**, **network_connect_timeout**, and **network_leave_open**.

  ☞ For more information, see "CREATE SYNCHRONIZATION USER statement [MobiLink]" [*ASA SQL Reference*, page 351].

New Web server support

- **Servlet Redirector** MobiLink now supports Web servers that support the Java servlet API 2.2, including Apache Tomcat.

  ☞ For more information, see "Synchronizing Through a Web Server" [*MobiLink Synchronization User's Guide*, page 313].

Security enhancements

- **RSA cipher suite supported** You can now use RSA encryption as well as the existing elliptical-curve encryption for synchronization security. The utilities gencert and readcert support the RSA certificates as well as elliptical-curve certificates.

  ☞ For more information, see "Transport-Layer Security" [*MobiLink Synchronization User's Guide*, page 337].

- **gencert can sign pregenerated certificate requests** The certificate generation utility gencert has a new command line option that allows you to sign pregenerated certificate requests.

  ☞ For more information, see "Certificate generation utility" [*MobiLink Synchronization Reference*, page 311].

SQL Remote new features

SQL Remote version 8.0.2 includes the following new features.

- **Error logs sent to consolidated database** For improved troubleshooting of errors at remote sites, log information can be collected at the consolidated database.

  ☞ For more information, see "Troubleshooting errors at remote sites" [*SQL Remote User's Guide*, page 226].

UltraLite new features

UltraLite 8.0.2 introduces several new features:

- **UltraLite Component Suite** UltraLite database technology can now be used from new development platforms in an easy-to-use fashion. The UltraLite Component Suite brings UltraLite technology to users of eMbedded Visual Basic, AppForge MobileVB, and Java. The component for Java is an alternative to the UltraLite for Java described in this book. The component is not a 100% pure Java implementation, but instead uses native classes for better performance.

  ☞ For more information on the UltraLite Component Suite, see the separate documentation set. Click Start ➤ Programs ➤ SQL Anywhere 9 ➤ UltraLite ➤ Online Books.

- **Upgrading UltraLite databases** When deploying a new version of an application, you can now choose to upgrade the schema of UltraLite database to the schema of the new application.

  ☞ For more information, see "ULEnableGenericSchema function" [*UltraLite Embedded SQL User's Guide*, page 111].

- **Java runtime is thread-safe** The UltraLite Java runtime is now thread-safe, enabling the development of multi-threaded UltraLite applications.

- **Deleting UltraLite database files** You can delete an UltraLite database file from an application using the ULDropDatabase function.

  ☞ For more information, see the following:

- Embedded SQL: “ULDropDatabase function” [*UltraLite Embedded SQL User’s Guide*, page 109]
- C++ API: “Drop method” [*UltraLite Static C++ User’s Guide*, page 89]
- Java: “drop method” [*UltraLite Static Java User’s Guide*, page 65]

♦ **Universally unique identifiers** UltraLite databases can now use the UNIQUEIDENTIFIER Adaptive Server Anywhere data type. This type is a BINARY(16) used for storing universally unique identifiers (UUIDs or GUIDs). UNIQUEIDENTIFIER columns that use the NEWID function as a default value can guarantee unique primary keys across a whole MobiLink installation, as an alternative to GLOBAL AUTOINCREMENT.

☞ For more information, see “The NEWID default” [*ASA SQL User’s Guide*, page 82].

♦ **New security options for synchronization** Two new secure synchronization protocols are introduced in this release. HTTPS is HTTP implemented over a transport-layer security protocol, and RSA is a form of transport-layer security encryption used over HTTP or TCP/IP networks.

These security options use Certicom technology. Use of Certicom technology requires that you obtain the separately-licensable SQL Anywhere Studio security option and is subject to export regulations. For more information on this option, see “Welcome to SQL Anywhere Studio” [*Introducing SQL Anywhere Studio*, page 4].

☞ For more information about RSA encryption, see “Security synchronization parameter” [*UltraLite Database User’s Guide*, page 169]. For information about using RSA encryption from Java, see “Initializing the synchronization parameters” [*UltraLite Static Java User’s Guide*, page 45].

☞ For more information about HTTPS synchronization, see “Stream Type synchronization parameter” [*UltraLite Database User’s Guide*, page 173] and “HTTPS stream parameters” [*UltraLite Database User’s Guide*, page 186].

♦ **Reset last download time** To resynchronize previously downloaded data, for example to set an application to a clean state, you can reset the last download timestamp.

☞ For more information, see “ULResetLastDownloadTime function” [*UltraLite Embedded SQL User’s Guide*, page 127], and “ResetLastDownloadTime method” [*UltraLite Static C++ User’s Guide*, page 84].

- **Troubleshooting previous synchronizations** Functions are now available to obtain information about the success or failure of the most recent synchronization. This feature is particularly useful for Palm OS applications that use HotSync, in which case the synchronization is carried out externally to the application.

  ☞ For more information, see "GetSynchResult method" [*UltraLite Static C++ User's Guide,* page 79], and "ULGetSynchResult function" [*UltraLite Embedded SQL User's Guide,* page 116]. This feature is not yet available for UltraLite Java applications.

- **Generate more and smaller files** The -x option causes the UltraLite generator to write out more and smaller files for C/C++ projects. This option is to help in cases where the generated code is too large for the compiler to handle in a single file.

  ☞ For more information, see "The UltraLite generator" [*UltraLite Database User's Guide,* page 96].

- **Improved synchronization observer** The synchronization observer function has been enhanced. More states and fields have been added to the interface to enable the design of more responsive and informative synchronization dialogs.

Behavior changes in version 8.0.2

This section lists the behavior changes introduced in components of SQL Anywhere Studio version 8.0.2.

Adaptive Server Anywhere behavior changes

The following is a list of behavior changes from previous versions of the software.

- **Windows CE 2.11 no longer supported** Support has been dropped for the Windows CE 2.11 platform.
- **SH3 and SH4 chips no longer supported** Support for Windows CE devices using the SH3 and SH4 chips has been dropped.

  ☞ For a list of supported platforms, see "Operating system versions" [*Introducing SQL Anywhere Studio*, page 138], and "Windows and NetWare operating systems" [*Introducing SQL Anywhere Studio*, page 125].
- **OPTIMIZATION_GOAL setting** The default setting for the OPTIMIZATION_GOAL option is set to **all-rows** rather than **first-row**. This affects the execution plan chosen for some queries and so will change performance characteristics.

  ☞ For more information, see "OPTIMIZATION_GOAL option [database]" [*ASA Database Administration Guide*, page 613].
- **xp_cmdshell displays a command window on Windows operating systems** It is now possible to control whether xp_cmdshell starts a new window. The behavior change applies to databases created with or upgraded to version 8.0.2 or later. On older databases, the previous behavior of not displaying a command window is maintained. The new behavior is compatible with other databases such as Adaptive Server Enterprise and Microsoft SQL Server.

  You can hide the command window by specifying a second parameter in the call to **xp_cmdshell**.

  ☞ For more information, see "xp_cmdshell system procedure" [*ASA SQL Reference*, page 759].
- **Full-length English day names are recognized regardless of the language used by the database server** When creating events, the full-length English day names are recognized by the database server, regardless of the language (German, Chinese, etc.) the database server is using. This means that event definitions in the reload script will be recognized by a server running with a different language.

Events that use the abbreviated English day names (Mon, Tue, and so on) are not recognized by servers running in languages other than English.

☞ For more information, see "CREATE EVENT statement" [*ASA SQL Reference,* page 304].

- **OPTION settings validated** Integer options with minimum and maximum values are now validated. Setting an option to an invalid value gives the error "Invalid setting for option '%1"' [*ASA Error Messages,* page 239].

  If you unload and reload a database that contains invalid option settings, they are set to the closest legal value.

  The affected options are as follows. The square brackets indicate an inclusive range.

Option	Range
ISOLATION_LEVEL	[0, 3]
PRECISION	[0, 127]
SCALE	[0, 127]
NEAREST_CENTURY	[0, 100]
MAX_HASH_SIZE	[2, 64]
MAX_WORK_TABLE_HASH_SIIZE	[2, 64]
FIRST_DAY_OF_WEEK	[1, 7]
DEFAULT_TIMESTAMP_INCREMENT	[1, 60 000 000]

- **Renamed joins** The names of two joins have changed, both in the graphical plan and in the documentation. Nested loops join not exists (JNE) are now called Nested loops antisemijoin (JNLA), and nested loops exists joins (JE) are now called nested loops semijoins (JNLS)

  ☞ For more information, see "Join algorithms" [*ASA SQL User's Guide,* page 381].

Deprecated and unsupported features

This list includes features that are no longer supported and that impact existing applications.

- **-d server option deprecated on Windows** When used on NetWare, the -d option forces the use of POSIX I/O rather than DFS (Direct File System) I/O. In Windows, the option is still allowed on the command line, but is ignored.

☞ For more information, see "-d server option" [*ASA Database Administration Guide*, page 141].

MobiLink behavior changes

The following is a list of behavior changes from previous versions of the software.

- **serial communications protocol no longer supported** The serial protocol is no longer supported. In its place, you can use HTTP, HTTPS, or TCP/IP.
- **Certicom no longer a certificate-issuing authority** You can no longer obtain transport-layer security certificates from Certicom. However, you can continue to use the Certicom reqtool utility to generate certificate requests, and you can purchase the certificates from a variety of other sources, including VeriSign and Entrust Technologies.

  ☞ For more information, see *http://www.verisign.com/* or *http://www.entrust.com/certificate_services/index.htm.*
- **dbmlsrv option -vw deprecated** The -vw dbmlsrv command line option, which was used to suppress warning messages, has been deprecated. In its place, you can use -zw or -zwd.

  ☞ For more information, see "-zw option" [*MobiLink Synchronization Reference*, page 31] and "-zwd option" [*MobiLink Synchronization Reference*, page 32].
- **dbmlsync option -v behavior change** The -v dbmlsync command line option has been altered and expanded. Now, using -v alone causes minimum verbosity.

  ☞ For more information, see "-v option" [*MobiLink Synchronization Reference*, page 80].
- **Full-length English day names are recognized regardless of the language used by the synchronization server** When creating schedules for MobiLink users, publications, and subscriptions, or when specifying scheduling information on the dbmlsync command line, you must use the full-length form of English day names (such as Monday) if you want the schedule to be recognized by a synchronization server running in a language other than English.

  Schedules that use the abbreviated English day names (such as Mon) are not recognized by synchronization servers running in languages other than English.

  ☞ For more information, see "CREATE SYNCHRONIZATION USER statement [MobiLink]" [*ASA SQL Reference*, page 351].

- **Better support for long data in dbmlsync** DBMLSync now handles blobs in a much more efficient way while building the upload stream. Blobs are now read into memory in pieces, so the ability to handle long blobs is no longer limited by available memory. When multiple publications are synchronized at one time, blob data is stored one time and shared between the upload streams. The output log now prints the size of the blob and its first 32 bytes.
- **HTTP option use_cookies removed** The use_cookies option has been removed. If you use it, the option is ignored. MobiLink now automatically detects when it needs cookies.

UltraLite behavior changes

The following is a list of behavior changes from previous versions of the software.

- **Windows CE 2.11 no longer supported** Support has been dropped for the Windows CE 2.11 platform.
- **SH3 and SH4 chips no longer supported** Support for Windows CE devices using the SH3 and SH4 chips has been dropped.

  ☞ For a list of supported platforms, see "Operating system versions" [*Introducing SQL Anywhere Studio*, page 138], and "Windows and NetWare operating systems" [*Introducing SQL Anywhere Studio*, page 125].
- **serial communications protocol no longer supported** The serial protocol is no longer supported. The major use of serial synchronization was from clients on the Palm Computing Platform. These clients can use HotSync synchronization instead.
- **No transport-layer security on VxWorks** The Certicom libraries that provide transport-layer security for synchronization are no longer supported on the VxWorks operating system.
- **VxWorks 5.5 not supported** VxWorks 5.3 and 5.4 are the supported versions of the VxWorks operating system.

  VxWorks unsupported in version 9
  Support for the VxWorks platform is dropped entirely in version 9.
- **Certicom libraries require JDK 1.2** The Certicom security libraries have been updated with this release. The new libraries for Java applications require JDK 1.2, rather than JDK 1.1.4.

CHAPTER 4

What's New in Version 8.0.1

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in Adaptive Server Anywhere version 8.0.1.

Contents

New features in version 8.0.1

This section lists the new features introduced in components of SQL Anywhere Studio version 8.0.1.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 8.0.1. It provides an exhaustive listing of major and minor new features, with cross references to locations where each feature is discussed in detail.

- **Specify space to be reserved in table pages** You can reduce table fragmentation by specifying the percentage of free space that should be reserved in table pages.

  ☞ For more information, see "Table fragmentation" [*ASA SQL User's Guide*, page 193] and "ALTER TABLE statement" [*ASA SQL Reference*, page 250].

To specify the percentage of space to be allocated on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

- **New system tables** Two new system tables, SYSATTRIBUTE and SYSATTRIBUTENAME, have been added.

  ☞ For more information, see "SYSATTRIBUTE system table" [*ASA SQL Reference*, page 618] and "SYSATTRIBUTENAME system table" [*ASA SQL Reference*, page 620].

- **sa_disk_free_space system procedure** This procedure allows you to determine the space available for your dbspaces, temporary file, transaction log, and transaction log mirror.

  ☞ For more information, see "sa_disk_free_space system procedure" [*ASA SQL Reference*, page 714].

- **sa_flush_statistics system procedure** Database administrators can use this procedure to ensure that cost model statistics that exist only in the database server cache are flushed out.

  ☞ For more information, see "sa_flush_statistics system procedure" [*ASA SQL Reference*, page 716].

- **New ways to obtain server message window contents** There is a new system procedure and three new properties that return information from the server window.

☞ For more information, see "sa_get_server_messages system procedure" [*ASA SQL Reference,* page 719]; and MessageText, MessageTime, and MessageWindowSize in "Server-level properties" [*ASA Database Administration Guide,* page 657].

- **Determine ANSI equivalency of non-ANSI statements** The REWRITE function accepts a new argument, ANSI, which causes the function to return the ANSI equivalent of any SELECT, UPDATE, or DELETE statement.

  ☞ For more information, see "REWRITE function [Miscellaneous]" [*ASA SQL Reference,* page 177].

- **Variable assignment allowed in UPDATE statement** The SET clause of the UPDATE statement can now be used to assign a value to a variable, in addition to updating the table. This feature is compatible with Adaptive Server Enterprise.

  ☞ For more information, see "UPDATE statement" [*ASA SQL Reference,* page 592].

- **Alternative to autoincrement** The get_identity function is provided as an alternative for allocating identity values to autoincrement columns.

  ☞ For more information, see "GET_IDENTITY function [Miscellaneous]" [*ASA SQL Reference,* page 134].

- **Square brackets can delimit identifiers** You can use square brackets to delimit identifiers. Square brackets can always be used, regardless of the setting of the QUOTED_IDENTIFIER option.

  ☞ For more information, see "Identifiers" [*ASA SQL Reference,* page 7].

- **Specify isolation level in FROM clause** You can use the WITH table-hint argument to specify a locking method for a particular table or view for a particular SELECT, UPDATE, or DELETE statement.

  ☞ For more information, see "FROM clause" [*ASA SQL Reference,* page 445].

- **Data Migration wizard** The Data Migration wizard allows you to migrate remote tables to an Adaptive Server Anywhere database from Sybase Central.

  You cannot migrate foreign keys if the target database is version 8.0.0 or earlier. To migrate foreign keys, you must upgrade the target database's file format by unloading and reloading the database.

  ☞ For more information, see "Migrating databases to Adaptive Server Anywhere" [*ASA SQL User's Guide,* page 548].

- **Unload a version 5.x or 6.x database from Sybase Central** Sybase Central now allows you to connect to a version 5.x of 6.x database in order to upgrade the database file format using the Unload Database wizard. To do this, you must run the database on a version 8.0.0 or later server.

  ☞ For more information, see "Upgrading Adaptive Server Anywhere" on page 176.

- **Back up and shut down your database from the Upgrade Database wizard** You can now back up your database files, including the main database file, the transaction log, and dbspaces from the Sybase Central Upgrade Database wizard. The wizard also allows you to shut down your database when the upgrade is complete.

  ☞ For more information, see "Upgrading a database" on page 177.

- **sa_migrate enhancement** The sa_migrate procedure has an optional argument, *migrate_fkeys* that allows you to specify whether or not you want to migrate foreign key mappings when you migrate tables from a remote database. In previous releases, foreign key mappings were always migrated when you used the sa_migrate procedure.

  ☞ For more information, see "sa_migrate system procedure" [*ASA SQL Reference*, page 724].

  To use this feature on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

- **New SORT_COLLATION database option** The SORT_COLLATION database option allows implicit use of the SORTKEY function on ORDER BY expressions. When the value of this option is set to a valid *collation name* or *collation ID*, any string expression in the ORDER BY clause is treated as if the SORTKEY function had been invoked.

  ☞ For more information, see "SORT_COLLATION option [database]" [*ASA Database Administration Guide*, page 624].

- **Use an IP address/port to connect to a server** You can use the VerifyServerName=NO connection parameter to skip the verification of the server name and allow Adaptive Server Anywhere clients to connect to an Adaptive Server Anywhere server if they know only an IP address/port. The VerifyServerName parameter is only used if DoBroadcast=NONE is specified.

  ☞ For more information, see "VerifyServerName communication parameter [VERIFY]" [*ASA Database Administration Guide*, page 218] .

- **New LocalOnly connection parameter controls broadcasts** You can use the LocalOnly connection parameter to connect only to a server on

the local machine, if one exists. Setting LocalOnly=YES uses the regular broadcast mechanism, except that broadcast responses from servers on other machines are ignored.

☞ For more information, see "LocalOnly communication parameter [LOCAL]" [*ASA Database Administration Guide*, page 209].

- **Specify how much of the cache is used for pinning cursors** You can use the PINNED_CURSOR_PERCENT_OF_CACHE option to adjust the amount of cache that can be used for pinning cursors. Lowering the limit can improve performance in low memory environments.

  ☞ For more information, see "PINNED_CURSOR_PERCENT_OF_CACHE option [database]" [*ASA Database Administration Guide*, page 617].

- **Monitor database file and log file fragmentation** You can use the DBFileFragments and LogFileFragments database properties to choose monitor file fragmentation. Fragmentation of the transaction log file is usually not a significant concern; however, fragmentation of the database file can be a cause of reduced performance and may warrant use of a disk defragmentation utility.

  ☞ For more information, see "Database-level properties" [*ASA Database Administration Guide*, page 664].

- **New connection properties.** Two new connection properties have been added. LivenessTimeout returns the liveness timeout of the connection, and IdleTimeout returns the idle timeout of the connection.

  ☞ For more information, see "Connection-level properties" [*ASA Database Administration Guide*, page 647].

- **New server properties** The new IdleTimeout server property returns the default idle timeout value.

  ☞ For more information, see "Server-level properties" [*ASA Database Administration Guide*, page 657].

- **Non-deterministic functions** Functions that modify underlying data, or that rely on underlying data that may change during the course of query execution, can be declared NOT DETERMINISTIC. Functions that are declared this way are re-evaluated each time they are called during query execution. Otherwise, the function value is cached and re-used for better performance.

  ☞ For more information, see "CREATE FUNCTION statement" [*ASA SQL Reference*, page 315].

- **Ensure all transactions in backup are complete** By default, the BACKUP statement renames or truncates the transaction log without waiting for open transactions to complete. You can now ensure that all transactions contained in a backup are complete by specifying a WAIT AFTER END clause.

  ☞ For more information, see "BACKUP statement" [*ASA SQL Reference*, page 263]

MobiLink new features

Following is a list of changes and additions to the software introduced in version 8.0.1.

- **Full error context reporting** The MobiLink synchronization server now shows the full error context in its output file when an error occurs during synchronization.

  ☞ For more information, see "-o option" [*MobiLink Synchronization Reference*, page 13].

- **User ID mapping** MobiLink now allows you to more readily find a database user ID or map a MobiLink username to a user ID.

  ☞ For more information, see "modify_user connection event" [*MobiLink Synchronization Reference*, page 184].

- **Set address and type as client options** The MobiLink client now allows you specify the communication type and address on the command line to connect to the MobiLink synchronization server.

  ☞ For more information, see "-e extended options" [*MobiLink Synchronization Reference*, page 44].

- **Log MobiLink-issued ODBC statements** You can instruct MobiLink to log to an ODBC output file all the ODBC statements issued by MobiLink.

  ☞ For more information, see "-t option" [*MobiLink Synchronization Reference*, page 19].

- **Modify the download timestamp** You can modify the last download timestamp or the next last download timestamp in two new events.

  ☞ For more information, see "modify_last_download_timestamp connection event" [*MobiLink Synchronization Reference*, page 180] and "modify_next_last_download_timestamp connection event" [*MobiLink Synchronization Reference*, page 182].

- **Automatic timestamp conflict tolerance** In the event of a timestamp conflict between the consolidated and remote database, this option allows timestamp values with a precision higher than the lowest-precision to be used for conflict detection purposes.

  ☞ For more information, see "-zp option" [*MobiLink Synchronization Reference*, page 30].

SQL Remote new features

SQL Remote version 8.0.1 includes the following new features.

- **SMTP user authentication** Parameters are provided for separate user authentication on SMTP servers when using the SMTP/POP message system.

  ☞ For more information, see "The SMTP message system" [*SQL Remote User's Guide*, page 218].

UltraLite new features

UltraLite 8.0.1 introduces several new features:

- **CodeWarrior 8 support** This release supports CodeWarrior version 8.
- **Support for multi-threaded applications** UltraLite applications can now be multi-threaded on platforms that support this kind of application.
- **Pocket PC 2002 support** Pocket PC 2002 is added to the list of supported platforms.
- **JDBC ResultSet methods added** The ResultSet.findColumn and ResultSet.get*Type* methods are now supported.

  ☞ For more information, see "JDBC features in UltraLite" [*UltraLite Static Java User's Guide*, page 58].
- **Access to information from UltraLite Java** The **JdbcConnection.getLastIdentity** method, **getLastDownLoadTime** method, and **JdbcDatabase.countUploadRows** method allow access to useful information. These features were previously available only in C/C++ applications.

  ☞ For more information, see "Class JdbcConnection" [*UltraLite Static Java User's Guide*, page 59].
- **User authentication in UltraLite Java** The Java version of UltraLite now supports user authentication.

  ☞ For more information, see "Adding user authentication to your application" [*UltraLite Static Java User's Guide*, page 38], "Class

JdbcSupport" [*UltraLite Static Java User's Guide,* page 67], and "Class JdbcDatabase" [*UltraLite Static Java User's Guide,* page 64].

- **HotSync synchronization progress displayed** The status field of the HotSync Progress dialog on your desktop computer now shows the progress of synchronization with UltraLite applications.

- **HotSync configuration** You can configure the HotSync conduit from Palm Desktop.

  ☞ For more information, see "Configuring the MobiLink HotSync conduit" [*MobiLink Synchronization User's Guide,* page 214].

- **Automatic scripting from UltraLite applications** UltraLite applications can now provide column names to the MobiLink synchronization server so that synchronization scripts can be automatically generated.

- **Get SQL data type of a column from the C++ API** The GetColumnSQLType method returns the data type of a column.

  ☞ For more information, see "GetColumnSQLType method" [*UltraLite Static C++ User's Guide,* page 100].

- **Optional checkpoint during synchronization** Synchronizations that download large numbers of updates can cause the UltraLite database to grow significantly in size. This growth can be limited by carrying out checkpoints during synchronization. The new **checkpoint_store** synchronization parameter controls checkpointing. By default, no checkpoints are carried out.

  ☞ For more information, see "Checkpoint Store synchronization parameter" [*UltraLite Database User's Guide,* page 164].

Behavior changes in version 8.0.1

This section lists the behavior changes introduced in components of SQL Anywhere Studio version 8.0.1.

Adaptive Server Anywhere behavior changes

The following is a list of behavior changes from previous versions of the software.

- **New naming convention for renamed transaction log files** Double digits at the end of transaction log files renamed during backup have been changed to double characters. For example, the renamed log file from the first backup on December 10, 2000, is now named *001210AA.log* instead of *00121001.log*. The first two digits indicate the year, the second two digits indicate the month, the third two digits indicate the day of the month, and the final two characters distinguish among different backups made on the same day. This increases the number of backups possible in a day from 100 to 676.

- **LOAD TABLE now recalculates computed columns** LOAD TABLE now detects computed columns and evaluates them for each row inserted into the table.

- **DBCONSOLE now allows connections to be reconnected** Previously a DBConsole session only allowed one connection. Connections can now be disconnected and reconnected without exiting the application.

Deprecated and unsupported features

This list includes features that are no longer supported and that impact existing applications.

- **DEBUG connection parameter deprecated** The DEBUG connection parameter has been deprecated. You can still use LOG parameter to create a log file containing the debug information. From version 8.0.1 on, `LOG=filename` does what `DEBUG=YES;LOG=filename` used to do.

  ☞ For more information, see "Connection parameters" [*ASA Database Administration Guide*, page 70].

- **AGENT connection parameter deprecated** The AGENT connection parameter has been deprecated. You can use the CommLinks parameter with appropriate communication parameters to achieve the same behavior.

  ☞ For more information, see "Connection parameters" [*ASA Database Administration Guide*, page 70].

- **Port connection property removed** The port connection parameter has been removed.
- **Adaptive Server Anywhere Translation Driver removed** Use of translation drivers is no longer recommended. The server automatically handles character set translation.
- **SharedMemory tried first** The ports specified in the LINKS= connection parameter were tried in the order in which they were specified. Now, if the sharedmemory (shmem) port is specified, it is tried first, followed by the other ports specified in the order in which they appear.
- **GLOBAL AUTOINCREMENT** The default value has been changed from 0 to 2147483647. GLOBAL_DATABASE_ID can now be set to 0 and will cause values to be generated starting at 1.

MobiLink behavior changes

- **Timestamp mismatch notification** When the timestamps between consolidated and remote databases are at variance, the MobiLink synchronization server will log a warning with each synchronization.
- **GLOBAL AUTOINCREMENT** The default value has been changed from 0 to 2147483647. GLOBAL_DATABASE_ID can now be set to 0 and will cause values to be generated starting at 1.

  It is still the case that if GLOBAL_DATABASE_ID is not set, or is set to the default value, attempts to cause a global autoincrement value to be generated result in a NULL. This commonly gives an error when attempting to insert the value into a non-nullable primary key column and is the indication that the GLOBAL_DATABASE_ID option has not been set.

  Disallowing a setting of 0 for GLOBAL_DATABASE_ID prevented generation of values starting at 1. Instead, values would start at the partition size specified for the column.

  ☞ For more information, see “GLOBAL_DATABASE_ID option [database]” [*ASA Database Administration Guide*, page 595].
- **dbmlstop performs soft shutdown** By default (if none of -w, -f, -h or -t are specified), dbmlstop does a soft shutdown. This means that it stops accepting new connections and exits when the current synchronizations are complete.

☞ For more information, see “MobiLink stop utility” [*MobiLink Synchronization Reference*, page 303].

UltraLite behavior changes

- **Palm database backup** In previous releases, if the ULUtil application was used to backup a database, the database would be backed up on each subsequent HotSync operation.

  Most UltraLite data is effectively backed up by synchronization. As the most common use of an explicit backup is to create an initial database for deployment, continuing to make backups on HotSync is not the desired behavior in most cases. Now, each time an UltraLite application starts, it disables backups on future HotSync operations.

  If you wish to explicitly require backups for databases every time a HotSync is performed, you can do so by setting the **palm_all_backup** parameter in the UL_STORE_PARMS macro.

  ☞ For more information, see "UL_STORE_PARMS macro" [*UltraLite Database User's Guide*, page 216].

Deprecated and unsupported features

UltraLite support for synchronization on the Palm Computing Platform using ScoutSync technology is deprecated. Version 8.0.x will continue to support ScoutSync up to version 3.6, but the next major release of SQL Anywhere Studio will not support ScoutSync.

CHAPTER 5

What's New in Version 8.0.0

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in Adaptive Server Anywhere version 8.0.0.

Contents

New features in version 8

This section lists the new features introduced in components of SQL Anywhere Studio version 8.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 8.0. It provides an exhaustive listing of major and minor new features, with cross references to locations where details of each feature appear in the manuals.

If you have the printed version of this book, and if you do not have the complete SQL Anywhere Studio documentation set, you should look in the online documentation for a detailed description of each feature.

Some new features require that you upgrade the database to version 8, or that you upgrade the database file format by unloading and reloading the database. If a database upgrade or file format upgrade is required to access a particular feature, the requirement is indicated in the description below.

☞ For information on how to carry out these tasks, see "Upgrading Adaptive Server Anywhere" on page 176.

The Adaptive Server Anywhere new features are grouped under the following headings:

Query processing and database performance

- **Improved query processing** This version includes enhancements to the query execution engine and the optimizer, resulting in a significant

improvement in performance, especially for complex queries. Enhancements to Adaptive Server Anywhere query processing include the following:

- More sophisticated internal processing of joins.
- Improvements to the optimizer's cost model used to assess alternative access plans.
- Improvements to the execution model.

  ☞ Most of these changes are internal. Documentation is provided in "Query Optimization and Execution" [*ASA SQL User's Guide*, page 367].

An effect of these changes is that it is no longer the case that the materialization of results is necessarily inefficient. Use of temporary work tables may be a very efficient way to execute a query. For more information, see "Use of work tables in query processing" [*ASA SQL User's Guide*, page 185].

The optimizer now performs cost-based selection of indexes, and does not solely rely on predicate selectivities as was the case with prior releases.

Much of the improved query processing does not require an upgraded database. To use the new cost model on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

♦ **New index type** A new type of index has been added that improves performance for multiple column indexes and for indexes that include wide columns. It is a compressed B-tree index.

Adaptive Server Anywhere automatically creates the appropriate type of index based on index width (the sum of the width of all columns in the index). A compressed B-tree index is created when the width of the index is greater than nine bytes and less than one-eighth of the page size to a maximum of 256 bytes; otherwise, Adaptive Server Anywhere creates hash B-tree indexes.

The WITH HASH SIZE clause of the CREATE INDEX statement is deprecated.

☞ For more information about these indexes, see "Types of index" [*ASA SQL User's Guide*, page 401].

☞ To use the new index types on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

☞ A new limitation is imposed: foreign key indexes must have the same size and type as the corresponding primary key index.

☞ *dbunload* now omits the hash size specification if it was originally specified with the default (WITH HASH SIZE 10).

- **New database option OPTIMIZATION_GOAL** Determines whether query processing is optimized towards returning the first row quickly, or minimizing the cost of returning the complete result set. The default is to optimize for the first rows.

  ☞ For more information, see "OPTIMIZATION_GOAL option [database]" [*ASA Database Administration Guide*, page 613].

- **Performance enhancements for table scans** Databases created in Adaptive Server Anywhere 8.0 with 2K, 4K, or 8K pages have performance-enhancements for queries that require sequential table scans. Adaptive Server Anywhere creates bitmaps, also known as page maps, for large tables. A bitmap lists all of the pages containing data for a given table. This feature permits searching large tables in only one I/O operation.

  ☞ For more information, see "Table and page sizes" [*ASA SQL User's Guide*, page 393].

  ☞ To gain the benefits of this enhancement on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

- **Improved storage of checkpoint log** The checkpoint log is now stored in consecutive pages at the end of the database file. This leads to improved performance by allowing sequential scans and multipage writes of the material in the checkpoint log.

  ☞ For more information about the checkpoint log, see "Checkpoints and the checkpoint log" [*ASA Database Administration Guide*, page 360].

  ☞ To gain the benefits of this enhancement on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

- **Plan caching** Adaptive Server Anywhere now caches execution plans for queries and INSERT, UPDATE and DELETE statements performed inside stored procedures, user-defined functions, and triggers. The maximum number of plans to cache is specified with the option setting MAX_PLANS_CACHED. To disable plan caching, set this option to 0.

  ☞ For more information, see "Access plan caching" [*ASA SQL User's Guide*, page 375].

- **Overriding the default I/O cost model** You can now override the default I/O cost model using the ALTER DATABASE statement with the CALIBRATE clause.

  ☞ For more information, see "ALTER DATABASE statement" [*ASA SQL Reference*, page 225].

- **New database option MAX_PLANS_CACHED** Sets the maximum number of execution plans that are stored in cache.

  ☞ For more information, see "MAX_PLANS_CACHED option [database]" [*ASA Database Administration Guide*, page 606].

- **New database option MIN_TABLE_SIZE_FOR_HISTOGRAM** This option sets the minimum table size for which histograms are created. Histograms store information about the distribution of values in a column, and the optimizer uses them to choose an efficient execution plan.

  ☞ For more information, see "MIN_TABLE_SIZE_FOR_HISTOGRAM option [database]" [*ASA Database Administration Guide*, page 609].

Security

- **Strong encryption over TCP/IP** Adaptive Server Anywhere now supports certificate-based encryption over TCP/IP ports on Solaris, Linux, NetWare, and all supported Windows operating systems with the exception of Windows CE. Strong encryption protects the confidentiality and integrity of network packets as they pass between the client and the server. This encryption is also called Transport Layer Security (TLS).

  The database server `-ec` command line option allows you to set the server's connection parameters and replaces the `-e` command line option in previous versions of Adaptive Server Anywhere. You can set the client connection parameters with the encryption connection parameter.

  ☞ For more information, see "-ec server option" [*ASA Database Administration Guide*, page 141] and "Encryption connection parameter [ENC]" [*ASA Database Administration Guide*, page 188].

  ☞ To use this feature, you must use version 8 software at both the client and the server. You do not need to upgrade the database.

- **Strong encryption of the database file** The database file itself can now be strongly encrypted for greater security, especially on notebook and laptop computers prone to theft.

  ☞ For more information, see the following locations:

  - "Creating a database using the dbinit command-line utility" [*ASA Database Administration Guide*, page 486]
  - "-ek database option" [*ASA Database Administration Guide*, page 168]
  - "-ep server option" [*ASA Database Administration Guide*, page 143]
  - "CREATE DECRYPTED FILE statement" [*ASA SQL Reference*, page 299].

    ☞ You must use version 8 software to create encrypted database files.

SQL features

- **Full outer joins** Full outer joins are now supported. In addition, the keyword OUTER is now optional for right, left, and full outer joins.

  ☞ For more information, see "Outer joins" [*ASA SQL User's Guide,* page 274].

- **CASE statements** The ANSI standard allows two forms of CASE statements. Adaptive Server Anywhere 8.0 supports both syntaxes.

  ☞ For more information, see "CASE statement" [*ASA SQL Reference,* page 275].

- **WAITFOR statement** This statement delays processing for the current connection for a specified amount of time or until a given time.

  ☞ For more information, see "WAITFOR statement" [*ASA SQL Reference,* page 605].

- **RAISERROR statement allows connections to be disallowed** This statement can now be used to disallow or limit connections.

  ☞ For more information, see "RAISERROR statement [T-SQL]" [*ASA SQL Reference,* page 515].

- **Timezone adjustment** To permit easier coordination of date/time values across time zones, the following new features have been added:
  - **CURRENT UTC TIMESTAMP** Adjusts the time zone value by the server's time zone adjustment value.
  - **DEFAULT UTC TIMESTAMP** Specifies a default value for INSERTs and sets updated columns to the value.
  - **TimeZoneAdjustment property** returns the number of minutes that must be added to the Coordinated Universal Time (UTC) to display the new local time.
  - **TIME_ZONE_ADJUSTMENT option** Allows a connection's time zone adjustment to be modified.

- **New collation functions** The SORTKEY function generates values that can be used to sort character data. SORTKEY allows you to perform sorting beyond the default behavior of Adaptive Server Anywhere collation.

  The COMPARE function allows you to directly compare two character strings based on alternate collation rules.

  ☞ For more information, see "SORTKEY function [String]" [*ASA SQL Reference,* page 184] and "COMPARE function [String]" [*ASA SQL Reference,* page 105].

- **ERRORMSG function** The new SQL function ERRORMSG can be used to obtain error messages.

  ☞ For more information, see "ERRORMSG function [Miscellaneous]" [*ASA SQL Reference*, page 124].

- **Data type conversion functions** The ISDATE and ISNUMERIC functions test if a string can be converted to a date or number, respectively.

  ☞ For more information, see "ISDATE function [Data type conversion]" [*ASA SQL Reference*, page 144], and "ISNUMERIC function [Miscellaneous]" [*ASA SQL Reference*, page 145]

Development and administration tools

- **Accessibility features** SQL Anywhere Studio is compliant with Section 508 of the US Federal Rehabilitation Act. The user interfaces and documentation have been prepared in compliance with the act. An accessibility enablement component provides software that enables the use of accessibility tools. The accessibility enablement component is not installed by default.

  ☞ For more information, see "Welcome to SQL Anywhere Studio" [*Introducing SQL Anywhere Studio*, page 4].

- **Query Editor** A graphical query editor has been added to Interactive SQL. With the Query Editor, you can create or edit SELECT statements without using SQL code. You can open the Query Editor in Interactive SQL by clicking Tools ➤ Edit Query.

  ☞ For more information, see "Introducing the Query Editor" [*SQL Anywhere Studio Help*, page 190].

- **Editable data in Interactive SQL and Sybase Central** You can update the database by editing Interactive SQL result sets, and by editing tables and views in Sybase Central. You can copy, edit, insert, and delete row values.

  Data displayed in Sybase Central can be copied to the clipboard.

  ☞ For more information, see "Editing table values in Interactive SQL" [*ASA Getting Started*, page 76].

- **Interactive SQL supports SQL escape syntax handling** Interactive SQL now supports JDBC escape syntax that allows you to access a library of functions implemented by the JDBC driver.

  ☞ For more information, see "Using JDBC escape syntax" [*ASA Programming Guide*, page 131].

- **Procedure profiling** Sybase Central contains a Profile tab that displays information about the number of calls and execution times for stored procedures, functions, events, and triggers. You can also view information about the execution speed for each line within a procedure. Profiling information is available through Sybase Central and SQL stored procedures.

  ☞ For more information about viewing procedure profiling information in Sybase Central, see "Profiling database procedures" [*ASA SQL User's Guide,* page 197].

  ☞ For more information about obtaining procedure profiling information with SQL stored procedures, see "sa_procedure_profile_summary system procedure" [*ASA SQL Reference,* page 736] and "sa_procedure_profile system procedure" [*ASA SQL Reference,* page 735].

  ☞ To use this feature, you must upgrade the database.

- **Improved information for access plans** There are two new ways to view the plan, a graphical display and a graphical display with statistics. These new plans provide more information about the processing cost of your query, and allow you to examine the cost of subsets of the query. The default access plan is now the graphical plan. The long and short plans are now based on the Ariadne syntax used by Adaptive Server Enterprise, and have new abbreviations.

  ☞ For more information, see "Reading access plans" [*ASA SQL User's Guide,* page 420].

- **Results pane displays query execution plan** The Interactive SQL Results pane now has a Results tab. The Results tab displays the results of your query, and the Plan tab displays the execution plan for the query. Previously, the query execution plan appeared in the Interactive SQL Messages pane.

  ☞ For more information, see "Interactive SQL" [*ASA Getting Started,* page 30].

- **Results pane displays UltraLite plan** The Interactive SQL Results pane now has an UltraLite Plan tab. This tab displays the UltraLite plan optimization strategy in XML format, as a string.

  ☞ For more information, see "GRAPHICAL_ULPLAN function [Miscellaneous]" [*ASA SQL Reference,* page 137].

- **XML export using the OUTPUT statement** You can export query results as XML format. The output has an embedded DTD. Binary values

are encoded in CDATA blocks with the binary data rendered as two-hexadecimal-digit strings.

☞ For more information, see "OUTPUT statement [Interactive SQL]" [*ASA SQL Reference*, page 501].

- **Interactive SQL batch options** Additional control is given to Interactive SQL when running batch files, through the `-codepage` and `-onerror` command line options. Also, the `-d1` command line option provides feedback useful for debugging batch files.

  ☞ For more information, see "The Interactive SQL utility" [*ASA Database Administration Guide*, page 492].

Application development

- **New cursor types** The cursors supplied by Adaptive Server Anywhere have been enhanced to provide cleaner semantics, to better match new cursor types such as keyset-driven cursors, and to take advantage of the new query optimization possibilities.

  ☞ For more information, see "Adaptive Server Anywhere cursors" [*ASA Programming Guide*, page 30].

- **Improved fetching for long columns** The amount of data that can be fetched in a single operation has been increased from 32 kb to a configurable value with a default of 256 kb. In ODBC the value can be set using the SQL_ATTR_MAX_LENGTH statement attribute. In embedded SQL, use the DT_LONGVARCHAR and DT_LONGBINARY types.

  ☞ For more information, see "Retrieving data" [*ASA Programming Guide*, page 248], and "Sending and retrieving long values" [*ASA Programming Guide*, page 190].

- **New embedded SQL function to obtain database properties** The function **db_get_property** can be used to obtain database properties.

  ☞ For more information, see "db_get_property function" [*ASA Programming Guide*, page 212]. For information on database properties, see "Database properties" [*ASA Database Administration Guide*, page 647].

- **BLOCKING_TIMEOUT option** The new BLOCKING_TIMEOUT option lets you control how long a transaction waits to obtain a lock.

  ☞ For more information, see "BLOCKING_TIMEOUT option [database]" [*ASA Database Administration Guide*, page 581].

- **RETURN_DATE_TIME_AS_STRING option** The RETURN_DATE_TIME_AS_STRING option allows you to control how

date, time, and timestamp values are returned over jConnect and Open Client.

☞ For more information, see "RETURN_DATE_TIME_AS_STRING option [database]" [*ASA Database Administration Guide*, page 622].

Administration and troubleshooting

In addition to the administration enhancements added to Sybase Central, listed above, version 8 includes the following administration enhancements.

- **Improve table performance without disrupting access** The REORGANIZE TABLE statement can be used to improve performance when a full rebuild of the database is not possible, due to the requirements for continuous access to the database. Use this statement to defragment rows in a table, or to compress indexes which have become sparse due to DELETEs. It can also reduce the total number of pages used to store the table and its indexes, as well as reduce the number of levels in an index tree.

  To reorganize tables based on a primary key, foreign key, or index, the database must be Adaptive Server Anywhere version 7 or above.

  ☞ For more information, see "REORGANIZE TABLE statement" [*ASA SQL Reference*, page 522].

- **Fast database validation** A new type of validation check has been added that reduces the amount of time it takes to validate a database. This option is of particular interest to people who need to validate large databases with small cache sizes. Affected tools include the sa_validate system procedure, the Validation utility (**dbvalid**) and the VALIDATE TABLE statement.

  ☞ For more information, see "Improving performance when validating databases" [*ASA Database Administration Guide*, page 365].

  ☞ To use this feature on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

- **Backup does not need to wait for outstanding transactions to complete** If a backup instruction requires the transaction log to be truncated or renamed, uncommitted transactions are carried forward to the new transaction log. This means that the server no longer waits for outstanding transactions to be committed or rolled back before initiating a backup.

  For more information, see "Translating a transaction log using the dbtran command-line utility" [*ASA Database Administration Guide*, page 508] and "Backup internals" [*ASA Database Administration Guide*, page 359].

☞ To use this feature on databases created before this release, you must upgrade the database file format by unloading and reloading the database.

- **Obtaining fragmentation statistics** File, table, and index fragmentation can all decrease performance. In Adaptive Server Anywhere 8.0 when you start a database on Windows NT, the server automatically displays information about the number of file fragments in each dbspace.

  The new system procedures, sa_table_fragmentation and sa_index_density, allow database administrators to obtain information about the fragmentation in a database's tables and indexes.

  ☞ For more information about file fragmentation, see "File fragmentation" [*ASA SQL User's Guide*, page 193].

  ☞ For more information about table fragmentation, see "Table fragmentation" [*ASA SQL User's Guide*, page 193] and "sa_table_fragmentation system procedure" [*ASA SQL Reference*, page 744].

  ☞ For more information about index fragmentation, see "Index fragmentation" [*ASA SQL User's Guide*, page 195] and "sa_index_density system procedure" [*ASA SQL Reference*, page 719].

- **Obtain the most recently prepared SQL statement for a connection** The database server `-zl` command line option turns on capturing of the most recently prepared SQL statement for each connection to databases on a server. You can also turn on this feature using the sa_server_option stored procedure with the remember_last_statement setting.

  When this feature is turned on, the **LastStatement** property function and the sa_conn_activity system procedure return the most recently prepared SQL statement for the current connection and all connections to databases on a server respectively.

  ☞ For more information, see "-zl server option" [*ASA Database Administration Guide*, page 166], "sa_conn_activity system procedure" [*ASA SQL Reference*, page 708], and "sa_server_option system procedure" [*ASA SQL Reference*, page 739].

- **-cw command line option** This server option lets you use cache sizes up to 64 Gb on Windows 2000, Windows XP, and Windows Server 2003.

  ☞ For more information, see "-cw server option" [*ASA Database Administration Guide*, page 138].

- **-qp option** This server option lets you suppress messages about performance in the database server window.

  ☞ For more information, see "-qp server option" [*ASA Database Administration Guide*, page 156].

- **Improved debugging server log** The information logged in the connection debugger has been improved to give more context about the portion of the connection being attempted; to remove the CONN: prefix; to increase the number of TCP/IP messages.

- **Databases can hold more procedures** The primary key values for the SYSPROCEDURE, SYSPROCPARM, SYSPROCPERM, and SYSTRIGGER system tables have been changed from SMALLINT to UNSIGNED INT. This change increases the number of procedures that a database can hold.

  ☞ For more information about the number of procedures a database can hold, see "Size and number limitations" [*ASA Database Administration Guide*, page 674].

  To use this feature, you must upgrade the database file format.

- **Monitoring query performance** New system procedures and utilities have been included to measure query performance.

  ☞ For more information, see "sa_get_request_profile system procedure" [*ASA SQL Reference*, page 717], "sa_get_request_times system procedure" [*ASA SQL Reference*, page 718], and "Monitoring query performance" [*ASA SQL User's Guide*, page 195].

- **New diagnostic properties** Properties allow you to obtain information about connections, databases, and the current database server. The following connection properties have been added in this release:
  - UtilCmdsPermitted property
  - TempTablePages property
  - LastStatement property
  - PacketSize property
  - Max_plans_cached property
  - QueryCachePages property
  - QueryLowMemoryStrategy property
  - Min_table_size_for_histogram property

  ☞ For more information, see "Connection-level properties" [*ASA Database Administration Guide*, page 647].

  The following database properties have been added in this release:
  - DBFileFragments property
  - LogFileFragments property
  - BlobArenas property
  - SeparateForeignKeys property

- VariableHashSize property
- TableBitMaps property
- FreePageBitMaps property
- SeparateCheckpointLog property
- Histograms property
- LargeProcedureIDs property
- PreserveSource property
- TransactionsSpanLogs property
- Capabilities property
- TempTablePages property
- CompressedBTrees property
- ProcedurePages property
- QueryCachePages property
- QueryLowMemoryStrategy property

  ☞ For more information, see "Database-level properties" [*ASA Database Administration Guide*, page 664].

The following server properties have been added in this release:

- MachineName property
- IsJavaAvailable property
- PlatformVer property

  ☞ For more information, see "Server-level properties" [*ASA Database Administration Guide*, page 657].

♦ **Additional performance monitor statistics** Several performance monitor statistics have been added for this release.

☞ For more information, see "Database performance statistics" [*ASA Database Administration Guide*, page 638].

♦ **Login procedure allows connections to be disallowed** The LOGIN_PROCEDURE option allows a stored procedure to be called for each new connection. This procedure can now be used to disallow database connections.

☞ For more information, see "LOGIN_PROCEDURE option [database]" [*ASA Database Administration Guide*, page 603].

♦ **dbsvc enhancements** The dbsvc command line utility for managing Windows services has been extended to list service name used to start and stop the service with the system **net start** and **net stop** commands, and to handle dependencies on other services and groups.

☞ For more information, see "The Service Creation utility" [*ASA Database Administration Guide*, page 519].

- **Source format preserved for stored procedures** The source format, including spaces and line breaks, is now stored in the database as a comment. This comment is used for procedure profiling.

Client/server connections

- **Improved buffer size negotiation** Buffer sizes can now be specified separately for both the client and the server.

  ☞ To use this feature, you must use version 8 software at both the client and the server. You do not need to upgrade the database.

- **Communication compression** A new type of communication compression can lead to improved performance if you are transferring data across networks with limited bandwidth, including some wireless networks, some modems, serial links and some WANs.

  For more information, see "Adjusting communication compression settings to improve performance" [*ASA Database Administration Guide*, page 102].

  ☞ To use this feature, you must use version 8 software at both the client and the server. You do not need to upgrade the database.

- **Enhanced dbping** The *dbping* utility has additional options to help diagnose connection problems. These include the ability to use ODBC to connect, and the ability to report connection, database, and server properties upon connection.

  ☞ For more information, see "The Ping utility" [*ASA Database Administration Guide*, page 514].

- **Suppress TDS debugging option** The SUPPRESS_TDS_DEBUGGING option controls whether TDS debugging information appears in the server window.

  ☞ For more information, see "SUPPRESS_TDS_DEBUGGING option [database]" [*ASA Database Administration Guide*, page 627].

- **PrefetchBuffer connection parameter** This connection parameter lets you specify the maximum amount of memory for storing prefetched rows.

  ☞ For more information, see "PrefetchBuffer connection parameter [PBUF]" [*ASA Database Administration Guide*, page 197].

- **PrefetchRows connection parameter** The PrefetchRows connection parameter lets you specify the maximum number of rows to prefetch when querying the database. In some circumstances, increasing the number of rows prefetched from the database server by the client can improve query performance.

☞ For more information, see "PrefetchRows connection parameter [PROWS]" [*ASA Database Administration Guide*, page 198].

- **Client can specify idle timeout** Each client can specify its own idle timeout using the IDLE connection parameter. Previously, all connections to a server used the same idle timeout which was specified by the `-ti` server command line option.

  ☞ For more information, see "Idle connection parameter [IDLE]" [*ASA Database Administration Guide*, page 192].

Java in the database

Java in the database includes the following new features:

- **Java 2 support** Java in the database can now use classes from Java 2 (JDK 1.2 and 1.3) and Java.

  ☞ For more information, see "Java-enabling a database" [*ASA Programming Guide*, page 87].

  To use this feature, you must upgrade the database using ALTER DATABASE or by using the *dbupgrad* command line utility and supplying the `-jdk` option.

- **JDBC 2.0** Java classes in the database can now use the JDBC 2.0 interface to access data.

  ☞ For more information, see "JDBC in the database features" [*ASA Programming Guide*, page 106].

  To use this feature, you must upgrade the database using ALTER DATABASE or by using the *dbupgrad* command line utility and supplying the `-jdk` option.

- **Diagnostic procedure** A new system procedure, **sa_java_loaded_classes**, lists all classes loaded by the Java virtual machine.

  ☞ For more information, see "sa_java_loaded_classes system procedure" [*ASA SQL Reference*, page 720].

  To use this feature, you must upgrade the database.

- **Security manager** You can use a built-in security manager or provide your own implementation to control access to security-sensitive Java features.

  ☞ For more information, see "Security management for Java" [*ASA Programming Guide*, page 96].

Documentation

Several new features have been added to the Adaptive Server Anywhere documentation set to help you find, access and use the information more quickly.

- **Re-organized books** There have been two major changes to the documentation set since the last release:
  - The *Replication and Synchronization Guide* has been split into two books, describing each of the two synchronization technologies separately. These new books are the *MobiLink Synchronization User's Guide* and the *SQL Remote User's Guide.*
  - The Adaptive Server Anywhere *User's Guide, Programming Interfaces Guide*, and *Reference Manual* have been replaced by a *Database Administration Guide*, a *SQL User's Guide*, a *SQL Reference Manual*, and a *Programming Guide.* The database error messages have been moved into their own book. The new organization makes each book a more manageable size in printed form.
- **New context-sensitive Help** All the user-interface tools, including Sybase Central, Interactive SQL, the Adaptive Server Anywhere debugger, and the Query Editor, share a common cross-platform context-sensitive help system, complete with links to the online books.
- **Enhanced online books** The HTML Help version of the online books includes a menu bar for quick access to SQL Anywhere Web links, tutorials, procedures, and more.

Miscellaneous

- **Connections persist across hibernation times** Connections from embedded SQL, ODBC or OLE DB clients now persist while a computer hibernates. Previously, TCP/IP connections between a client and a server on the same machine would be dropped when the machine was woken from hibernation if the machine hibernated for longer than the liveness or idle timeout time.
- **Viewing current license information** The dblic utility now accepts an argument that allows you to view current license information for a server executable without starting the server.

  ☞ For more information, see "The License utility" [*ASA Database Administration Guide*, page 499].
- **Viewing collation label and name for custom collations** The dbinfo command line utility now returns the collation label and name for custom

collations. As well, two new fields, collationnamebuffer and collationnamebufsize, have been added to the a_db_info structure in *dbtools.h*.

☞ For more information, see "Obtaining database information using the dbinfo command-line utility" [*ASA Database Administration Guide*, page 483] and "a_dbtools_info structure" [*ASA Programming Guide*, page 292].

- **sp_remote_tables system procedure** A new argument, **tabletype**, has been added to the sp_remote_tables stored procedure. This argument returns the remote table's type.

  ☞ For more information about the **tabletype** *argument*, see "sp_remote_tables system procedure" [*ASA SQL Reference*, page 749].

- **-ct command line option** Using the `-ct` command line option, you can turn character set translation on and off. Character set translation is now enabled by default, and to turn it off, you can specify `-ct-`. To turn character set translation on, use `-ct+`.

  ☞ For more information, see "-ct server option" [*ASA Database Administration Guide*, page 137].

- **Obtain remote table foreign key information** Two new stored procedures, sp_remote_exported_keys and sp_remote_imported_keys, allow you to obtain information about foreign keys and their corresponding primary keys for remote tables.

  ☞ For more information, see "sp_remote_exported_keys system procedure" [*ASA SQL Reference*, page 747] and "sp_remote_imported_keys system procedure" [*ASA SQL Reference*, page 748].

- **xp_sendmail** There are now extended stored procedures for sending email over SMTP as well as MAPI. For more information, see "xp_startsmtp system procedure" [*ASA SQL Reference*, page 755] and "xp_stopsmtp system procedure" [*ASA SQL Reference*, page 759].

  The xp_sendmail stored procedure now accepts messages of any length. The length of the long VARCHAR parameters for the procedure is limited to the amount of memory available on your system.

  ☞ For more information, see "xp_sendmail system procedure" [*ASA SQL Reference*, page 756].

- **Replication Server 12 feature for the log transfer manager** The **qualify_table_owner** parameter in the LTM configuration file provides support for the Replication Server 12 feature allowing the table names, owners, and column names in the primary databases to be different from the replication databases.

☞ For more information, see "The LTM configuration file" [*ASA Database Administration Guide*, page 503].

- **ASANYSH8 environment variable** A new environment variable, ASANYSH8, has been added. Interactive SQL, Sybase Central, the Console utility, and the debugger use this environment variable to located the shared components directory.

  ☞ For more information about the ASANYSH8 environment variable, see "ASANYSH8 environment variable" [*ASA Database Administration Guide*, page 245].

MobiLink new features

The following is a list of changes and additions to the software introduced in version 8.0.

Flexibility

- **Java synchronization logic** Synchronization scripts can now be implemented in Java instead of or in addition to the SQL language. These scripts are run in an external JRE using the MobiLink Java environment.

  ☞ For more information see "Writing Synchronization Scripts in Java" [*MobiLink Synchronization User's Guide*, page 227].

- **Synchronization using publications** All the data in a MobiLink client no longer needs to be synchronized at the same time. Rather, data can be organized into publications and each publication synchronized independently. A new syntax for publications and synchronization subscriptions is provided, that is simpler and more precise than the previous syntax.

  ☞ For more information see "Adaptive Server Anywhere Clients" [*MobiLink Synchronization User's Guide*, page 167].

- **Configuring Web servers to handle MobiLink synchronization** You can now carry out HTTP synchronization with the MobiLink synchronization server behind a firewall. A Web server plug-in for popular Web servers allows you to carry out HTTP synchronization through Web servers.

  ☞ For more information, see "Synchronizing Through a Web Server" [*MobiLink Synchronization User's Guide*, page 313].

- **ActiveSync support for Windows CE clients** Both Adaptive Server Anywhere and UltraLite Windows CE MobiLink clients can use the Windows CE ActiveSync synchronization software.

  ☞ For more information, see "Using ActiveSync synchronization" [*MobiLink Synchronization User's Guide*, page 189].

- **Enhanced client command line functionality** You can specify extended options in both `CREATE/ALTER SYNCHRONIZATION SUBSCRIPTION` statements and on the command line.

  ☞ For more information see "MobiLink synchronization client" [*MobiLink Synchronization Reference*, page 36].

- **Extended options can be stored in the database** Using the `CREATE/ALTER SYNCHRONIZATION SUBSCRIPTION` statements it is possible to store extended options and connection parameters in the database and associate them with subscriptions, users or publications. *Dbmlsync* now reads this information from the database.

  ☞ For more information see "MobiLink synchronization client" [*MobiLink Synchronization Reference*, page 36].

Performance

- **Statement-based uploads** MobiLink now allows statement-based uploads that are not only more intuitive than cursor-based uploads, but also significantly faster. Statement-based uploads employ the **upload_insert**, **upload_delete**, **upload_update**, **upload_new_row_insert**, and **upload_old_row_insert** events. **upload_fetch** script is used for conflict resolution.

  ☞ For more information see "Writing scripts to upload rows" [*MobiLink Synchronization User's Guide*, page 54].

- **Multi-processor administration** MobiLink has a new option for setting the maximum number of processors to use. The `-zt` option provides for greater control of the resources used by the MobiLink synchronization server. It can also help to discover and/or work around an ODBC driver with multi-processor issues.

  ☞ For more information see "-zt option" [*MobiLink Synchronization Reference*, page 31].

- **Optional download acknowledgement** The MobiLink synchronization client can now synchronize without a download acknowledgement, so that the MobiLink synchronization server worker thread does not need to wait for the client to apply the download, freeing up the worker thread sooner for its next synchronization. Download acknowledgement is now an option. Eliminating the download acknowledgement can improve throughput, particularly for slower clients. Note that without a download acknowledgement, the consolidated side will not know that the download succeeded until the next synchronization.

  ☞ For more information, see "-e extended options" [*MobiLink Synchronization Reference*, page 44].

- **Buffered download stream** The MobiLink synchronization server now buffers the download stream in a download cache. Since acknowledgement is not required from the client to commit the download transaction, the buffered download stream is sent to the client after the commit. The download transaction is no longer potentially held up by network delays.

  ☞ For more information, see "-d option" [*MobiLink Synchronization Reference,* page 11].

  The download stream can also be buffered at Adaptive Server Anywhere clients. The size of the buffer available can be set using the *dbmlsync* DownloadBufferSize extended option.

  ☞ For more information, see "-e extended options" [*MobiLink Synchronization Reference,* page 44].

- **Bulk loading of connection and table scripts** The first connection or table script requested for a specific table to version_id pairing will cause a bulk load of all the scripts into the cache. The result is improved performance by getting all the scripts in bulk rather than individually.

- **MobiLink synchronization server shutdown enhancements** You can tell *dbmlstop* to wait until the MobiLink synchronization server is completely shutdown before proceeding. You can also use *dbmlstop* to stop a specific MobiLink synchronization server by name.

  ☞ For more information see "MobiLink stop utility" [*MobiLink Synchronization Reference,* page 303].

- **Connection timeout** MobiLink database connections that are unused for a specified amount of time are now disconnected automatically by the server. The timeout can be set using the `-ct` (connection timeout) command line option.

  ☞ For more information see "-ct option" [*MobiLink Synchronization Reference,* page 11].

- **Maximum number of concurrent uploaders option** The `-wu` command line option can set the maximum number of worker threads allowed to upload concurrently, resulting in, for some deployments, increased throughput.

  ☞ For more information see "-wu option" [*MobiLink Synchronization Reference,* page 23].

Security

- **MobiLink user authentication** A password-based system for user authentication adds additional security to your MobiLink installation. Now, using `-zu,` you can allow automatic addition of users when the

authenticate_user script is undefined. This allows for user schema information to be used as MobiLink authentication.

☞ For more information, see "Authenticating MobiLink Users" [*MobiLink Synchronization User's Guide,* page 103].

- **MobiLink user administration** The dbmluser utility has been extended to allow users to be deleted from the system as well as added. Other refinements have been made to this utility. The dbmluser command line options -pf, -pp, and -pu have been deprecated and replaced with -f, -p, and -u respectively.

  ☞ For more information, see "MobiLink user authentication utility" [*MobiLink Synchronization Reference,* page 308].

Enhanced reporting

- **Statistical scripts** MobiLink now has scripts for tracking synchronization statistics. Once gathered, these synchronization statistics may be used for monitoring the performance of your synchronizations.

  ☞ For more information, see "synchronization_statistics connection event" [*MobiLink Synchronization Reference,* page 202], "synchronization_statistics table event" [*MobiLink Synchronization Reference,* page 205], "upload_statistics connection event" [*MobiLink Synchronization Reference,* page 224], and "upload_statistics table event" [*MobiLink Synchronization Reference,* page 227].

- **Detailed network error information** The MobiLink synchronization server and client now display detailed error information along with error codes to help you better resolve any errors as they arise. You will see the network layer reporting the error, the network operation being performed, the error itself and a system-specific error code.

- **Remote Adaptive Server Anywhere output log sent to MobiLink synchronization server on error** Troubleshooting synchronization problems is simplest when both the remote log and the MobiLink synchronization server log are available for inspection. This new feature sends the ASA remote's output log up to the MobiLink synchronization server when a client-side error occurs.

  ☞ For more information see "-e option" [*MobiLink Synchronization Reference,* page 12].

- **Log messages identify the worker thread** Messages displayed to the MobiLink synchronization server log now indicate the worker thread that logged the message. This makes it possible to distinguish messages that are due to the same user attempting to synchronize concurrently. It also helps distinguish messages when the same user synchronizes twice without delay.

- **Verbose logging** You can use additional modifiers on the MobiLink synchronization server `-v` command line option to configure MobiLink synchronization server logging.

  ☞ For more information see "-v option" [*MobiLink Synchronization Reference,* page 21].

- **Ignored rows are reported to clients** If the MobiLink synchronization server ignores any uploaded rows because of absent scripts, a messages is returned to the client. The message is displayed as a warning by Adaptive Server Anywhere clients, and in the ignored_rows synchronization parameter in UltraLite clients.

  ☞ For more information, see "Ignored Rows synchronization parameter" [*UltraLite Database User's Guide,* page 165].

Ease of use

- **Last download timestamp** The last download timestamp is written to the MobiLink client database automatically.

- **Automatic synchronization script generation** MobiLink can be instructed to generate scripts suitable for snapshot synchronization. The `-za` option controls creation and activation of these scripts.

  ☞ For more information see "Generating scripts automatically" [*MobiLink Synchronization User's Guide,* page 40]

- **Example synchronization script generation** MobiLink can be instructed to generate example synchronization scripts. The `-ze` command line option is used to control whether example scripts are to be generated.

  ☞ For more information, see "Generating example scripts" [*MobiLink Synchronization User's Guide,* page 41]

Adaptability

- **Support for popular RDBMSs** As consolidated databases, MobiLink now supports Oracle 8i and 9i, Microsoft SQL Server 7, Microsoft SQL Server 2000, IBM's DB2 and more.

  ☞ For more information, see "ODBC drivers supported by MobiLink" [*MobiLink Synchronization Reference,* page 336].

- **Liveness detection in TCP/IP streams** The TCP/IP-based streams that are used during MobiLink synchronization now accept a new parameter, both on the client and server side, called **keep_alive**, that enables liveness checking.

  ☞ For more information, see "-x option" [*MobiLink Synchronization Reference,* page 24].

UltraLite new features

UltraLite 8.0 introduces the following new features:

Security

- **User authentication** In previous releases, UltraLite databases had no user authentication mechanism to govern access. In this release, a built-in user authentication mechanism is provided. Unlike user IDs for most relational database-management systems, the UltraLite user IDs do not imply any ownership of tables and other database objects.

  ☞ For more information, see "User authentication" [*UltraLite Database User's Guide*, page 38].

- **Database encryption** You can improve the security of your data by encrypting your database. Two methods are supplied.
  - **Strong encryption** The database can be encrypted using a strong encryption algorithm for maximum security. There is a performance penalty to pay for this security. The encryption is key-based and uses the AES 128-bit algorithm.
  - **Database obfuscation** You can improve the security of your data by obfuscating the database. Without obfuscation, the data in the database is viewable using a tool such as a hex editor. Obfuscation prevents casual attempts at viewing data but does not offer the watertight protection of strong encryption. Obfuscation does not have the performance penalty that strong encryption carries.

    ☞ For more information, see "Encrypting UltraLite databases" [*UltraLite Database User's Guide*, page 36].

- **Secure synchronization for UltraLite Java applications** Secure synchronization using Certicom transport-layer security was previously available only from C/C++ UltraLite applications. It is now available from UltraLite Java applications.

  ☞ For more information, see "Using transport-layer security" [*UltraLite Static Java User's Guide*, page 48].

Synchronization

- **ActiveSync synchronization** UltraLite applications on Windows CE devices can use ActiveSync to synchronize.

  ☞ For more information, see "Adding ActiveSync synchronization to your application" [*UltraLite Embedded SQL User's Guide*, page 96].

- **More flexible synchronization** Several new features have been added to enable more efficient and flexible selection of data to synchronize:
  - You can use publications to partition your data into different sets, which can be synchronized separately. This permits the efficient synchronization of time-sensitive data, perhaps over slow connection links, while other data can be synchronized at a more convenient time.

- Download-only synchronization permits you to add read-only tables to your UltraLite database, and to synchronize them efficiently using a download-only synchronization.

  ☞ For more information, see "Including read-only tables in an UltraLite database" [*UltraLite Database User's Guide*, page 158].

- You can mark a table to be synchronized each time, whether or not the data in the table has changed. This feature allows you to maintain user-configurable information on the UltraLite client that controls synchronization.

  ☞ For more information, see "Using client-specific data to control synchronization" [*UltraLite Database User's Guide*, page 159].

♦ **Global autoincrement default column values** This feature provides a straightforward way of maintaining primary key uniqueness in a synchronizing database.

☞ For more information, see "Declaring default global autoincrement columns" [*UltraLite Database User's Guide*, page 151].

♦ **Additional control for UltraLite generator** New command line options have been added for the ulgen and sqlpp executables:

- **Script version** You can associate a script version with generated synchronization scripts.
- **Log query execution plans** The query execution plans for generated queries can be exported and displayed in Interactive SQL.

  ☞ For more information, see "The UltraLite generator" [*UltraLite Database User's Guide*, page 96], and "The SQL preprocessor" [*UltraLite Database User's Guide*, page 92].

♦ **Error reporting** The **stream_error** field on the **ul_synch_info** structure can be used to determine the cause of synchronization errors.

☞ For more information, see "Stream Error synchronization parameter" [*UltraLite Database User's Guide*, page 172].

Database management

♦ **Re-use of existing databases** In previous releases of UltraLite, any change to a database application required a rebuild and synchronization of the database. With this release, you can continue to use an UltraLite database with a new version of your application as long as the database schema does not change. Changes to queries do not of themselves require a new database, unless they reference new columns and so change the schema of the generated database.

♦ **Database defragmentation** The UltraLite store is designed to efficiently reuse free space, so that explicit defragmentation is not

required under normal circumstances. For applications with extremely strict space requirements, an explicit defragmentation function is provided.

☞ For more information, see "Defragmenting UltraLite databases" [*UltraLite Embedded SQL User's Guide*, page 60].

- **Choice of page size** You can choose to use 2 kb page sizes as an alternative to the default 4 kb pages.

  ☞ For more information, see "UL_STORE_PARMS macro" [*UltraLite Database User's Guide*, page 216].

Development features

- **CodeWarrior 7 support** The UltraLite plugin for CodeWarrior now supports CodeWarrior version 7.
- **eMbedded Visual C++** Development using this tool is supported, and an eMbedded Visual C++ project is supplied for the CustDB sample application.
- **Palm OS 4.0 and file-based data storage** UltraLite now supports version 4.0 of the Palm Computing Platform. Beginning with Palm 4.0, a variety of secondary storage schemes is introduced. You can use a file-based UltraLite data store on an expansion card for a Palm 4.0 device.

  ☞ For more information, see "ULEnableFileDB function" [*UltraLite Embedded SQL User's Guide*, page 110].
- **Improved synchronization for Palm Computing Platform** A new and simplified synchronization mechanism for HotSync and ScoutSync synchronization on the Palm Computing Platform has several benefits over previous synchronization mechanisms:
  - Launch and exit times are fast.
  - No extra storage is required on the Palm device during synchronization.
  - The application can be synchronized several times without launching.
  - No stream parameter needs to be specified.

  The **ULPalmDBStream** and **ULConduitStream** functions are deprecated.

  ☞ For more information, see "Adding HotSync synchronization to Palm applications" [*UltraLite Embedded SQL User's Guide*, page 81].
- **Easier deployment on the Palm Computing Platform** You can deploy initial copies of the UltraLite database to your end users so that the first synchronization does not have to download an initial copy of the data for each user.

  ☞ For more information, see "Deploying Palm applications" [*UltraLite Embedded SQL User's Guide*, page 84].

- **Improved handling of Palm segments** When developing for the Palm Computing Platform, application code must be divided into segments of limited size.

  The segmentation method provided in earlier versions of the software allowed no user control over the segmentation of the UltraLite generated code, and tended to assign too many segments (which could degrade performance). A new mechanism generates fewer segments and provides customers with control over the assignment of segments.

  ☞ For more information, see "Enabling multi-segment code generation" [*UltraLite Embedded SQL User's Guide*, page 78].

- **LONG values in embedded SQL** You can use host variables for long values (between 32 kb and 64 kb) using the DECL_LONGVARCHAR and DECL_LONGBINARY.

- **Analyzer hooks in the reference database** The UltraLite generator now invokes stored procedures before and after the analysis process.

  ☞ For more information, see "The SQL preprocessor" [*UltraLite Database User's Guide*, page 92], and "The UltraLite generator" [*UltraLite Database User's Guide*, page 96].

- **Query plan information** The UltraLite generator can now output the access plan to be used for queries in UltraLite applications. Also, you can view the access plan that would be used for UltraLite from Interactive SQL.

  ☞ For more information, see "The UltraLite generator" [*UltraLite Database User's Guide*, page 96].

- **Script version control** You can specify the script version to be used for synchronization on the UltraLite generator command line.

  ☞ For more information, see "The UltraLite generator" [*UltraLite Database User's Guide*, page 96].

- **Additional SQL and API features** The following features are now available to UltraLite applications.
  - **@@identity supported** The @@identity global variable is now supported by UltraLite. This feature is useful in the context of global autoincrement default column values. In the C++ API, use the **ULConnection::GetLastIdentity()** method.

    ☞ For more information, see "GetLastIdentity method" [*UltraLite Static C++ User's Guide*, page 78].

  - **Number of rows in a table** From the C++ API programming interface you can determine the number of rows in a table using the

ULTable::GetRowCount() method. Embedded SQL users continue to use the SELECT COUNT(*) FROM *table-name* statement.

☞ For more information, see "GetRowCount method" [*UltraLite Static C++ User's Guide*, page 111].

- **Delete all rows in a table** From the C++ API programming interface you can delete all rows in a table using the **ULTable::DeleteAllRows()** method. Embedded SQL users continue to use the DELETE FROM *table-name* statement.

  ☞ For more information, see "DeleteAllRows method" [*UltraLite Static C++ User's Guide*, page 108].

- **Number of rows affected** From embedded SQL you can determine the number of rows affected by the last INSERT, UPDATE, or DELETE statement using the SQLCOUNT macro.

- **Number of rows to be uploaded** You can determine the number of rows that need to be synchronized.

  ☞ For more information, see "ULCountUploadRows function" [*UltraLite Embedded SQL User's Guide*, page 108], and "CountUploadRows method" [*UltraLite Static C++ User's Guide*, page 77].

- **Last download time** You can obtain the last download time of a publication from the UltraLite application.

  ☞ For more information, see "ULGetLastDownloadTime function" [*UltraLite Embedded SQL User's Guide*, page 115] and "GetLastDownloadTime method" [*UltraLite Static C++ User's Guide*, page 78].

- **Additional cursor operations** The **ULTable** class of the C++ API has additional methods (**FindFirst**, **FindNext**, **FindPrevious**, **FindLast**) to locate rows in a result set.

  ☞ For more information, see "ULTable class" [*UltraLite Static C++ User's Guide*, page 108].

- **Queries from DUMMY system table** Queries of the form SELECT ... FROM DUMMY are now supported.

- **Updating multiple tables** Cursors over multiple tables can now accept updates that modify more than one table.

- **Improved LONG data type handling for embedded SQL** The DECL_LONGVARCHAR and DECL_LONGBINARY host variable types can be used to send or retrieve data over 32 kb in a single operation.

  ☞ For more information, see "Data types in embedded SQL" [*UltraLite Embedded SQL User's Guide*, page 30].

SQL Remote new features

- **Event-hook procedures** A set of event-hook procedures have been added to enable customization of the replication process. By writing stored procedures with specified names, you can add customizations at several points in the actions the Message Agent takes during replication.

  ☞ For more information, see "SQL Remote event-hook procedures" [*SQL Remote User's Guide*, page 318].

Behavior changes in version 8

This section lists the behavior changes introduced in components of SQL Anywhere Studio version 8.

Adaptive Server Anywhere behavior changes

The following are behavior changes from previous versions of the software.

☞ For a list of newly deprecated and unsupported features, see "Deprecated and unsupported features" on page 111.

- **Java in the database separately licensable** As a consequence, the default behavior when creating a database is to exclude support for Java in the database.

  Java in the database is no longer needed in UltraLite reference databases, as the UltraLite generator has been changed to use an external Java VM.

  ☞ For more information, see "Welcome to SQL Anywhere Studio" [*Introducing SQL Anywhere Studio*, page 4].

- **Aggregate functions and outer references** Adaptive Server Anywhere version 8 follows new SQL/99 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 correct queries may now produce error messages, and result sets may change.

  ☞ For more information, see "Aggregate functions and outer references" [*ASA SQL User's Guide*, page 233].

- **User-supplied selectivity estimates** Adaptive Server Anywhere allows you to specify explicit selectivity estimates to guide the choice of access plan. These estimates were most useful as workarounds to performance problems where the software-selected access plan was poor. The new USER_ESTIMATES connection option controls whether the optimizer uses or ignores user-supplied selectivity estimates.

  If you have used these estimates as a workaround to performance problems, we recommend setting the USER_ESTIMATES option to OFF because an explicit estimate may become inaccurate and may force the optimizer to select poor plans. This version includes query processing enhancements such as internal join algorithms which provide a significant improvement in query performance.

  ☞ For more information about user-supplied selectivity estimates, see "USER_ESTIMATES option [database]" [*ASA Database Administration*

Guide, page 633] and “Explicit selectivity estimates” [*ASA SQL Reference,* page 30].

- **Row ordering** A side-effect of improvements to query processing for version 8.0 is that row ordering is less deterministic. In the absence of an ORDER BY clause, Adaptive Server Anywhere returns rows in whatever order is most efficient. This means the appearance of result sets may vary depending on when you last accessed the row and other factors. The only way to ensure that rows are returned in a particular order is to use ORDER BY.

  The LIST function is among those functions particularly affected by this change.

- **Access plan changes** The access plans selected by this release of Adaptive Server Anywhere are less likely to use indexes than previous releases of the software. Improvements to the efficiency of table scans, together with a more selective cost model used in comparing the cost of access plans, leads to a more accurate assessment of the usefulness of indexes than in previous versions of the software.

- **Cursor changes** A side effect of cursor enhancements is that the cursors in this version provide behavior closer to defined standards than before. This may produce cursor sensitivity changes for some cursors, as Adaptive Server Anywhere supplies behavior that better matches the expectations of ODBC and other interfaces. For example, embedded SQL SCROLL cursors now disallow prefetching, so that value changes are reflected in the cursor.

  This change may affect existing applications that check return codes only for SQL_SUCCESS and not SQL_SUCCESS_WITH_INFO. Applications that check for SQL_SUCCESS_WITH_INFO receive a warning if the cursor behavior is different from that requested. The warning is SQLCODE=121, SQLSTATE 01S02.

  Insensitive cursors are not updatable.

  ☞ For more information, see “Insensitive cursors” [*ASA Programming Guide,* page 35].

- **Stored procedure storage** Stored procedures are now stored as written. Adaptive Server Anywhere does create an internal representation of the procedure, which is used for profiling.

- **OPEN CURSOR on insert not supported** The ability to open a cursor on an INSERT statement has been dropped. Opening an updateable cursor on a SELECT statement gives the same capabilities in an industry-standard manner.

- **User-defined functions** User defined function parameters and return values are now cached. If a function is used several times within a SQL statement, the cached parameter values may result in the cached result being used, instead of the function being evaluated again. In previous releases, user-defined functions were re-evaluated each time they were needed. The new behavior provides better performance and more consistent results, but may change results compared to previous releases of the software.

- **NUMBER(*) function changes** The use of the NUMBER function has been restricted to avoid problematic behavior. NUMBER is intended for use in the select-list of a query, to provide a sequential row-numbering of the result set, and this use is still permitted.

  The NUMBER function may now give negative numbers in cases where it previously did not, such as if you carry out an absolute fetch with a value of -1 and then move backward through the cursor. The new behavior corresponds to the ISO/ANSI fetch offset.

  Use of the NUMBER function in many circumstances, such as a WHERE clause or a HAVING clause, now gives an error.

  ☞ For more information, see "NUMBER function [Miscellaneous]" [*ASA SQL Reference*, page 164].

- **Custom collation changes** Previously, the `-d` option in the Collation utility accepted three parameters; now it accepts only two parameters. The *cust-map-file* parameter is no longer accepted. The syntax for the Collation utility is

  ```
  dbcollat -d coll-defn-file custom-file
  ```

  As well, the script files *collsqmp.sql* and *custmap.sql* are no longer present and cannot be used for built-in or custom collations, respectively.

  For newly-created databases, the SYSCOLLATIONMAPPINGS table contains only one row with the collation mapping. For databases created with previous versions of Adaptive Server Anywhere, this table contains a row for each built-in collation.

  ☞ For more information, see "The Collation utility" [*ASA Database Administration Guide*, page 462] and "SYSCOLLATIONMAPPINGS system table" [*ASA SQL Reference*, page 625].

- **Trigger name changes** Trigger names no longer need to be unique across a database. They only need to be unique within the table to which they apply. The syntax of DROP TRIGGER and COMMENT ON TRIGGER has consequently changed so that you can only specify an owner if you also specify a table. This means that older scripts that

qualify triggers with only an owner will now result in a “Table not found” error.

- **Addresses changed in sample database** The addresses in the Adaptive Server Anywhere 9.0 Sample database are different from those in previous releases.

- **JAR file name for internal JDBC driver changed** The internal JDBC driver classes are now installed as a JAR file named ASAJRT instead of ASAJDBC.

- **RESTORE DATABASE statement permissions** A connection to the utility database is no longer required to execute a RESTORE DATABASE statement. The permissions required to execute a RESTORE DATABASE statement are controlled by the `-gu` command line option.

  ☞ For more information, see “RESTORE DATABASE statement” [*ASA SQL Reference*, page 525].

- **Return empty string as a NULL string for TDS connections** The TDS_EMPTY_STRING_IS_NULL option controls whether the server returns empty strings as a string containing one blank character or a NULL string for TDS connections.

  ☞ For more information, see “TDS_EMPTY_STRING_IS_NULL option [database]” [*ASA Database Administration Guide*, page 628].

- **COMMENT statement changed** Previously, the syntax for COMMENT ON INDEX included an optional owner name of the index. The index name can now optionally include the owner and table. The syntax for COMMENT ON INDEX is now

  COMMENT ON INDEX [[*owner*.]*table*.]*index-name* **IS** *comment*

  ☞ For more information, see “COMMENT statement” [*ASA SQL Reference*, page 282].

- **Character set translation enabled by default** In previous versions of Adaptive Server Anywhere, character set translation was turned off by default and you had to specify the `-ct` command line option to enable character set translation. Character set translation is now enabled by default, but can be disabled using the -ct- command line option.

  When the server determines that the connection’s character set differs from the database’s character set, the server applies character set translation to all the character strings sent to and from the server for that connection.

The server disables character set translation for a connection when it determines that the database and the connection have equivalent character sets.

In most cases, character set translation should be enabled. One possible change in behavior occurs when binary data is inserted into a database and is fetched as character data, or vice versa. In this case, the data may not be returned exactly as it was entered because the server applies character set translation only to character data. To avoid this problem, applications should not send or fetch character data using a binary type.

☞ For more information, see "-ct server option" [*ASA Database Administration Guide,* page 137] and "Turning off character set translation on a database server" [*ASA Database Administration Guide,* page 329].

- **CONVERT, TIMESTAMP_FORMAT and DATE_FORMAT** When using the TIMESTAMP_FORMAT or DATE_FORMAT options, if you specify a character symbol in mixed case (such as `Mmm`), Adaptive Server Anywhere now chooses the case that is appropriate for the language that is being used. In addition, the CONVERT function now converts character dates into the case that is appropriate to the language that is being used. For example, in English the appropriate case is `May`, while in French it is `mai`.

  ☞ For more information, see "DATE_FORMAT option [compatibility]" [*ASA Database Administration Guide,* page 587], "TIMESTAMP_FORMAT option [compatibility]" [*ASA Database Administration Guide,* page 629], and "CONVERT function [Data type conversion]" [*ASA SQL Reference,* page 107].

- **Change to three-valued Boolean logic** Two-valued Boolean logic applies only to cases of *expr* = NULL, where *expr* refers to a base column or an expression over a base column. Otherwise, three-valued logic applies. The ANSINULL option now affects only this specific case in the query's WHERE clause.

- **Sybase Central and Interactive SQL accept COMMLINKS connection parameter** In previous versions of Adaptive Server Anywhere, Sybase Central and Interactive SQL (the dbisql command line utility) ignored the COMMLINKS connection parameter. Sybase Central and Interactive SQL now accept this parameter.

  As a result of this change, some connection strings may behave differently than in previous versions of Adaptive Server Anywhere. Specifically, if you do not supply `COMMLINKS=tcpip`, Interactive SQL and Sybase Central do not look for servers on the network.

  ☞ For more information, see "CommLinks connection parameter [LINKS]" [*ASA Database Administration Guide,* page 179].

- **Clients ignore SQLLOCALE environment variable** Clients can use the CharSet connection parameter to specify the character set to be used on a connection. In previous versions of Adaptive Server Anywhere, the CHARSET parameter of the SQLLOCALE environment variable was used to change the client's default character set if the CharSet connection parameter was not supplied. Clients now ignore the SQLLOCALE environment variable.

- Unsupported character sets cause connection failure Clients can use the CharSet connection parameter to specify the character set to be used on a connection. However, if the server does not support the requested character set, the connection fails. When a client requested an unsupported character set in previous versions of Adaptive Server Anywhere, the connection succeeded with a warning. If the client does not specify a character set, but the client's local character set is unsupported by the server, the connection succeeds, but with a warning that the character set is not supported.

  This behavior occurs in version 8 clients connecting to version 6.x, version 7.x, and version 8 database servers.

- **Default packet size change** The default packets size for client/server communications has been changed from 1024 bytes to 1460 bytes.

  ☞ For more information on packet size, see "CommBufferSize connection parameter [CBSIZE]" [*ASA Database Administration Guide,* page 178], and "-p server option" [*ASA Database Administration Guide,* page 154].

- **dbdsn utility manages Adaptive Server Anywhere data sources only** The dbdsn command line utility for managing Adaptive Server Anywhere ODBC data sources is now explicitly restricted to Adaptive Server Anywhere data sources only.

- **LOGIN_PROCEDURE option requires DBA authority** The LOGIN_PROCEDURE option can only be set by a user with DBA authority. In previous versions of Adaptive Server Anywhere, DBA authority was not required to set this option. A user with DBA authority can change the setting of this option for other users, but users without DBA authority cannot change their own setting of this option. As a result of this change, the DBA can ensure that a common procedure, if necessary, is executed when a user connects.

  ☞ For more information, see "LOGIN_PROCEDURE option [database]" [*ASA Database Administration Guide,* page 603].

- **ESTIMATE_SOURCE returns new values** The ESTIMATE_SOURCE function returns more detailed values than previously.

☞ For more information, see "ESTIMATE_SOURCE function [Miscellaneous]" [*ASA SQL Reference*, page 125].

Deprecated and unsupported features

This list includes features that are no longer supported and that impact existing applications.

- **NetWare 4.10 unsupported** Novell NetWare version 4.11 and later is still supported. Versions 3.x and 4.10 are unsupported.
- **NetBios unsupported** The NetBios port is no longer supported. If you use NetBios, you should switch to TCP/IP or SPX.
- **IPX unsupported** The IPX port is no longer supported. If you use IPX, you should switch to SPX or TCP/IP.
- **Deprecated collations** The following collations are no longer supported. Where indicated, they have been superceded by different collations:

Deprecated	Superceded by
437	437LATIN1
850	850LATIN1
852	852LATIN2
860	860LATIN1
863	863LATIN1
865	865NOR
SJIS	932JPN
SJIS2	932JPN
WIN_LATIN1	1252LATIN1
WIN_LATIN5	1254TRK
Internal	850LATIN1
437EBCDIC	

- **-e option no longer supported** The `-e` command line option and the `-e` option in the Data Source Utility, used to encrypt client/server communications, are no longer supported. The `-ec` option has replaced them. On the server, `-ec` **simple** uses the same encryption algorithm as `-e` in previous versions of Adaptive Server Anywhere.

- **NONE parameter deprecated** The NONE parameter for the ISQL_PLAN option is no longer supported. The query optimization plan now appears on the Plan tab in the Results pane. When you click the Plan tab, a plan always appears. Previously, the plan appeared in the Messages pane.
- **WITH HASH SIZE n clause deprecated** The WITH HASH SIZE clause is no longer supported.
- **MAX_WORK_TABLE_HASH_SIZE option deprecated** The MAX_WORK_TABLE_HASH_ SIZE option is no longer supported.
- **MAX_HASH_SIZE option deprecated** The MAX_HASH_SIZE option is no longer supported.
- **SATMP environment variable deprecated** The SATMP environment variable used by UNIX versions of Adaptive Server Anywhere to indicate a directory where temporary files are kept is no longer supported. On UNIX, the ASTMP environment variable can be used to indicate where temporary files are kept.

  ☞ For more information, see "ASTMP environment variable" [*ASA Database Administration Guide*, page 246].
- **dbtran -id option removed** The `-id` command line option on the dbtran command line utility is not present in this software.

MobiLink behavior changes

- **MobiLink Adaptive Server Anywhere client setup** MobiLink clients are now configured using publications and synchronization subscriptions, rather than synchronization definitions.

  ☞ For more information, see "Adaptive Server Anywhere Clients" [*MobiLink Synchronization User's Guide*, page 167].
- **Last download timestamp parameter changes scripts** The addition of a new parameter to many scripts makes timestamp-based synchronization easier to implement. The new parameter breaks existing scripts, as it is supplied as the first parameter to many scripts. To continue using existing scripts, change the behavior to supply the last download timestamp as the final parameter by supplying the `-zd` MobiLink synchronization server command line option.
- **MobiLink shutdown** Previously, *dbmlstop* commands from a remote connection could cause the MobiLink synchronization server to shut down. Now only *dbmlstop* requests from the same machine as the

MobiLink synchronization server will cause the MobiLink synchronization server to shut down. The `-zs` option, which would allow *dbmlstop* to stop the server, is no longer required.

- **Default setting for liveness detection in TCP/IP streams has changed** The default setting for keep_alive is now 1 (ON).
- **MobiLink can hide dbmluser information** The amount of information displayed when the dbmluser command line utility is used, such as timestamp, copyright, and other MobiLink synchronization server messages no longer appear by default.
- **MobiLink user authentication** You must use the `-zu+` option on the MobiLink synchronization server command if you do not use MobiLink user authentication.
- **Default log extension now .mls** Each file is now named DDMMYYNN.MLS where DD is the day of the month, MM is the month number, and YY is the year in the century. NN is a sequence number that starts at 1 with the first file.

UltraLite behavior changes

- **Required code change for Palm applications** Your code must specify whether to use standard record-based database storage or to use the file-based expansion card storage for Palm Computing Platform version 4.x. You must add a single function call before calling ULPalmLaunch (embedded SQL) or ULData.PalmLaunch (C++ API). The function calls are as follows:

  ```
  ULEnablePalmRecordDB( &sqlca );
  ```

  or

  ```
  ULEnableFileDB( & sqlca );
  ```

  Supply **ULEnablePalmRecordDB** if you use record-based storage, and **ULEnableFileDB** for file-based storage. If the device does not support file-based storage, ULPalmLaunch sets SQLCODE -82.

The following platforms are no longer supported by UltraLite:

- **DOS target platform** DOS is no longer a supported platform.
- **Metrowerks CodeWarrior 5 development platform** CodeWarrior 6 is now required for UltraLite development.
- **Palm 2.x no longer supported** UltraLite no longer supports development for Palm OS 2.x devices such as the PalmPilot Professional. Version 3.0 or later is required.

- **ULPalmDBStream and ULConduitStream deprecated** The new synchronization stream for HotSync or ScoutSync synchronization on the Palm Computing Platform means that the **ULPalmDBStream** and **ULConduitStream** functions are obsolete. They are still accepted, but have no effect.

- **UltraLite generator uses external Java VM** The UltraLite Analyzer now runs external to the database engine, and so can be used against reference databases even if they are not Java-enabled.

- **UltraLite JDBC package name changed** The package name for the UltraLite JDBC functions has been changed from **com.sybase.asa.ultralite.jdbc** to **ianywhere.ultralite.jdbc**. This requires a change to the `import` statements used for UltraLite applications.

- **All changes must be committed before download synchronization** Download-only synchronization is no longer an exception to the rule that all changes must be committed before synchronization.

- **dbmlsync StreamCompression extended option deprecated** This option is now ignored.

☞ You should also check Adaptive Server Anywhere behavior changes, as some may have an impact on your application. For more information, see "Adaptive Server Anywhere behavior changes" on page 105.

CHAPTER 6

What's New in Version 7.0.3

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in SQL Anywhere Studio version 7.0.3.

Contents

New features

This section introduces the new features in Adaptive Server Anywhere version 7.0.3. It provides a listing of major and minor new features, with cross references to locations where each feature is discussed in detail.

- **Database properties for blank padding and case sensitivity** You can now use two new properties to determine if your database uses blank padding when comparing strings (**BlankPadding**) or if your database is case sensitive (**CaseSensitive**).

  ☞ For more information see "Database-level properties" [*ASA Database Administration Guide*, page 664].

- **Server property for C2 security mode** You can now use the new C2 server property to determine whether the database server was started using the `-sc` option. The `-sc` option is intended for use in a C2-certified environment.

  ☞ For more information see "Server-level properties" [*ASA Database Administration Guide*, page 657].

- **Login procedure allows connections to be blocked** The **login_procedure** option allows a stored procedure to be called for each new connection. This procedure can now be used to disallow database connections.

  ☞ For more information see "LOGIN_PROCEDURE option [database]" [*ASA Database Administration Guide*, page 603].

- **FileDSN now supported on UNIX** The FileDSN connection parameter for ODBC data sources is now supported on UNIX.

Behavior changes

The following is a behavior changes from previous versions of the software.

- **Load table semantics changed** The LOAD TABLE command now has improved semantics if a column list is specified. A column list must specify each of the columns that exist in the file in the order in which they appear. Column names that do not appear in the list are set to NULL, zero, an empty string, or a default value, depending on the column nullability, data type, and default behavior.

  Columns that exist in the input file but which are to be ignored by LOAD TABLE can be specified using the column name **filler()**.

  ☞ For more information see "LOAD TABLE statement" [*ASA SQL Reference*, page 486].

CHAPTER 7

What's New in Version 7.0.2

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in Adaptive Server Anywhere version 7.0.2.

Contents

New features in version 7.0.2

This section lists the new features introduced in components of SQL Anywhere Studio version 7.0.2.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 7.0.2. It provides an exhaustive listing of major and minor new features, with cross references to locations where each feature is discussed in detail.

- **Dynamic cache sizing** On Windows 95/98, the size of the database server cache increases and decreases depending on the load on the database server and the other demands on system memory. This feature removes the need for choosing an explicit cache size in many circumstances, and can also boost performance.

  ☞ For more information, see "Dynamic cache sizing (Windows NT/2000/XP, Windows 95/98/Me)" [*ASA SQL User's Guide*, page 178].

- **Viewing current license information** The License [dblic] utility now accepts an argument that allows you to view current license information for a server executable without starting the server.

  ☞ For more information, see "The License utility" [*ASA Database Administration Guide*, page 499].

- **Additional collations** There are three new collations available: one to support Russian and Ukrainian (1251CYR, ANSI Code Page 1251), one to support Turkish (1254TRK, ANSI Code Page 1254) and one to support specialty requirements for some German users (1252DEU, ANSI Code Page 1252).

  The 1252LATIN1 collation continues to be the recommended German collation. 1252DEU is a specialty collation only, and should not be used without understanding its sorting and comparison properties.

  ☞ For a complete list of available collations, see "Choosing collations" [*ASA Database Administration Guide*, page 307].

- **Interactive SQL return codes** When run from the command prompt, Interactive SQL now sets a program exit code indicating the success or otherwise of the operations in the session.

  ☞ For more information, see "The Interactive SQL utility" [*ASA Database Administration Guide*, page 492].

- **DELETE_OLD_LOGS enhancement** The DELETE_OLD_LOGS database option is used in management of offline transaction logs in a replication environment. The option has been enhanced to permit more control over when processed transaction logs are deleted.

  ☞ For more information, see "DELETE_OLD_LOGS option [replication]" [*ASA Database Administration Guide*, page 590].

- **Connection troubleshooting and enhancements** The following changes have been made to permit better troubleshooting and tuning of client/server communications:
  - The APPINFO string is now added to the client debug log file.

    ☞ For more information, see "AppInfo connection parameter [APP]" [*ASA Database Administration Guide*, page 175].
  - Two new connection parameters can be used to tune prefetching of rows.

    ☞ For more information, see "PrefetchRows connection parameter [PROWS]" [*ASA Database Administration Guide*, page 198] and "PrefetchBuffer connection parameter [PBUF]" [*ASA Database Administration Guide*, page 197].
  - The **ConnectionName** connection parameter value was previously overridden for ODBC clients. You can now use the **ConnectionName** parameter from ODBC clients.

    ☞ For a list of connection parameters, see "Connection parameters" [*ASA Database Administration Guide*, page 70].

- **Language utility** The Language [dblang] utility allows you to report and modify the language registry for the Adaptive Server Anywhere messages and Sybase Central interface elements.

  ☞ For more information, see "The Language utility" [*ASA Database Administration Guide*, page 497].

- **dbspawn enhancement** The Spawn [dbspawn] utility optionally reports the operating system process ID of the database server.

  ☞ For more information, see "The Spawn utility" [*ASA Database Administration Guide*, page 523].

- **First day of week option** The default first day of week is now 7, which is Sunday. This value affects the result of DATEPART when obtaining a weekday value. You can change the first day of week using the DATEFIRST option in the Transact-SQL SET statement. You can set it permanently using SET OPTION FIRST_DAY_OF_WEEK=*n*.

  ☞ For more information, see "SET statement [T-SQL]" [*ASA SQL Reference*, page 550], or "FIRST_DAY_OF_WEEK option [database]" [*ASA Database Administration Guide*, page 593].

- **New migration tool** You can migrate (import) remote Oracle, DB2, Microsoft SQL Server, Sybase Adaptive Server Enterprise, Sybase Adaptive Server Anywhere and Microsoft Access databases into Adaptive Server Anywhere using the new sa_migrate set of stored procedures.

  ☞ For more information, see "Migrating databases to Adaptive Server Anywhere" [*ASA SQL User's Guide*, page 548].

- **Event handlers** Adaptive Server Anywhere can now determine how many instances of a particular event handler is executing at any given time. This is useful for limiting event handlers to only one instance at a time.

  ☞ For more information, see "EVENT_PARAMETER function [System]" [*ASA SQL Reference*, page 128].

- **New connection property** A new connection property helps distinguish between internal connections used to run event handlers.

  ☞ For more information, see "CONNECTION_PROPERTY function [System]" [*ASA SQL Reference*, page 106].

- **Dbdsn supports user and system specifiers** The Data Source [dbdsn] utility now supports the `u` (user) and `s` (system) options.

  ☞ For more information, see "The Data Source utility" [*ASA Database Administration Guide*, page 472].

- **Support for comments in @filename files** Adaptive Server Anywhere now supports comment lines in @filename files.

  ☞ For more information, see "@filename server option" [*ASA Database Administration Guide*, page 131].

- **Truncate timestamp option** To allow for greater compatibility with non-Adaptive Server Anywhere databases, you can now truncate timestamp values.

  ☞ For more information, see "TRUNCATE_TIMESTAMP_VALUES option [database]" [*ASA Database Administration Guide*, page 631].

- **Obtaining licensing information** Engine properties have been added to help you obtain accurate licensing information about your copy of Adaptive Server Anywhere.

  ☞ For more information, see "Server-level properties" [*ASA Database Administration Guide*, page 657].

- **Resetting the autoincrement value** The sa_reset_identity system procedure allows you to reset an autoincrement value for the next row.

☞ For more information, see "sa_reset_identity system procedure" [*ASA SQL Reference*, page 738].

MobiLink new features

- **Maximum number of threads applying upload streams** To reduce database contention, the `-wu` command line option can now be used to set the maximum number of worker threads allowed to upload concurrently. The upload requests are processed in first-come, first-serve order.
- For more information, see "MobiLink Synchronization Server Options" [*MobiLink Synchronization Reference*, page 3].

Behavior changes in version 7.0.2

This section lists the behavior changes introduced in components of SQL Anywhere Studio version 8.

Adaptive Server Anywhere behavior changes

The following are behavior changes from previous versions of the software.

- **Aliases must be defined before first reference** In earlier versions of SQL Anywhere, it was possible to refer to an alias in a SELECT list before the definition of the alias had appeared. An attempt to do so will now generate the error “Definition for alias alias-name must appear before its first reference”. To prevent this error, the SELECT list must be re-ordered so that the alias definition appears before its first use.

CHAPTER 8

What's New in Version 7.0.1

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in Adaptive Server Anywhere version 7.0.1.

Contents

New features in version 7.0.1

This section lists the new features introduced in components of SQL Anywhere Studio version 7.0.1.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 7.0.1. It provides an exhaustive listing of major and minor new features, with cross references to locations where each feature is discussed in detail.

- **New Service Creation utility** Running a database server as a service under NT allows databases to keep running without tying up the machine on which they are running. Previously, you added services using the Create a New Service wizard from Sybase Central. In Version 7 of Adaptive Server Anywhere, you can now also manage ASA services on Windows NT using a the Service Creation [dbsvc] utility. A variety of options allow you to add or delete a service, list all ASA services, or display the details of a particular service. This feature is particularly useful for embedding the creation of a service in installations.

  ☞ For more information about the Service Creation utility, see "The Service Creation utility" [*ASA Database Administration Guide*, page 519].

- **Windows CE 3.0 support** In addition to Windows CE 2.11, Adaptive Server Anywhere now supports Windows CE 3.0 on the following processors:
  - MIPS
  - Hitachi SH3.
  - ARM.

  Support for Windows CE 2.11 is provided on a wider range of platforms. For more information, see "Windows and NetWare operating systems" [*Introducing SQL Anywhere Studio*, page 125].

  With support for Windows CE 3.0, the OLE DB driver on CE works without installing any additional software.

- **Embedded SQL enhancements** A new function, **db_locate_servers**, provides a programmatic way of locating Adaptive Server Anywhere database servers listening on TCP/IP.

  ☞ For more information, see "db_locate_servers function" [*ASA Programming Guide*, page 213].

  A new callback function, DB_CALLBACK_CONN_DROPPED, provides a way of adding logic when the database server is about to drop a connection.

☞ For more information, see "db_register_a_callback function" [*ASA Programming Guide*, page 214].

- **Connection-level Debug and Logfile connection parameters** The DBG and LOG client-side connection parameters are now connection-specific, so you can configure debug information separately for different connections, even from the same application.

  ☞ For more information, see "Logfile connection parameter [LOG]" [*ASA Database Administration Guide*, page 194].

- **New database property** The **LTMGeneration** property has been added for users of the Replication Agent, or LTM. This property is primarily for use in technical support cases.

  ☞ For more information, see "Database-level properties" [*ASA Database Administration Guide*, page 664].

- **New deployment feature** Users of InstallShield Professional 5.5 and up can use the new SQL Anywhere Studio InstallShield Template Projects to deploy their own application. This feature allows you to quickly build your application's installation using the entire template project, or just the parts that apply to your install.

  For more information, see "Using InstallShield for deployment" [*ASA Programming Guide*, page 474].

- **New backup statement feature** When using the Backup statement, you can specify an empty string as a directory to rename or truncate the log without copying it first. This is particularly useful in a replication environment where space is a concern. You can use this feature with an event handler on transaction log size to rename the log when it reaches a given size, and with the DELETE_OLD_LOGS option to delete the log when it is no longer needed.

  ☞ For more information, see the "BACKUP statement" [*ASA SQL Reference*, page 263].

MobiLink new features

Following is a list of changes and additions to the software introduced in version 7.0.1.

- **User authentication** A password-based system for user authentication adds additional security to your MobiLink installation.

  ☞ For more information, see "Authenticating MobiLink Users" [*MobiLink Synchronization User's Guide*, page 103].

- **Extensive documentation of transport-layer security** The transport-layer security documentation has been extended to describe a variety of architectures possible with this powerful security mechanism.

  ☞ For more information, see "Transport-Layer Security" [*MobiLink Synchronization User's Guide*, page 337].

- **Customizing synchronization and synchronization-related processes** The Adaptive Server Anywhere synchronization client *dbmlsync* now supports a set of events. You can add stored procedures to your Adaptive Server Anywhere database to program event-based actions. This adds flexibility to the synchronization process, including the ability to schedule synchronization.

  ☞ For more information, see "Customizing the client synchronization process" [*MobiLink Synchronization User's Guide*, page 194].

- **Synchronization optimizations** You can optimize the following aspects of the synchronization process.
  - UltraLite client applications can specify that a synchronization includes only uploads, and that no download phase should be attempted.

    This option lessens the overall synchronization time when only uploads are needed.
  - Adaptive Server Anywhere clients can specify an incremental upload option to reduce memory requirements for large uploads.

    ☞ For more information, see "Customizing synchronization" [*MobiLink Synchronization User's Guide*, page 186].
  - Adaptive Server Anywhere clients can permit concurrent modification of rows during synchronization.

    ☞ For more information, see "Concurrency during synchronization" [*MobiLink Synchronization User's Guide*, page 187].

- **Scheduling synchronization** You can use an extended option to configure the *dbmlsync* utility or a synchronization definition to synchronize according to a schedule.

  ☞ For more information, see "Scheduling synchronization" [*MobiLink Synchronization User's Guide*, page 198].

- **Adaptive Server Anywhere client synchronization utility enhancements** There are several enhancements to the *dbmlsync* utility:
  - You can supply the `-mp` and `-mn` options to supply or change the MobiLink password.
  - You can supply repeated `-n` options to synchronize more than one synchronization definition.

- The `-v` option now generates more useful information, including options set in the synchronization definition.
- The `-r` option is extended to allow more flexibility in uploads when the recorded progress indicators in the client and consolidated databases do not match.
- The `-x` option renames and restarts the transaction log. This option is useful if you use the consolidated database as a backup of the data at the client, so that client-side backups are not required.
- If you do not specify connection parameters on the command line, *dbmlsync* displays a dialog on which you can provide connection parameters and startup options.
- The *dbmlsync* window displays synchronization progress, and allows you to cancel synchronization.

  ☞ For more information, see "MobiLink synchronization client" [*MobiLink Synchronization Reference*, page 36].

♦ **New MobiLink synchronization server options** The MobiLink synchronization server provides additional options.

☞ For more information, see "MobiLink Synchronization Server Options" [*MobiLink Synchronization Reference*, page 3].

♦ **New script events** New scripts have been added for handling and reporting errors arising from the ODBC Driver Manager, and to provide additional flexibility when designing synchronization techniques.

☞ For more information, see the following:

- "handle_odbc_error connection event" [*MobiLink Synchronization Reference*, page 177]
- "prepare_for_download connection event" [*MobiLink Synchronization Reference*, page 192]
- "report_odbc_error connection event" [*MobiLink Synchronization Reference*, page 196]

♦ **Interface to dbmlsync features** Developers using the C programming language can add features of the *dbmlsync* utility to their application.

☞ For more information, see "Initiating synchronization from an application" [*MobiLink Synchronization User's Guide*, page 188].

SQL Remote new features

SQL Remote version 7.0.1 includes the following new features.

♦ **More message links on Novell NetWare** You can now use the FTP and SMTP/POP message links on Novell NetWare.

- **Enhanced verbose mode** Verbose mode for the Message Agent now writes out full connection information, with user IDs and passwords replaced by asterisks.

UltraLite new features

UltraLite 7.0.1 introduces several new features:

- **Develop UltraLite applications with MobileBuilder** MobileBuilder is a rapid application development tool for small devices. You can use MobileBuilder to develop UltraLite applications, using the UltraLite Database component included with SQL Anywhere 7.0.1.

  ☞ For information on this feature, see the separate book *Developing UltraLite Applications with MobileBuilder.*

- **New synchronization stream for Palm Computing Platform** In addition to the current **ULPalmDBStream** synchronization stream, a new synchronization stream is available for the Palm Computing Platform in this release. The new stream is called **ULConduitStream**, and in many circumstances this stream can provide dramatic performance improvements for HotSync synchronization.

  This feature superceded
  A new conduit-based synchronization stream introduced in version 8.0.0 supercedes both **ULPalmDBStream** and **ULConduitStream**.

- **Monitoring and canceling synchronization** You can view synchronization status and build the ability to cancel synchronization into your UltraLite applications.

  ☞ For more information, see "Monitoring and canceling synchronization" [*UltraLite Embedded SQL User's Guide*, page 65].

- **User authentication in MobiLink** MobiLink synchronization now has its own user authentication scheme. Password fields and methods have been added to the UltraLite synchronization parameters to take advantage of this scheme.

  ☞ For more information, see "Stream parameters reference" [*UltraLite Database User's Guide*, page 179].

- **New platforms for secure synchronization** You can now use the transport-layer security features for synchronization from a wider range of target platforms, including Windows CE on the Hitachi SH4 chip, and VxWorks on Intel x86 chips and on the Windows VxSim emulator.

  ☞ For more information, see the following:

- "SQL Anywhere Studio Supported Platforms" [*Introducing SQL Anywhere Studio*, page 121].
- "Stream parameters reference" [*UltraLite Database User's Guide*, page 179].
- "Synchronization on Windows CE" [*UltraLite Embedded SQL User's Guide*, page 96].

VxWorks unsupported in version 9
Support for the VxWorks platform is dropped entirely in version 9.

- **Non-synchronizing tables** You can include tables in the reference database that are included in the UltraLite database, but are not synchronized. Other than synchronization, the tables can be used like any other table in the remote database.

  ☞ For more information, see "Including non-synchronizing tables in UltraLite databases" [*UltraLite Database User's Guide*, page 155].

- **Windows CE emulator support enhancements** You can now run UltraLite applications under Windows CE x86 emulators.

- **Synchronization optimization** Client applications can specify that a synchronization includes only uploads, and that no download phase should be attempted. This option lessens the overall synchronization time when only uploads are needed, especially over slow communication links.

  ☞ For more information, see "Stream parameters reference" [*UltraLite Database User's Guide*, page 179].

- **Automatic HTTP version detection** The MobiLink synchronization server now detects and uses the HTTP version used by each client. This capability renders the **version** parameter on the MobiLink synchronization server `-x` option redundant.

  ☞ For information on the MobiLink synchronization server command line, see "MobiLink Synchronization Server Options" [*MobiLink Synchronization Reference*, page 3].

- **Client port specification** You can specify, at a client, a range of ports used by a client during synchronization. This feature can be useful when synchronizing from a client inside a firewall to a MobiLink synchronization server outside.

  ☞ For more information, see "Stream parameters reference" [*UltraLite Database User's Guide*, page 179].

CHAPTER 9

What’s New in Version 7.0.0

About this chapter

This chapter provides an overview of the new features and behavior changes introduced in Adaptive Server Anywhere version 7.0.0.

Contents

New features in version 7.0.0

The primary format for the documentation is HTML Help. The HTML Help Home Page gives you easy access to the new features, information about how to contact Sybase, and other starting points for this release.

If you do not have Internet Explorer 4.0 or HTML Help installed on your machine, you will install Windows Help instead of HTML Help. The content is the same except for the HTML Help home page, which is not present in Windows Help.

If you are using Windows Help, you should look at Chapter 1 of *Getting Started with Adaptive Server Anywhere* for information on Adaptive Server Anywhere new features, and at the first chapters of the *UltraLite Developer's Guide* and the *Replication and Synchronization Guide* for information on new features in those technologies.

Adaptive Server Anywhere new features

This section introduces the new features in Adaptive Server Anywhere version 7.0. It provides an exhaustive listing of major and minor new features, with cross references to locations where details of each feature appear in the manuals.

If you have the printed version of this book, and if you do not have the complete SQL Anywhere Studio documentation set, you should look in the online documentation for the detailed description of each feature. To locate the information in the online documentation, go to the index and enter the specified title.

Administration and ease of use enhancements

- **Task scheduling and event handling in the database** You can now add scheduled operations to the database. This can be useful for automatic backups, periodic reports to fill summary tables, and other tasks.

  The database server can also be instructed to execute event handlers when certain events occur, including disk space thresholds on the drives holding the database file or the transaction log file, or failed connection attempts.

  Event handlers can be created and altered using Sybase Central, and can be debugged using the Adaptive Server Anywhere debugger.

  ☞ For more information see "Automating Tasks Using Schedules and Events" [*ASA Database Administration Guide*, page 267], and "CREATE EVENT statement" [*ASA SQL Reference*, page 304].

- **Updated Sybase Central** Sybase Central has been rewritten and contains significant new features. In particular, Sybase Central is now

available from any supported platform, and not just Windows operating systems.

- **Updated Interactive SQL** The Interactive SQL [dbisql] utility has been enhanced and is now available as a windowed-application from any supported platform.

- **New validation features** Additional validation of databases is provided by the new VALIDATE INDEX statement and by enhancements to the VALIDATE TABLE statement. This statement is called both by the Validation [dbvalid] utility, and by the sa_validate system procedure. The enhancements are available through all these routes.

  ☞ For more information, see "VALIDATE INDEX statement" [*ASA SQL Reference,* page 602], and "VALIDATE TABLE statement" [*ASA SQL Reference,* page 603].

- **Lock troubleshooting** A new system procedure, sa_locks, provides information on locks in the database. If lock issues are identified, information on the connection processes involved can be found using the **AppInfo** connection property.

  ☞ For more information, see "sa_locks system procedure" [*ASA SQL Reference,* page 721], and "AppInfo connection parameter [APP]" [*ASA Database Administration Guide,* page 175].

- **Unloading result sets** The new UNLOAD SQL statement allows query result sets to be unloaded into a comma-delimited text file.

  ☞ For more information, see "UNLOAD statement" [*ASA SQL Reference,* page 588].

- **Validate backup copies of databases** If you backup a database using the WAIT BEFORE START clause, the backup copy is created in such a fashion that it can be started in read-only mode and validated.

  ☞ For more information, see "BACKUP statement" [*ASA SQL Reference,* page 263].

- **Default global autoincrement** This feature provides an easy way to generate integer keys which are unique across all databases in a SQL Remote replication environment.

Integration with distributed computing architectures

- **Distributed transactions and three-tiered computing** Distributed transactions include operations on more than one server in a single transaction. A transaction server controls the commit and rollback behavior of distributed transactions.

  In this release, Adaptive Server Anywhere can participate in distributed transactions coordinated by the Microsoft Distributed Transaction

Coordinator (DTC). Products such as Sybase Enterprise Application Server and Microsoft Transaction Server can use DTC for transaction coordination, so DTC support enables Adaptive Server Anywhere to participate in three-tiered computing with these products.

☞ For more information, see "Three-tier Computing and Distributed Transactions" [*ASA Programming Guide*, page 455].

Integration with COM

- **OLE DB provider** OLE DB is a data access model from Microsoft. It uses the Component Object Model (COM) interfaces and, unlike ODBC, OLE DB does not assume that the data source uses a SQL query processor. While it has been possible to access Adaptive Server Anywhere via OLE DB using an OLE DB/ODBC bridge provided by Microsoft, this release of Adaptive Server Anywhere includes an OLE DB provider. This provider brings several benefits:
  - OLE DB is the principal data access option for the forthcoming version of Windows CE.
  - Some features, such as updating through a cursor, are not available using the OLE DB/ODBC bridge.
  - If you use the Adaptive Server Anywhere OLE DB provider, ODBC is not required in your deployment.

  ☞ For more information, see "The OLE DB and ADO Programming Interfaces" [*ASA Programming Guide*, page 313].

Connectivity enhancements

- **Java connectivity improvements** If you use jConnect to connect to Adaptive Server Anywhere from a Java application, you can now take advantage of many of the features previously available only to ODBC and Embedded SQL applications, such as autostarting of database servers, and detailed control over network communications using communications parameters.

- **TCP/IP connectivity** Establishing a client/server connection over TCP/IP is now simpler. Clients no longer need to specify the port number when attempting to connect, even if the server is running on a port other than the default port number (2638). If the default port number is in use when a database server is started, the server acquires an unused port number from the operating system.

  If you are trying to connect through a firewall (using UseUDP=NO), and if the database server is not running on port 2638, you must still specify a port number. For more information on this scenario, see "Connecting across a firewall" [*ASA Database Administration Guide*, page 95].

  The Server Location [dblocate] utility displays all Adaptive Server Anywhere database servers running TCP/IP on a network. For more information, see "The Server Location utility" [*ASA Database Administration Guide*, page 518].

- **SPX connectivity** You can use the SPX protocol for connecting to databases. This feature is particularly useful in Novell NetWare environments with IPX/SPX as the primary network protocol. SPX is recommended over IPX.

  ☞ For more information on SPX at the client, see "CommLinks connection parameter [LINKS]" [*ASA Database Administration Guide,* page 179]. For information on SPX on the server, see "-x server option" [*ASA Database Administration Guide,* page 162]. For network communication parameters that you can use with SPX, see "Network communications parameters" [*ASA Database Administration Guide,* page 201].

Performance enhancements

- **Dynamic cache sizing** On Windows NT and UNIX, the size of the database server cache increases and decreases depending on the load on the database server and the other demands on system memory. This feature removes the need for choosing an explicit cache size under in many circumstances, and can also boost performance. On Windows 95/98, a less comprehensive cache resizing is implemented.

  ☞ For more information, see "Using the cache to improve performance" [*ASA SQL User's Guide,* page 176].

- **Indexing enhancements** Additional flexibility has been added to control the amount of information stored in indexes (the **hash size**) to improve index selectivity. Also, the architecture of primary and foreign key indexes has been altered.

  For indexes on multiple columns, or for indexes on columns in which the first set of characters or digits are similar across many rows, control over hash size provides a way of increasing the selectivity of indexes, and so improving performance.

  ☞ For more information, see "Using indexes" [*ASA SQL User's Guide,* page 163], "CREATE INDEX statement" [*ASA SQL Reference,* page 319], and "CREATE TABLE statement" [*ASA SQL Reference,* page 361].

  ☞ For information on how to find the number of levels in an index, see "sa_index_levels system procedure" [*ASA SQL Reference,* page 720].

  In previous releases, primary and foreign keys have had a single index automatically associated with them, which describes all primary key values and all the related foreign key entries. In some situations, this architecture lead to poor performance. The new index organization separates these indexes, which leads to improved performance in some situations.

  ☞ For more information on key indexes, see "Using keys to improve query performance" [*ASA SQL User's Guide,* page 182].

Your database must be unloaded and reloaded to take advantage of variable hash size indexes, and separate key indexes. Running the Upgrade [dbupgrad] utility is not sufficient.

- **Separate storage for string extensions** The physical storage of values longer than 255 characters has been reorganized. The pages allocated for a table are now divided into two disjoint sets. The first set contain only rows. Where a column value in a row contains a string longer than 255 characters, only a prefix of the string (up to 255 characters) and a reference to a string extension are stored in the row. For strings longer than 255 characters, the string extensions are allocated in the second set of table pages. This change improves performance on queries requiring scans of tables storing long values because a sequential scan of a table only needs to traverse the pages in the first set.

  Your database must be unloaded and reloaded to take advantage of this feature.

- **New database page-sizes** In addition to 1K, 2K, and 4K page sizes, you can now create databases with page sizes of 8K, 16K or 32K.

  Large page sizes can improve performance in some cases, particularly for large databases. However, there are additional memory requirements with large page sizes, and so they should only be used after investigation of the costs and benefits.

  ☞ For more information, see "Creating a database using the dbinit command-line utility" [*ASA Database Administration Guide*, page 486], and "CREATE DATABASE statement" [*ASA SQL Reference*, page 292].

  ☞ For information on the number of indexes per table and how it depends on page size, see "Size and number limitations" [*ASA Database Administration Guide*, page 674].

- **Optimizer tuning** You can use the OPTIMIZATION_GOAL option to instruct the optimizer to optimize for the time it takes to return the first row of a query, or the overall time it takes to return all rows. The default is to optimize for the first row. If you are using applications such as PowerBuilder DataWindow applications, which require a complete result set, you may wish to change this option setting.

  ☞ For more information, see "OPTIMIZATION_GOAL option [database]" [*ASA Database Administration Guide*, page 613].

- **Optimizer enhancements** Further enhancements to the optimizer have been implemented to assist with performance of queries that use internal temporary tables and that use primary and foreign key indexes. These enhancements require no user action.

- **Larger numbers of users and other identifiers** Many identifiers in the system tables identifying database objects have been changed from SMALLINT to UNSIGNED INTEGER. This change increases the number of objects that can be held in a database without violating an absolute limit.

- **Inserting and exporting images and documents** Two new system external functions allow you to read and write the contents of files. These functions allow direct inserting of images, documents, and so on into tables from environments such as Interactive SQL.

  ☞ For more information, see "Inserting documents and images" [*ASA SQL User's Guide*, page 360], "xp_read_file system procedure" [*ASA SQL Reference*, page 761], and "xp_write_file system procedure" [*ASA SQL Reference*, page 762].

- **New interface for external functions** Stored procedures and user-defined functions that reference external libraries now use a new interface. The new interface provides a wider range of operating systems (including UNIX), a wider range of data types, removes the restriction that returned data fit into 255 bytes, and supports NULL as a valid value for arguments. The older interface is still supported, but should not be used for new development work.

  ☞ For more information, see "Creating procedures and functions with external calls" [*ASA SQL User's Guide*, page 664].

- **START DATABASE, STOP DATABASE and STOP ENGINE statements** These statements were previously available only from Interactive SQL. They are now available from all applications.

  ☞ For more information, see "START DATABASE statement" [*ASA SQL Reference*, page 566], "STOP DATABASE statement" [*ASA SQL Reference*, page 575], and "STOP ENGINE statement" [*ASA SQL Reference*, page 576].

- **FIRST and TOP clause in updates and deletes** The FIRST and TOP clauses can be used to update or delete only the first one or more of any set of rows satisfying a WHERE clause.

  ☞ For more information, see "DELETE statement" [*ASA SQL Reference*, page 399], and "UPDATE statement" [*ASA SQL Reference*, page 592].

- **Explicit table locking** The LOCK TABLE statement allows direct control over concurrency at a table level, independent of the current isolation level.

  ☞ For more information, see "LOCK TABLE statement" [*ASA SQL Reference*, page 493].

- **Expressions in Transact-SQL outer joins** The *= and =* operators in a WHERE clause provide a way of specifying outer joins for users who wish to use the Transact-SQL dialect. In previous releases, only column names could be used in such joins. Now as long as each side of the join operator refers to a single table, any expression can be used in these joins. For example, the following query is now possible:

  ```
  select *
  from customer, sales_order
  where substr( customer.id, 1, 1 ) *=
        substr( sales_order.cust_id, 1, 1)
  ```

- **Cursors in stored procedures can reference variables** In stored procedures and user-defined functions, you can declare a cursor on a variable using the following syntax:

  DECLARE *cursor-name* **CURSOR USING** *variable-name*

  where *variable-name* is a string variable containing the SELECT statement for the cursor.

  ☞ For more information, see "DECLARE CURSOR statement [ESQL] [SP]" [*ASA SQL Reference*, page 390].

- **Additional database and server properties** The following properties have been added:
  - **PageSize** The database server uses a single page size from startup until it is closed down. This page size is the maximum page size database that can be mounted by the database server. You can now obtain this page size using the PageSize server-level property function:

    ```
    select property( 'PageSize' )
    ```

  - **AppInfo** This function provides identification information for a client application. It is a connection property:

    ```
    select connection_property( 'AppInfo' )
    ```

    ☞ For more information, see "AppInfo connection parameter [APP]" [*ASA Database Administration Guide*, page 175].
  - **IsRuntimeServer** This function returns YES if the database server is a limited desktop runtime personal database server. Otherwise, it returns NO.
  - **Log truncation points** Properties for replication-specific log offsets have been added. The properties **LTMTrunc**, **RemoteTrunc**, and **SyncTrunc** return the minimal confirmed log offset for the Replication Agent, SQL Remote, and MobiLink *dbmlsync* replication, respectively. These offsets are also known as truncation points because

they indicate the point at which the transaction log can be truncated. The property **CurrentRedoPos** returns the current offset in the log file, where the next database operation is to be logged.

☞ For a complete list of property functions and information on how to access them, see "Database properties" [*ASA Database Administration Guide*, page 647].

- **Referential integrity checks before commit** A new system procedure (sa_check_commit) allows you to check for referential integrity conflicts before committing changes to a database.

  ☞ For more information, see "sa_check_commit system procedure" [*ASA SQL Reference*, page 707].

- **SQL function enhancements** The following functions have been added or enhanced.
  - **REPLACE function** This new function replaces all occurrences of a substring with another substring.

    ☞ For more information, see "REPLACE function [String]" [*ASA SQL Reference*, page 176].
  - **LIST function enhancement** The LIST function now accepts an optional second value, which is the delimiter string that separates the list items.

    ☞ For more information, see "LIST function [Aggregate]" [*ASA SQL Reference*, page 148].

- **Output redirection change** The output redirection functionality in Interactive SQL has been extended to include three new Interactive SQL statements and an Export option in the File menu.

  You can now use an OUTPUT TO statement to redirect content from the Results pane to a new file. You can add an APPEND clause to append the content to the end of an existing file, or you can add a VERBOSE clause to include the content of the Messages pane with the output.

  In earlier versions, output redirection in Interactive SQL could only be done with the symbols >#, >>#, >&, and >>&. You can still use these symbols, but the new Interactive SQL statements allow for more precise output and code that is easier to read.

  ☞ For more information, see "Exporting query results" [*ASA SQL User's Guide*, page 534] in the *ASA User's Guide*.

- **Embedded SQL enhancements** A new function, **db_string_ping_server**, has been introduced to test that a database server can be located with a specified current connection string.

☞ For more information, see “db_string_ping_server function” [*ASA Programming Guide*, page 220].

- **New LOAD TABLE / UNLOAD TABLE format** A new format has been added to the UNLOAD TABLE statement to allow data to be output in BCP format and to the LOAD TABLE statement to allow the import of Adaptive Server Enterprise generated BCP out files containing blobs.

  ☞ For more information, see “LOAD TABLE statement” [*ASA SQL Reference*, page 486] or “UNLOAD TABLE statement” [*ASA SQL Reference*, page 590].

- **Last default timestamp** The new global variable **@@dbts** returns a TIMESTAMP value that represents the last value generated for a column using DEFAULT TIMESTAMP.

  ☞ For more information, see “Global variables” [*ASA SQL Reference*, page 39].

- **Troubleshooting enhancements** On starting the database server, you can log operations executed by the server to a file using the `-zr` option. You can use the sa_server_option procedure to control the same behavior while the server is running.

  ☞ For more information, see “sa_server_option system procedure” [*ASA SQL Reference*, page 739], and “-zr server option” [*ASA Database Administration Guide*, page 167].

- **Archive backup on NetWare** The archive backup format is now supported on NetWare. Archive backups to tape require NetWare 5.

  ☞ For more information, see “BACKUP statement” [*ASA SQL Reference*, page 263].

- **Added filtering for dbtran** The command version of the Log Translation [dbtran] utility allows further filtering of the output.

  ☞ For more information, see “Log translation utility options” [*ASA Database Administration Guide*, page 510].

- **Faster table truncation** The TRUNCATE TABLE statement is much faster for version 7.0 databases, for tables with foreign keys.

- **Suppressing event log messages** If you run the database server as a Windows NT service, you can suppress event log messages using a registry entry.

  ☞ For more information, see “Suppressing Windows event log messages” [*ASA Database Administration Guide*, page 130].

SQL Remote new features

SQL Remote version 7.0 includes the following new features.

- **Globally unique primary keys** You can now use a DEFAULT GLOBAL AUTOINCREMENT column in an Adaptive Server Anywhere database, together with a GLOBAL_DATABASE_ID option setting in each database, to guarantee unique primary keys throughout a SQL Remote installation of Adaptive Server Anywhere databases. This is a more convenient method than the more manual primary key pool technique.

  ☞ For more information, see "Using global autoincrement default column values" [*SQL Remote User's Guide*, page 129].

- **Internal unload for dbxtract** The Extract [dbxtract] utility now uses the UNLOAD statement introduced in Adaptive Server Anywhere version 7.0 by default, rather than the slower OUTPUT statement. Options have been introduced to allow you to choose a combination of internal (server-side) and external (client-side) unload and load operations.

  ☞ For a complete listing of options, see "The extraction utility" [*SQL Remote User's Guide*, page 303].

MobiLink and UltraLite new features

Following is a list of changes and additions to the software since version 6.0.3.

- **Adaptive Server Anywhere clients** MobiLink technology now supports Adaptive Server Anywhere as a client, as well as UltraLite applications.

  ☞ For more information, see "Adaptive Server Anywhere Clients" [*MobiLink Synchronization User's Guide*, page 167].

- **mlxtract creates Adaptive Server Anywhere client databases** *mlxtract* creates Adaptive Server Anywhere databases, suitable for use as MobiLink clients, using an Adaptive Server Anywhere reference database as a template.

- **Synchronization script versions** Synchronization scripts can now be grouped by assigning a script version name with each script. This feature allows the MobiLink synchronization server to respond differently when synchronizing different types of applications, or different versions of the same application.

- **New data types** LONG BINARY and LONG VARCHAR data types can now be replicated using MobiLink technology.
- **New HotSync conduit** A new HotSync conduit allows HotSync synchronization with a centrally located MobiLink synchronization server. The MobiLink synchronization server no longer needs to be on the same machine as the HotSync manager is.
- **ScoutSync conduit** UltraLite applications for the Palm Computing Platform can now synchronize using ScoutSync technology, available from Riverbed Technologies.
- **report_error script** A new script provides a convenient way to report errors during synchronization. The **report_error** script also makes debugging the behavior of the **handle_error** script much easier. The **report_error** script has the same parameters as the **handle_error** script, except that the first parameter is the action code returned by **handle_error**.

Behavior changes in version 7.0.0

This section lists deprecated and unsupported features, and behavior changes from previous versions of the software.

Adaptive Server Anywhere behavior changes

Deprecated and unsupported features

This list includes features that are no longer supported and that impact existing applications.

- **Windows 3.x and Windows CE 2.0 no longer supported** Windows 3.1 and Windows 3.11 are no longer supported. Windows CE 2.0 is no longer supported.
- **DDE protocol no longer supported** The DDE protocol was used to communicate from 16-bit Windows 3.x applications to a Windows 95/98 database server on the same machine. It is no longer required: Windows 3.x applications based on older versions of the software can use TCP/IP to communicate with the version 7.0 database server.
- **IPX protocol deprecated** Although communications using IPX are still supported in the present release, it is highly recommended that you use the SPX protocol instead. The communications parameters are the same as for IPX, and performance is better. Support for IPX will be dropped in a future release.

  By default, both the database server and the client software do not start the IPX protocol unless you instruct it to do so explicitly using the `-x` option or the **CommLinks** connection parameter. The SPX protocol is started by default.

  ☞ For information on using SPX from the client side, see "CommLinks connection parameter [LINKS]" [*ASA Database Administration Guide*, page 179]. For information on using SPX from the server side, see "-x server option" [*ASA Database Administration Guide*, page 162].
- **Deprecated network communication parameters** The Broadcast and CommAutoStop communication parameters are still allowed, but have no effect. They will not be supported in future versions of Adaptive Server Anywhere.
- **No dbclient compatibility executable** In version 6, the dbcli6.exe utility provided easier compatibility with version 5 client connection methods. There is no comparable utility in version 7.

Behavior changes

This list includes behavior changes in existing features that may impact applications or have an impact during development or database management.

- **Interactive SQL changes** The new version of Interactive SQL has some changes from previous versions. As it is an interactive tool, most do not need documentation.

  ☞ The supported formats for INPUT and OUTPUT statements have changed, and now include:

  - **INPUT** ASCII, DBASE, DBASEII, DBASEIII, EXCEL, FIXED, FOXPRO, LOTUS
  - **OUTPUT** ASCII, DBASE, DBASEII, DBASEIII, EXCEL, FIXED, FOXPRO, HTML, LOTUS, SQL

- **Server name space change** It is now disallowed for more than one database server with the same name to be running on TCP/IP anywhere on the network. Previously, multiple servers with the same name were allowed as long as they were on separate ports.

- **Mirrored logs deleted when DELETE_OLD_LOGS is ON** Previously, any mirror of an old transaction log was not deleted, although the primary copy of the old transaction log was deleted.

- **ODBC SQLDescribeCol behavior** A **SQLDescribeCol** call on the **@@identity** field now returns SQL_BIGINT. In earlier versions, it returned SQL_INTEGER.

- **Update constraints** A new ANSI_UPDATE_CONSTRAINTS option has been added. Setting this option to CURSORS, or STRICT, restricts updates to those allowed by the ANSI standard. Setting this option to OFF, which is the historical behavior, allows a greater range of updates.

  ☞ For more information, see "ANSI_UPDATE_CONSTRAINTS option [compatibility]" [*ASA Database Administration Guide*, page 576], and "UPDATE statement" [*ASA SQL Reference*, page 592].

- **Identifier length limit** Long identifiers are treated more consistently than in the past. Identifiers longer than 128 bytes were sometimes accepted and sometimes not, depending on the type of database object being named. Now any attempt to define identifiers longer than 128 bytes reports an error.

  ☞ For more information, see "Identifiers" [*ASA SQL Reference*, page 7].

- **jConnect connections** If you use the REMOTEPWD field to connect via jConnect to a named database on an Adaptive Server Anywhere database server, you must assign the field in a different manner for jConnect version 4.2 and above, included with this software.

  ☞ For more information, see "Supplying a URL for the server" [*ASA Programming Guide*, page 112].

- **User-defined errors** Within procedures and triggers, you can declare exceptions in the range 99000 to 99999 as user-defined errors in compound statements. You can use the SIGNAL statement to handle these errors.

  ☞ For more information, see "BEGIN statement" [*ASA SQL Reference*, page 267], and "SIGNAL statement" [*ASA SQL Reference*, page 565].

- **LOAD TABLE and UNLOAD TABLE security** A database server option has been added to control the permissions required to execute the LOAD TABLE and UNLOAD TABLE statements.

  ☞ For more information, see "-gl server option" [*ASA Database Administration Guide*, page 147].

- **@@identity in triggers** If a table (T1) with an autoincrement column has an insert trigger which causes an insert into a second table (T2) also having an autoincrement column, it was not previously possible to obtain the autoincrement value assigned for T1 after the insert had completed. At that point, the value of @@identity would be the value assigned to T2. The behavior of @@identity has been altered to make the value accessible.

  ☞ For the new behavior of @@identity within triggers, see "@@identity global variable" [*ASA SQL Reference*, page 45].

- **Embedded SQL DECL_FIXCHAR** In previous releases, the SQL preprocessor converted a type DECL_FIXCHAR to an array. For example, `DECL_FIXCHAR(12)` was converted to char `name_fixchar[12]`.

  In the current release, the SQL preprocessor converts DECL_FIXCHAR declarations to `DECL_FIXCHAR(12) name_fixchar`, and references are of the form `name_fixchar.array[i]`.

SQL Remote behavior changes

The following behavior has changed in version 7.0:

- **dbxtract uses internal unload** The default behavior of dbxtract is now to use the UNLOAD statement to unload data on the server side, rather than the OUTPUT statement to unload data on the client side. The `-ii`, `-ix`, `-xi`, `-xx` options allow you to choose which combination of internal and external operations to use, and replace the options `-i` and `-x` available in previous releases.

  ☞ For a complete listing of options, see "The extraction utility" [*SQL Remote User's Guide*, page 303].

MobiLink and UltraLite behavior changes

- **New table and script names** The tables that hold synchronization scripts and related information in the consolidated database now have new names. Previously, these table names began with the prefix **ul_**. This prefix has been changed to **ml_**. Older consolidated databases must be upgraded for compatibility with version 7.0.

  Similarly, the stored procedure that facilitates adding table scripts has been renamed from **sp_table_script** to **ml_add_table_script** and the stored procedure that facilitates adding connection scripts has been renamed from **sp_connection_script** to **ml_add_connection_script**.

  Under DB2, these names are truncated to 18 characters.

- **Synchronization scripts require a version name** Synchronization scripts must now be assigned a script version name. Script version names allow the MobiLink synchronization server to treat different clients differently.

CHAPTER 10

What's New in Version 6.0.3

About this chapter

This chapter provides an overview to the new features and behavior changes introduced in Adaptive Server Anywhere 6.0.3.

Contents

New features in version 6.0.3

This section lists the new features introduced in components of SQL Anywhere Studio version 6.0.3.

Adaptive Server Anywhere new features

In addition to bug fixes, Adaptive Server Anywhere version 6.0.3 includes new features in both the software and the documentation.

- **Combined stored procedure and Java debugger** The Java debugger that was provided in previous releases has been upgraded. The new version of the debugger is able to debug not only Java classes within the database, but also SQL stored procedures and triggers.

  ☞ For information on how to use the debugger, see "Debugging Logic in the Database" [*ASA SQL User's Guide*, page 673].

- **Read-only databases** You can designate a database as read only when you start a database server. This feature makes deployment of databases on read-only media, such as CD-ROMs, more straightforward.

  The **ReadOnly** database property returns ON for read-only databases, and OFF for databases that are not being run in read-only mode.

  ☞ For more information on read-only databases, see "-r server option" [*ASA Database Administration Guide*, page 157].

- **Computed column extensions** New flexibility has been added to computed columns. You can now add computed columns to non-empty tables, and change the expression associated with a computed column. Computed columns are recalculated in a number of circumstances to ensure that the values are reliable.

  ☞ For more information, see "Working with computed columns" [*ASA SQL User's Guide*, page 46], and "Inserting and updating computed columns" [*ASA SQL User's Guide*, page 47].

  ☞ For information on syntax, see "ALTER TABLE statement" [*ASA SQL Reference*, page 250].

- **Support for the euro** Collations have been added that include the euro currency symbol. These collations are the 1252LATIN1 and ISO9LATIN1 collations.

  ☞ For more information, see "The 1252LATIN1 collation" [*ASA Database Administration Guide*, page 313], and "The ISO9LATIN1 collation" [*ASA Database Administration Guide*, page 313].

- **Additional collations** Other collations have been added to the list of supplied collations, including 852POL (OEM Code Page 852 (Latin 2), with Polish ordering), 1250POL (Windows Latin2 code page 1250 with Polish ordering), 1250Latin2 (Windows Latin2 Code page 1250). 932JPN (Japanese), 936ZHO (similar to EUC_CHINA), and 950TAI (similar to EUC_TAIWAN).

  ☞ For a complete list, see "Choosing collations" [*ASA Database Administration Guide*, page 307].

- **New Windows CE platforms** The SH4 and ARM processors are now supported under Windows CE 2.1x.

  ☞ For a list of supported devices, chips, and operating system versions, see "Windows and NetWare operating systems" [*Introducing SQL Anywhere Studio*, page 125].

- **ALTER TABLE extensions** The ALTER TABLE statement has been extended to provide SQL/92-compliant clauses to set and drop defaults on columns. These clauses are an alternative to the existing MODIFY clause.

  ```
  ALTER column-name SET DEFAULT default-value
  | ALTER column-name DROP DEFAULT
  ```

  ☞ For more information, see "ALTER TABLE statement" [*ASA SQL Reference*, page 250].

- **LOAD TABLE extensions** You can now load specific columns of a table using the LOAD TABLE statement. A new CHECK CONSTRAINTS option has been introduced to address rebuild issues.

  ☞ For more information, see "LOAD TABLE statement" [*ASA SQL Reference*, page 486].

- **Easier connections across firewalls** A set of communications parameters has been introduced to allow easier connections across firewalls.

  ☞ For more information, see "Connecting across a firewall" [*ASA Database Administration Guide*, page 95].

- **BACKUP statement extended** The MATCH keyword has been introduced to allow renaming of the backup copy of the transaction log to a filename of the form *YYMMDDnn.log*. If you use this keyword, you can execute the same statement multiple times without writing over data.

  ☞ For more information, see "BACKUP statement" [*ASA SQL Reference*, page 263].

- **Easier unload and reload** The Unload [dbunload] utility has been enhanced (`-ar` option) to allow a single-step unload and reload of a database that can be used whether or not your database is involved in replication.

  ☞ For more information, see "Unload utility options" [*ASA Database Administration Guide*, page 537], and "Rebuilding databases" [*ASA SQL User's Guide*, page 539].

- **Temporary file location** The database server checks for a new environment variable, ASTMP, when deciding on the location of the temporary file. This allows you to use directories other than system temporary directories for the temporary file.

  ☞ For more information, see "ASTMP environment variable" [*ASA Database Administration Guide*, page 246].

- **New system procedures** New system procedures allow DBA users to override some database server options (sa_server_option), and to flush the database server cache (sa_flush_cache).

  ☞ For more information, see "System and catalog stored procedures" [*ASA SQL Reference*, page 707].

- **Character set translation tuning** You can control the application locale used in character set translation for an individual connection using the new CharSet connection parameter.

  ☞ For more information, see "CharSet connection parameter [CS]" [*ASA Database Administration Guide*, page 177].

- **Re-organized Performance Monitor statistics** The statistics made available to the Windows NT Performance Monitor have been organized into areas. Some statistics have been added, and ones of little use have been removed.

  ☞ For a list of available statistics, see "Monitoring database statistics from Windows Performance Monitor" [*ASA SQL User's Guide*, page 190].

- **Database properties from the utility database** You can now execute SELECT statements, with no tables, against the utility database. This is primarily of use for retrieving database and connection properties.

  ☞ For more information, see "Using the utility database" [*ASA Database Administration Guide*, page 262].

- **New database properties** The following properties are available using the property function.

- **IsNetworkServer** Returns YES if connected to a network database server, and NO if connected to a personal database server.

  ☞ For more information, see "Server-level properties" [*ASA Database Administration Guide*, page 657].
- **DefaultCollation** You can use the new **DefaultCollation** property to find the default collation to be used when creating a database.

  ☞ For more information, see "Finding the default collation" [*ASA Database Administration Guide*, page 325], and "Server-level properties" [*ASA Database Administration Guide*, page 657].
- **MultiByteCharSet** You can use the **MultiByteCharSet** database property to determine whether a database is using a multi-byte or single-byte collation.

  ☞ For information on this property, see "Database-level properties" [*ASA Database Administration Guide*, page 664].

♦ **Support for some JDBC 2.0 functions in internal JDBC** The internal server-side JDBC driver now supports functions from the JDBC 2.0 interface. Server-side Java applications can now use features such as such as scrollable, updateable result sets and batch updates. A side effect is that you can now access result sets from Java methods from Interactive SQL.

☞ For more information, see "JDBC in the database features" [*ASA Programming Guide*, page 106].

♦ **Using the main method in Java classes** You can now execute a `main` method of a Java class from SQL.

☞ For more information, see "Calling the main method" [*ASA Programming Guide*, page 93].

♦ **User-defined functions using Java classes** You can wrap a Java method in a SQL user-defined function.

☞ For more information, see "CREATE FUNCTION statement" [*ASA SQL Reference*, page 315].

♦ **Extensions to stored procedures using Java methods** You can use OUT and INOUT parameters in stored procedures that are wrappers for Java methods.

☞ For more information, see "Returning values from Java via stored procedures" [*ASA Programming Guide*, page 96].

♦ **Multi-threaded Java classes in the database** Support has been added for the package `java.lang.thread`.

☞ For more information, see "Using threads in Java applications" [*ASA Programming Guide*, page 94].

- **File access from Java** Support has been added for all the classes in the package `java.io`, including those that enable file access from classes in the database. For security reasons, a new option has been introduced, which must be set by the DBA to enable this feature.

  This feature is supported on Windows NT only.

  ☞ For more information, see "JAVA_INPUT_OUTPUT option [database]" [*ASA Database Administration Guide*, page 601].

- **CONVERT function extensions** The date and time styles supported by the CONVERT function have been extended.

  ☞ For more information, see "CONVERT function [Data type conversion]" [*ASA SQL Reference*, page 107].

- **Database server startup dialog** On 32-bit Windows operating systems, if you start a database server with no arguments, a window appears where you can specify a database file and additional parameters.

  ☞ For more information, see "Starting the server" [*ASA Database Administration Guide*, page 8].

- **Log Translation [dbtran] utility enhancements** The Log Translation [dbtran] utility permits filtering of the transaction log operations to isolate subsets of operations.

  ☞ For more information, see "Translating a transaction log using the dbtran command-line utility" [*ASA Database Administration Guide*, page 508].

- **Transaction Log [dblog] utility enhancements** The Transaction Log [dblog] utility now displays additional summary information, including offset information.

  ☞ For more information, see "Managing log files using the dblog command-line utility" [*ASA Database Administration Guide*, page 527].

- **Spawn [dbspawn] utility enhancements** The Spawn [dbspawn] utility has a `-f` option to force a server to start even if one is already running. This option uses a ForceStart connection parameter, used only by the `db_start_engine` Embedded SQL function.

  ☞ For more information, see "The Spawn utility" [*ASA Database Administration Guide*, page 523], and "db_start_engine function" [*ASA Programming Guide*, page 216].

- **Replication Agent runs as a daemon** On UNIX operating systems, you can run the Replication Agent as a daemon by supplying the `-ud` option.

☞ For more information, see "The Log Transfer Manager" [*ASA Database Administration Guide*, page 501].

SQL Remote new features

In addition to bug fixes, SQL Remote version 6.0.3 includes the following new features. Some features in Adaptive Server Anywhere that are particularly relevant to SQL Remote are also included in this list:

- **FTP and SMTP/POP support on UNIX** The range of message systems supported on UNIX operating systems has been expanded to include FTP and SMTP/POP.

  ☞ For a listing of supported operating and message systems, see "Supported Platforms and Message Links" [*SQL Remote User's Guide*, page 443].

- **Message link options stored in the database** The message link parameters that control SQL Remote behavior over each message system can now be stored in the database as opposed to the registry. This simplifies deployment and management issues related to message link parameters.

  ☞ For more information, see "Setting message type control parameters" [*SQL Remote User's Guide*, page 214], "SET REMOTE OPTION statement" [*SQL Remote User's Guide*, page 372], and "sp_link_option procedure" [*SQL Remote User's Guide*, page 394].

- **Date and time replication formats** You can now specify database options that instruct SQL Remote what format to use when replicating dates and times. These options are SR_time_format, SR_date_format, and SR_timestamp_format.

  ☞ For more information, see "Replication of dates and times" [*SQL Remote User's Guide*, page 84], and "SQL Remote options" [*SQL Remote User's Guide*, page 313].

- **Message Agent and SQL Remote Open Server run as a daemon** On UNIX operating systems you can run these applications as a daemon using the -ud option.

  ☞ For more information, see "The Message Agent" [*SQL Remote User's Guide*, page 292], and "The SQL Remote Open Server" [*SQL Remote User's Guide*, page 310].

- **Easier unload and reload of Adaptive Server Anywhere databases** The Unload [dbunload] utility has been enhanced (`-ar` option) to allow a single-step unload and reload of a database that can be used whether or not your database is involved in replication.

☞ For more information, see “Unload utility options” [*ASA Database Administration Guide*, page 537], and “Rebuilding databases” [*ASA SQL User’s Guide*, page 539].

- **Enhanced transaction log [dblog] output** The Transaction Log [dblog] utility now displays additional summary information, including offset information.

  ☞ For more information, see “Managing log files using the dblog command-line utility” [*ASA Database Administration Guide*, page 527].

- **Log Translation [dbtran] utility enhancements** The Log Translation [dbtran] utility permits filtering of the transaction log operations to isolate subsets of operations. This is of particular use to SQL Remote administrators.

  ☞ For more information, see “Translating a transaction log using the dbtran command-line utility” [*ASA Database Administration Guide*, page 508].

MobiLink and UltraLite new features

Following is a list of changes and additions to the software since version 6.0.2.

- **New data types** Real and double data types are now fully supported.

- **Character set translation** The MobiLink synchronization server now translates all uploaded characters to Unicode and passes them to the consolidated database using the Unicode ODBC API. Conversely, it translates all downloaded characters from Unicode to the character set of your UltraLite application. Character set translation within the consolidated database server can influence the results, but the new system allows more consistent behavior across multiple platforms.

  ☞ For more information, see “Controlling ODBC driver character-set translation” [*MobiLink Synchronization Reference*, page 333].

- **MobiLink synchronization server runs as a Windows NT service** When you run the MobiLink synchronization server as a service, you can configure it to continue running when you log off the Windows NT workstation.

  ☞ For more information, see “Running MobiLink Outside the Current Session” [*MobiLink Synchronization User’s Guide*, page 329].

- **DB2 setup scripts provided** To make it easier to use IBM DB2 as a consolidated database, a DB2 setup script has been added to the available set scripts.

☞ For a list of setup scripts, see "Setting up a consolidated database" [*MobiLink Synchronization User's Guide*, page 11].

Behavior changes in version 6.0.3

This section lists the behavior changes introduced in components of SQL Anywhere Studio version 6.0.3.

Adaptive Server Anywhere behavior changes

- **Adding columns with default values** When an added column has a default value, the entire column is populated with the default. In previous releases, the column was populated with NULL.
- **Permissions of referential integrity actions** When changes are made to a primary table, referential integrity actions such as cascading deletes or updates can take place on a secondary table. These actions are implemented using system triggers. The triggers now execute with the permissions of the owner of the secondary table. Previously, they executed with permissions of the owner of the primary table. The new behavior means that cascaded operations can take place between tables with different owners, without additional permissions having to be granted.
- **datediff, MONTHS, and YEARS functions** The number of months between two dates is now calculated as the number of first-of-the-months between the dates. For example, the difference between January 25 and February 2 is 1; the difference between January 1 and January 31 is 0. The number of years is now calculated as the number of first-of-the-years between the dates.

  This changes the results of these functions by one number, in some cases. The change was made for compatibility with Adaptive Server Enterprise.

  For the smaller time units there are overflow values to the DATEDIFF function that are now imposed. Previous versions of the software gave incorrect answers if the limit was exceeded.

  ☞ For a full description, see “DATEDIFF function [Date and time]” [*ASA SQL Reference*, page 114].
- **Default page size** The default page size for databases is now 2048 bytes. This choice is a better choice for many users.
- **Default database collation** The default collation used when creating databases has changed. The default depends on your operating system settings.

  ☞ For information on how to find the default collation, see “Finding the default collation” [*ASA Database Administration Guide*, page 325].

- **SQL Preprocessor default collation** If no collation is explicitly specified, the Embedded SQL Preprocessor now uses locale information to choose a default collation. If the locale information is unavailable, then 850LATIN1 will be used. The collation used is reported following the banner. Previous behavior was to use 850.

  ☞ For information on the preprocessor, see "The SQL preprocessor" [*ASA Programming Guide*, page 203].

- **Enforced server name length** The server name is checked on startup, and is truncated to a maximum value of 40 characters. On NetBIOS, it is truncated to 16 characters. From the client side, the value of the EngineName parameter is also truncated to 40 characters.

  ☞ For more information, see "EngineName connection parameter [ENG]" [*ASA Database Administration Guide*, page 187].

- **Agent connection parameter** The Agent connection parameter behavior has been changed. The meaning of this parameter changed from version 5 to version 6, as the need for the *dbclient* executable was removed. The parameter meaning has changed to be more useful in a Version 6 environment.

  The Agent connection parameter is deprecated as of version 8.0.1.

SQL Remote behavior changes

The following behavior has changed in SQL Remote version 6.0.3:

- **Message link parameters stored in the database** By default, the message link parameters are now moved into the database when the Message Agent is run for the first time with the new version of the software. If you have software that explicitly accesses these parameters in their old locations external to the database, it will be affected by this change. You can continue using the old behavior by setting the External_remote_options database option to ON.

- **Passwords stored** When a password is entered for a message link, it was not stored in previous versions of the software. As the parameters are now held in the database, a saved password is not held on disk and so is more secure. Passwords are now saved by default. You can continue using the old behavior by setting the Save_remote_passwords option to OFF.

CHAPTER 11

What's New in Version 6.0.2

About this chapter

This chapter provides an overview to the new features and behavior changes introduced in Adaptive Server Anywhere 6.0.2.

Contents

New features in version 6.0.2

This section lists the new features introduced in components of SQL Anywhere Studio version 6.0.2.

Adaptive Server Anywhere new features

In addition to bug fixes, Adaptive Server Anywhere version 6.0.2 includes new features in both the software and the documentation.

Cross references
The printed documentation is not necessarily updated with each maintenance release. Cross references in this section may not be valid in the printed documents. For current information, see the online documentation.

- **UltraLite deployment option** UltraLite databases for small devices such as the PalmPilot and Windows CE computers can be developed with this version of the software.

  ☞ For information, see the book *UltraLite Developer's Guide.*

- **Backup and Restore SQL statements** Adding BACKUP and RESTORE as SQL statements provides server side backup and automation of backups using SQL scripts.

  The BACKUP statement provides direct backup to tape.

  ☞ For more information, see "BACKUP statement" [*ASA SQL Reference,* page 263].

- **Security features** New security features have been added.
  - **Auditing** Database administrators can keep track of activity performed on a database by turning on the AUDITING option. The record of activities is kept in the transaction log. By turning on auditing, you increase the amount of data saved in the transaction log to include login attempts, accurate timestamps of all events, all permissions checks, and all actions requiring DBA authority.

    ☞ For more information, see "Auditing database activity" [*SQL Anywhere Studio Security Guide,* page 9].
  - **Minimum password length** Database administrators can specify a minimum password length, to discourage easily discovered passwords.

    ☞ For more information, see "MIN_PASSWORD_LENGTH option [database]" [*ASA Database Administration Guide,* page 608].

- **Locating servers** A utility is provided for troubleshooting connections.

  ☞ For more information, see "The Ping utility" [*ASA Database Administration Guide,* page 514].

- **Starting databases from jConnect connections** Database connections over TDS, including connections from Java applications over jConnect, can start a database on a server.

  ☞ For more information, see "Supplying a URL for the server" [*ASA Programming Guide,* page 112].

- **getObject and setObject methods from Java client applications** From Java applications, you can now retrieve objects from a table and insert objects into a table without the explicit serialization and deserialization required in previous releases.

- **ODBC 3.51** The ODBC driver has been updated to ODBC 3.51. This version of ODBC includes support for Unicode applications.

  ☞ For more information, see "ODBC conformance" [*ASA Programming Guide,* page 228].

- **Control of allowed JOIN syntax** In previous releases, some multi-table queries have been allowed that have ambiguous join clauses. In the present release, you can set an option to disallow such queries.

  ☞ For more information, see "EXTENDED_JOIN_SYNTAX option [database]" [*ASA Database Administration Guide,* page 592].

- **Administration utility enhancements** Options have been added to the administration utilities to provide additional features.
  - **Transaction Log [dbtran] utility** If you use the new `-d` option, *dbtran* puts each operation as it occurs in the transaction log file. This makes transaction log output easier to read. This has been added primarily for auditing purposes.

    ☞ For more information, see "Translating a transaction log using the dbtran command-line utility" [*ASA Database Administration Guide,* page 508].
  - **Transaction Log [dbtran] utility** You can run *dbtran* against a running database server instead of against a log file. This feature has been added to increase the security of the transaction log—there is now no need to access the transaction log directly.

    ☞ For more information, see "Translating a transaction log using the dbtran command-line utility" [*ASA Database Administration Guide,* page 508].
  - **Log Transfer Manager [dbltm] utility logging** New options allow you to tune message logging from these utilities.

    ☞ For more information, see "The Log Transfer Manager" [*ASA Database Administration Guide,* page 501].

- **New Log Transfer Manager [dbltm] utility options** New options enable you to replicate only backed up transactions (**backup_only**), and to shut down as soon as all data is replicated (**continuous**).

  ☞ For more information, see "The LTM configuration file" [*ASA Database Administration Guide*, page 503].

SQL Remote new features

In addition to bug fixes, SQL Remote version 6.0.2 includes the following new features:

- **Performance enhancements** A major enhancement of the Adaptive Server Anywhere Message Agent (*dbremote*) operational model for scanning the transaction log and sending messages greatly improves the range of achievable replication turnaround times.

  Minimum lag times between entering data at one site and its replication to another site were limited in earlier versions to times on the order of ten minutes. With the new operational model, minimum lag times on the order of seconds can be achieved in some circumstances.

  When the Message Agent message-sending process runs in continuous mode, it now stays (**hovers**) at the end of the active transaction log while waiting for more data to be committed, instead of rescanning the transaction log each time. This allows you to poll more frequently, which can significantly reduce time for replication.

  ☞ For more information, see "Tuning Message Agent performance" [*SQL Remote User's Guide*, page 228].

- **SQL Remote message logging** New options allow you to tune message logging from these utilities.

  ☞ For more information, see "The Message Agent" [*SQL Remote User's Guide*, page 292].

Behavior changes in version 6.0.2

This section lists behavior changes in the components of SQL Anywhere Studio.

Adaptive Server Anywhere behavior changes

The following are behavior changes from previous versions of the software.

- **Permissions required to debug Java** In order to use the Java debugger, you must either have DBA authority, or be granted membership in the SA_DEBUG group. The SA_DEBUG group does not exist in databases created prior to 6.0.2, and in these older databases any user can use the Java debugger. The SA_DEBUG group was added to close a potential security hole.

  ☞ For more information, see "Requirements for using the debugger" [*ASA SQL User's Guide*, page 675].

- **Default packet size change** The default packets size for client/server communications has been changed from 512 bytes to 1000 bytes. This change improves performance for multi-row fetches and fetches of large rows. It also increases the memory requirements.

  ☞ For more information on packet size, see "CommBufferSize connection parameter [CBSIZE]" [*ASA Database Administration Guide*, page 178], and "-p server option" [*ASA Database Administration Guide*, page 154].

CHAPTER 12

What's New in Version 6.0.1

About this chapter

This chapter provides an overview to the new features introduced in Adaptive Server Anywhere 6.0.1

Contents

New features in version 6.0.1

This section introduces the new features in Adaptive Server Anywhere version 6.0.1. It provides a listing of major new features, with cross references to locations where each feature is discussed in detail.

Adaptive Server Anywhere for Windows CE

The Microsoft Windows CE operating system developed for handheld computing devices and embedded devices custom-built to carry out a specific task.

Starting with Version 6.0.1, Adaptive Server Anywhere is available for Windows CE. The Windows CE version of Adaptive Server Anywhere has the following characteristics:

- **Full-featured database** All SQL features in other versions of Adaptive Server Anywhere are available in the Windows CE version, including transaction processing, referential integrity actions, procedures and triggers, and so on.

  The Java features and the remote data access features are not available in Windows CE.

- **Administer from your desktop** When running Windows CE on a device that can be attached to a network or directly to a PC, you can administer your Windows CE database from a Sybase Central running on the PC.

- **ODBC and Embedded SQL applications** You can use either of these interfaces to develop client applications.

- **SQL Remote replication** The SQL Remote file link is implemented to be compatible with Windows CE ActiveSync synchronization.

Remote data access

Remote data access gives you access to data on external data sources, as if they were stored on the local database.

☞ For information about remote data access, see the chapter “Accessing Remote Data” [*ASA SQL User’s Guide*, page 557] and the chapter “Server Classes for Remote Data Access” [*ASA SQL User’s Guide*, page 589].

Character set translation

Character set translation has been added to translate strings automatically between different character sets as data is passed between client applications

and the database server. This enables more flexibility in mixed character-set environments.

Character set translation can be carried out among character sets that represent the same characters, but at different values. There needs to be a degree of compatibility between the character sets for this to be possible. For example, character set translation is possible between EUC-JIS and Shift-JIS character sets, but not between EUC-JIS and OEM code page 850.

To enable character-set translation, you must start the database server using the new `-ct` option. For example:

```
dbeng9 -ct asademo.db
```

Most of the character set translation features occur automatically, with little user intervention required.

☞ For a description of character set translation features, see "Starting a database server using character set translation" [*ASA Database Administration Guide*, page 329]. For more detailed information, see "Understanding character set translation" [*ASA Database Administration Guide*, page 317].

New Java features

There are some changes made to the Java support. These include the following:

- **Updating Java fields** You can now use standard UPDATE syntax to update a field in a column of Java data type. The following statement was not supported before 6.0.1, but is now supported:

  ```
  UPDATE Product
  SET JProd>>unit_price = 16.00
  WHERE ID = 302
  ```

  Use of the EVALUATE function is no longer required.

- **Compressed jar files** You can now install compressed jar files and zip files into the database. However, you should not use the *jar* utility that comes with the Sun JDK. Other zip utilities do produce suitable files.

- **Result sets from Java procedures** You can wrap Java methods in a stored procedure, which can return a result set or multiple result sets to the calling environment.

  ☞ For information on this feature, see "Returning result sets from Java methods" [*ASA Programming Guide*, page 94].

- **Default internal connection** When a database connection is established for internal JDBC operations, it is now recommended that you use the

following URL:

```
jdbc:default:connection
```

In version 6.0.0, an empty string was used to establish this connection. While the empty string does still work, it is deprecated. The new URL corresponds to the SQLJ1 proposed standard.

Additional new features

Several other features have been added to Adaptive Server Anywhere 6.0.1. These include the following:

- **jConnect 4.0** The version of jConnect included in this product has been updated to version 4.0.
- **AutoStart connection parameter** This parameter prevents a personal server from starting if no network connection is successful.

  ☞ For a description, see "AutoStart connection parameter [ASTART]" [*ASA Database Administration Guide*, page 176].
- **MESSAGE statement** Extensions to the MESSAGE statement allow messages to be directed to the client, the server window, or a log file.

  ☞ For a description, see "MESSAGE statement" [*ASA SQL Reference*, page 496].
- **Message callbacks** Windows Embedded SQL applications can handle messages received from the server while a request is being processed by registering a message callback function.

  ☞ For more information, see "Implementing request management" [*ASA Programming Guide*, page 201].
- **More control over operating system threads** A new database server option (`-gx`) controls the number of operating system threads that are in use. The existing `-gt` option controls how many can be in use at one time, effectively controlling the number of CPUs that can be exploited.

  ☞ For more information, see "The database server" [*ASA Database Administration Guide*, page 124].
- **Connection property system procedures** Two additional system procedures provide alternative ways of querying connection information.

  ☞ For more information, see "sa_conn_properties_by_conn system procedure" [*ASA SQL Reference*, page 711], and "sa_conn_properties_by_name system procedure" [*ASA SQL Reference*, page 712].

- **NULLIF function** This provides an abbreviated form of the CASE expression. NULLIF compares the values of the two expressions. If the first expression equals the second expression, NULLIF returns NULL. If the first expression does not equal the second expression, NULLIF returns the first expression. The NULLIF function provides a short way to write some CASE expressions.

  ☞ For more information, see "Miscellaneous functions" [*ASA SQL Reference*, page 88].

New features in SQL Remote

Several features have been added to SQL Remote.

- **Minimized Message Agent** The Message Agent can be made to start with a minimized window using the `-q` option.
- **Message Agent request to resend messages** The point at which the Message Agent requests that a missing message be resent is now user-configurable using the `-rp` option.

  ☞ For information on these options, see "The Message Agent" [*SQL Remote User's Guide*, page 292] and "Tuning incoming message polling" [*SQL Remote User's Guide*, page 230].
- **Cleaning the stable queue** For Adaptive Server Enterprise, the new `-fq` option on the Message Agent assists administration by cleaning confirmed messages from the stable queue.

  ☞ For information, see "The Message Agent" [*SQL Remote User's Guide*, page 292].

Behavior changes

This section describes behavior changes between version 6.0.0 and 6.0.1.

Java system table changes The system tables used to record Java class information (SYSJAR, SYSJARCOMPONENT, and SYSJAVACLASS) had SMALLINT primary keys. These data types have been altered to use INTEGER primary keys. This change allows more Java classes to be stored in a database, and more changes to the Java classes in the database.

This change takes effect for new databases and databases upgraded using the Upgrade [dbupgrad] utility from this or future releases.

Parentheses in -x command line In previous releases, database server command lines using the -x option have used the brace character ({) to surround options. The same applies to the string in the CommLinks connection parameter. For example:

```
dbsrv9 -x tcpip{Localhost=demo}
```

Existing command lines that use braces are still supported, but it is now recommended that you use parentheses to surround the options. Use of parentheses provides support for multi-byte character set identifiers in the database server command line and the CommLinks connection parameter.

```
dbsrv9 -x tcpip(Localhost=demo)
```

ANSINULL default for Transact-SQL and jConnect connections This has been changed to ON, which matches Adaptive Server Enterprise default behavior.

Database server -v option Prior to Version 6, this option produced verbose output to the transaction logs. This is obsolete, and `-v` is now used to supply version information.

Database server -gss option The behavior of the `-gs` server option, used to set the stack size, was complicated. The `-gs` option is now deprecated, and `-gss` provides the same functionality in a clearer way.

☞ For more information, see "The database server" [*ASA Database Administration Guide*, page 124].

Character set conversion in Interactive SQL Formerly, when the CHAR_OEM_TRANSLATION option was set to DETECT, Interactive SQL would fetch the collation label from the database to determine whether or not OEM to ANSI character set translation would be turned on. If the collation label started with a string that indicated an ANSI code page, translation would be turned off. Otherwise it would be turned on. When the

option was set to DETECT, Interactive SQL would display a message in the status window indicating the collation label of the database and the display translation setting.

The new behavior is as follows. If the option is set to DETECT, Interactive SQL will obtain the CharacterSet connection property from the server. This is the character set that the server is using for sending all character strings on this connection. If this character set indicates an ANSI code page, then OEM to ANSI translation is turned off. Otherwise it is turned on. A new message is displayed, indicating the collation label of the database, the character set used for communication over this connection, and the display translation setting.

CHAPTER 13

Upgrading Software and Databases

About this chapter

This chapter describes how to upgrade your software and databases.

There are additional concerns when upgrading Adaptive Server Anywhere version 5 applications. For more information, see “Upgrading Version 5 Applications” on page 191.

Contents

Upgrading Adaptive Server Anywhere

Running multiple versions

You can run multiple versions of Adaptive Server Anywhere (for example version 8.0 and version 9.0) simultaneously, provided that only one of the versions uses SPX.

Compatibility with existing software

Adaptive Server Anywhere database servers work with older databases. There is no need to upgrade your database in order to continue working with existing applications.

You should not use older versions of the Adaptive Server Anywhere database server with newer databases.

For Adaptive Server Anywhere software version 6.0.0 or later, you can upgrade the database server without upgrading client software.

- Version 9 database servers support connections from client applications using software from version 6.0.0 or later.
- Management of old databases and old database servers from the current version of Sybase Central is provided as follows:
  - Full management of version 7 and later databases running on version 7 and later servers.
  - You can connect to a version 5 or 6 database on a version 8 or later database server only to upgrade the database file format using the Unload Database wizard.
  - There is no support for version 6 and earlier databases running on version 7 and older database servers.

☞ For information on upgrading client/server installations from Version 5 and earlier, see "Upgrading Version 5 Applications" on page 191.

Reasons to upgrade

You must upgrade your database if you wish to use some of the new features in this version of the software.

There are two kinds of upgrade possible with version 9:

- **Upgrade the database** Use the Upgrade [dbupgrad] utility to upgrade your database. This process updates the system tables, provides new database options, adds or modifies system procedures.

  Upgrading the database can be done in-place on your existing database file. It does not require that you unload and reload your database.

  ☞ See "Upgrading a database" on page 177.

- **Upgrade the database file format** Some features, such as new index types, require a new database file format. The Upgrade [dbupgrad] utility does not change the format of the database file.

To upgrade the database file format, you must unload and reload your database.

☞ See "Upgrading the database file format" on page 178.

Upgrading a database

Upgrading a database adds and modifies system tables, system procedures and database options to enable version 9 features. It does *not* change the file format used to store and access data on disk and so does not give access to all new features and performance enhancements in the latest version of the software.

☞ For information on upgrading the database file format, see "Upgrading the database file format" on page 178.

❖ To upgrade a database (Sybase Central)

1. Carry out the standard precautions for upgrading software.

   ☞ See "Standard upgrade precautions" on page 181.

2. Start a version 9 database server running the database you wish to upgrade.
3. From Sybase Central, connect to the database you wish to upgrade.
4. Right click the database and choose Upgrade Database from the popup menu. The Upgrade a Database wizard appears.
5. Follow the instructions in the wizard.
6. Shut down the database and archive the transaction log before using the upgraded database.

❖ To upgrade a database (Command line)

1. Carry out the standard precautions for upgrading software.

   ☞ See "Standard upgrade precautions" on page 181.

2. Ensure that you have exclusive access to the database to be upgraded.
3. Ensure that the version 9 utilities are ahead of other utilities in your system path.

   ☞ See "Using the utilities" on page 182.

4. Execute the Upgrade [dbupgrad] utility against the database:

   ```
   dbupgrad -c "connection-string"
   ```

   The *connection-string* must connect to the database to be unloaded, with DBA authority.

   ☞ For more information, see "Upgrading a database using the dbupgrad command-line utility" [*ASA Database Administration Guide*, page 543].

5. Shut down the database and archive the transaction log before using the upgraded database.

❖ To upgrade a database (SQL)

1. Connect to the database from Interactive SQL or another application that can execute SQL statements.
2. Execute the ALTER DATABASE statement. Use the JAVA clause to specify whether you wish to upgrade the JDK level used by Java applications in the database.

   For example, the following statement upgrades a database without altering the JDK level:

   ```
   ALTER DATABASE UPGRADE
   ```

   ☞ For more information, see "ALTER DATABASE statement" [*ASA SQL Reference*, page 225].

Upgrading the database file format

This section describes how to use unload and reload your database to upgrade the file format of a database.

If you are upgrading the file format for a database that is involved in SQL Remote replication or that is a remote database in a MobiLink installation, and if you use the utility, you must be sure to use the `-ar` or `-an` option. The option ensures that the transaction log offsets for the new database are set to match those of the old database. In Sybase Central, following the instructions below ensures that the option is used.

When unloading and reloading a database that has proxy tables, you must create an external login to map the local user to the remote user, even if the user has the same password on both the local and remote databases. If you do not have an external login, the reload may fail because you cannot connect to the remote server.

Caution

Unloading and reloading a large database can be both time consuming and require a large amount of disk space. The process requires access to disk space approximately twice the size of your database to hold the unloaded data and the new database file.

☞ Users of version 5 who wish to unload and reload databases involved in replication should follow the instructions in "Unloading and reloading a database participating in replication" [*SQL Remote User's Guide*, page 258], rather than those in this section.

❖ To upgrade the database file format (Sybase Central)

1. Carry out the standard precautions for upgrading software.

   ☞ See "Standard upgrade precautions" on page 181.

2. Start a version 9 database server on the database you wish to upgrade. For example:

   ```
   dbeng9 mydatabase.db
   ```

3. Start Sybase Central.

   From the Start menu, choose Programs ➤ Sybase SQL Anywhere 9 ➤ Sybase Central.

4. Connect to the database you wish to upgrade.

   When you connect to a version 5.x or version 6.x database, a dialog appears asking you whether you want to use the Unload Database wizard to unload and reload your database. Click Yes. Once the wizard completes, the connection to the database is closed.

5. Right-click the database and choose Unload Database from the popup menu.

   The Unload Database wizard appears.

6. Read the text on the first page of the wizard and click Next.

7. Choose to unload the database to which you are connected. Click Next.

8. Specify the path and location for the unloaded database command file. The command file has the *.SQL* extension and is necessary to rebuild your database file from the files you unload.

   Unless space restrictions require otherwise, this location should be a directory on the same machine as the database server. Choose to unload structure and data.

   Click Next.

9. Specify the number of levels of view dependency.

   Specifying levels of view dependency allows you to recreate views based upon other views. For example, if you have one view based upon existing tables, you would enter the number 1 in this field. View one is independent and can be recreated from the tables alone. If, however, you have a second view based upon the first view, you would enter number 2 in this field. View 2 is dependent on view 1, and cannot be created until view 1 is created first.

   Click Next.

10. Specify whether the unloaded data is to be saved on the local machine or the server machine and select a location for the data.

    Specify whether you want to order the data. Exporting the data in an ordered format means that the data will be reloaded in an ordered format. This is useful if you want to improve performance of your database, or bypass a corrupted index.

    Click Next.

11. Check the box to create and reload into a new database file. Specify a new filename for the database.

    You can encrypt the database file if you wish. You need the encryption key each time you want to start the database.

    ☞ For more information about database file encryption, see "Encrypting a database" [*SQL Anywhere Studio Security Guide*, page 16].

12. Click Finish to start the process. You will need to examine the new database to confirm that the upgrade is completed properly.

❖ To upgrade the database file format (Command line)

1. Carry out the standard precautions for upgrading software.

   ☞ See "Standard upgrade precautions" on page 181.

2. Ensure that you have exclusive access to the database to be unloaded and reloaded. No other users can be connected.

3. Ensure that the version 9 utilities are ahead of other utilities in your system path.

   ☞ See "Using the utilities" on page 182.

4. Execute the Unload [dbunload] utility using the `-ar` option to create a new database.

   ```
   dbunload -ar -c "connection-string"
   ```

The *connection-string* must connect to the database to be unloaded with DBA authority. This command replaces the existing database with an upgraded database. To use the `-ar` option, you must connect to a personal server, or to a network server on the same machine as the Unload [dbunload] utility.

☞ For information on other Unload [dbunload] utility options, see "Unloading a database using the dbunload command-line utility" [*ASA Database Administration Guide*, page 534].

5. Shut down the database and archive the transaction log before using the reloaded database.

   If you want 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. For more information, see "Rebuilding databases" [*ASA SQL User's Guide*, page 539].

Standard upgrade precautions

There are several precautions you should take before upgrading any application, and these apply to Adaptive Server Anywhere upgrades just as to any other software.

- **Check the behavior changes** Before upgrading, you should confirm that none of the documented behavior changes in Version 9.0 affect your application.

  ☞ For more information, see "Adaptive Server Anywhere behavior changes" on page 37.

- **Test** You should test your application in a Version 9 environment thoroughly before upgrading any applications in production use.

- **Backup** You should back up your existing software and data before upgrading. In addition, as recovery cannot happen across a database upgrade, you should make a backup after upgrading to ensure recoverability going forward.

  For databases involved in synchronization, such as UltraLite databases or Adaptive Server Anywhere remote databases in MobiLink or SQL Remote installations, you should synchronize all outstanding changes before upgrading.

- **Test your upgrade procedure** If you are upgrading many end users, test your upgrade procedure carefully before carrying it out.

SQL Anywhere is used in so many different configurations that no upgrade guidelines can be guaranteed for all cases.

Using the utilities

If you have multiple versions of Adaptive Server Anywhere on your machine, you must pay attention to your system path when using utilities. Since setup adds the most recently installed version executable directory to the end of your system path, it's possible to install a new version of the software, and still inadvertently be running the previously installed version.

For example, if an Adaptive Server Anywhere Version 9 executable directory is ahead of the Version 9 executable directory in your path and you use the *dbinit* command, you will use the Version 7 utility, and consequently create a Version 7 database.

❖ To use the version 9.0 utilities

1. Change to the version 9.0 executable directory before executing your command,

   or

   Modify your system path so that the version 9.0 executable directory is ahead of any previous version Adaptive Server Anywhere executable directory.

Upgrading MobiLink

If you are upgrading an existing MobiLink installation, you must upgrade the components in the following order:

1. Upgrade the consolidated database.

   ☞ See "Upgrading your consolidated database" on page 183.

2. Upgrade the MobiLink synchronization server.

   ☞ See "Upgrading the MobiLink synchronization server" on page 185.

3. Upgrade the MobiLink clients.

   ☞ For information on Adaptive Server Anywhere remote databases, see "Upgrading Adaptive Server Anywhere MobiLink clients" on page 185. For information on UltraLite applications, see "Upgrading UltraLite applications" on page 187.

New MobiLink clients are incompatible with older versions of the MobiLink synchronization server, while the new MobiLink synchronization server is compatible with older clients.

Before upgrading, you should check for behavior changes that may affect you and carry out standard upgrade precautions.

☞ For more information, see "Behavior changes in version 8" on page 105, and "Standard upgrade precautions" on page 181.

Upgrading your consolidated database

You must upgrade the MobiLink system tables in your consolidated database before you can use the new MobiLink synchronization server.

If you are using an Adaptive Server Anywhere consolidated database, the usual upgrade process upgrades the MobiLink system tables—do not carry out the instructions in this section.

☞ For more information on upgrading Adaptive Server Anywhere databases, see "Upgrading a database" on page 177.

The upgrade procedure is slightly different depending on whether you are upgrading from MobiLink version 6 or MobiLink version 7.

❖ To upgrade a consolidated database (not DB2)

1. Only if you are upgrading from MobiLink version 6, create the MobiLink system tables and stored procedures by running the setup SQL script *MobiLink\setup\sync???.sql.*

2. Locate the upgrade script for your database-management system.

   The upgrade scripts are held in the *MobiLink/upgrade/6.0.x* or *MobiLink/upgrade/7.0.x* directory beneath the SQL Anywhere directory: The directory refers to the version of MobiLink from which you are upgrading.

3. Apply the upgrade script for your database management system.

   Use an administration or Interactive SQL [dbisql] utility for your database management system to apply the SQL script.

❖ To upgrade a DB2 consolidated database

1. Copy the *MobiLink\setup\SyncDB2.class* file (DB2 version 5) or *MobiLink\setup\SyncDB2Long.class* file (DB2 version 6 or later) to the *SQLLIB\FUNCTION* directory on the DB2 server machine. You probably need to restart the instance. For details, see your DB2 documentation.

2. Only if you are upgrading from MobiLink version 6, create the MobiLink system tables and stored procedures by running the setup SQL script *MobiLink\setup\syncdb2.sql.*

3. Locate the DB2 upgrade scripts.

   The upgrade scripts are held in the *MobiLink/upgrade/6.0.x* or *MobiLink/upgrade/7.0.x* directory beneath the SQL Anywhere directory: The directory refers to the version of MobiLink from which you are upgrading.

   IBM DB2 UDB version 5 allows only 18-character table and procedure names. Newer versions allow longer names. There are two upgrade scripts for DB2 databases:

   - *upgrade_db2tolong.sql* For use with DB2 version 6 and higher.
   - *upgrade_db2.sql* For use with DB2 version 5.

4. For DB2 UDB 5.x databases only:
   - Copy *upgrade_db2.sql* and modify the copy. Change the CONNECT statement at the start of the script so it will work with the instance you want to connect to.

- Apply the copied SQL script to the consolidated database.
  If an error occurs because the **ml_add_user script** is not defined when it is dropped, you may safely ignore the error.

5. For DB2 UDB 6 databases or higher only:
   - Copy *upgrade_db2tolong.sql* and modify the copy. Change the CONNECT statement at the start of the script so it will work with the instance you want to connect to.
   - Apply the copied SQL script to the consolidated database.
     If an error occurs because the **ml_add_user script** is not defined when it is dropped, you may safely ignore the error.

Upgrading the MobiLink synchronization server

Before using the version 9 MobiLink synchronization server instead of your earlier version, you should check the behavior changes to see if any affect you.

☞ For a list of behavior changes, see "MobiLink behavior changes" on page 112.

Upgrading Adaptive Server Anywhere MobiLink clients

In a production environment, you should only upgrade Adaptive Server Anywhere remote databases after you have upgraded the consolidated database and the MobiLink synchronization server.

There are several kinds of upgrade to consider:

- Upgrading the software.
- Upgrading the remote database itself.
- Upgrading the whole application.

Before any upgrade, you should synchronize all outstanding operations as a way of backing up the data in the remote database.

Upgrading the software

It is recommended that you upgrade the *dbmlsync* MobiLink client and the Adaptive Server Anywhere database engine at the same time. You must upgrade the remote database before running the new *dbmlsync* utility.

Version 9 MobiLink clients require a MobiLink version 9 synchronization server for synchronization. Version 9 MobiLink clients do not synchronize with a MobiLink synchronization server older than version 9. For information on upgrading MobiLink, see "Upgrading MobiLink" on page 183.

Upgrading the remote database

You can upgrade the Adaptive Server Anywhere remote database by running the Upgrade [dbugrad] utility or, for a more complete upgrade, you can upgrade the database file format.

Both of these operations can be carried out as described for Adaptive Server Anywhere databases. For instructions, see “Upgrading a database” on page 177, and “Upgrading the database file format” on page 178.

Upgrading applications

When deploying a new version of a MobiLink application, it is recommended that you use a new version name 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. This makes it easier to upgrade the remote databases incrementally, rather than all at once.

Upgrading UltraLite applications

To upgrade to version 9, you must rebuild your UltraLite application with version 9 software. This includes running the version 9 UltraLite generator and linking against the version 9 libraries.

To run the version 9 UltraLite generator, either explicitly include the path in your command line, or ensure that the version 9 directory appears ahead of older version directories in your system path.

To take advantage of added Palm code segmentation you must upgrade your reference database. Otherwise, all Palm code will go in the same segment. If you are developing for platforms other than the Palm Computing Platform, you do not need to upgrade your reference database to version 9 to generate a version 9 application.

Palm development

When working with version 8 or version 9 software, you must add a function call before calling **ULPalmLaunch** (embedded SQL) or **ULData.PalmLaunch** (C++ API), to indicate the storage type to use. The function calls are as follows:

```
ULEnablePalmRecordDB( &sqlca );
```

or

```
ULEnableFileDB( &sqlca );
```

Call **ULEnablePalmRecordDB** to use primary record-based storage, and **ULEnableFileDB** for secondary file-based storage. If the device does not support file-based storage, **ULPalmLaunch** sets SQLCODE -82.

Upgrading deployed databases

Before upgrading a deployed UltraLite database, you should ensure all changes are synchronized. For Palm applications, it is recommended that you use a new Creator ID for each version of your application, install a new application, and delete the old application and database from the Palm device. If you use the same Creator ID, you should delete the existing application first.

The UltraLite software can upgrade databases subject to some conditions. For more information, see “Altering the schema of UltraLite databases” [*UltraLite Database User’s Guide*, page 30], and “Upgrading UltraLite software” [*UltraLite Database User’s Guide*, page 31].

UltraLite version 9 applications require a MobiLink version 9 synchronization server for synchronization. UltraLite version 9 applications do not synchronize with a MobiLink synchronization server older than version 9. For information on upgrading MobiLink, see “Upgrading MobiLink” on page 183.

When deploying a new version of a MobiLink application, it is recommended that you use a new version name for the synchronization scripts. This makes it easier to upgrade the remote databases incrementally, rather than all at once.

Upgrading SQL Remote

If you are upgrading an existing SQL Remote installation from version 6 or later, you must upgrade each database server before or along with its message agent (*dbremote* and *ssremote*). You can upgrade message agents in any order.

☞ Version 5 users should follow the instructions in "Upgrading version 5 SQL Remote installations" on page 217 instead of this section.

- **No need to upgrade databases** Although it is not essential to upgrade databases, it is recommended for best performance that you 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.

  ☞ For information on upgrading your databases, see "Upgrading a database" on page 177, and "Upgrading the database file format" on page 178.

- **Software upgrades can be one site at a time** Older Message Agents can exchange messages with Version 9 Message Agents as long as the COMPRESSION database option is set to a value of -1 (minus one). There is no need to upgrade software throughout the installation simultaneously.

- **Message Agent and database server can be upgraded separately** The database server can be upgraded before the Message Agent. It is, however, recommended that you upgrade your Message Agent at the same time as the database server for performance reasons.

CHAPTER 14

Upgrading Version 5 Applications

About this chapter

The client/server communication protocol changed between SQL Anywhere versions 5 and Adaptive Server Anywhere Version 9. This chapter provides upgrade instructions for those users upgrading from Version 5 to a newer release of the software.

This chapter describes upgrade procedures that apply to the change in communication protocol for users of SQL Anywhere Versions 5.0 and 5.5, and users of Watcom SQL 4.0. In particular, it addresses the problem of stepwise upgrades of client/server installations.

Contents

Understanding version 5 upgrades

A database application and DBMS consists of several components. If your organization has a large SQL Anywhere installation, you may have many client machines, possibly running several applications, connected to more than one server.

When upgrading your system, you need to decide which components to upgrade, and in which order. This chapter guides you in making those choices and carrying out the upgrade.

SQL Anywhere Version 5
In this chapter, **SQL Anywhere Version 5** refers to both versions 5.0 and 5.5 of SQL Anywhere.

☞ In order to upgrade Version 5 applications, it is helpful to understand how connections work with later versions of the software. For information, see "Connecting to a Database" [*ASA Database Administration Guide*, page 37].

The components in your system

If you are currently running SQL Anywhere Version 5, you will have some or all of the following components in your system:

- **Application** Your application, aside from the SQL Anywhere components.
- **Connection parameters** SQL Anywhere Version 5.0 connection parameters. These may be assembled from an ODBC data source, or in some other way.
- **Driver manager** The ODBC driver manager, for ODBC applications.
- **ODBC driver** The SQL Anywhere Version 5 ODBC driver, for ODBC applications. For network applications, the ODBC driver is on the client machine.
- **Interface library** The SQL Anywhere Version 5 interface library is used by ODBC and embedded SQL applications. For network applications, the interface library is on the client machine.
- **SQL Anywhere client** The *dbclient.exe* executable and its command line, for network applications. The command line may specify the server name, and a set of network communications parameters. It may be stored in a batch file or an ODBC data source Start Line parameter. For network applications, the SQL Anywhere client is on the client machine.

- **The database server** The SQL Anywhere Version 5 database server. For network applications, this may be on a separate machine from the client components.
- **The database** A SQL Anywhere Version 5 database. This is on the same machine as the database server.

> **Database upgrades not required**
> You do not need to upgrade your database in order to use newer database server with an existing application.

Major upgrading issues

The major issues in upgrading from version 5 arise from the change in client/server communication protocol. This change means that the Version 5 interface library is not able to communicate with more recent database servers.

To help with this issue, Adaptive Server Anywhere includes a compatibility library. This DLL allows communications to both Version 5 and current database servers.

Behavior changes

You should also check the behavior change lists elsewhere in this book for versions since version 5 for any changes that may affect your application.

In addition, applications using Transact-SQL outer joins should note the following incompatibility with newer releases:

The null-supplying table in a Transact-SQL outer join cannot also participate in another regular or outer join. For example, in the following query, table S violates this limitation as it is the null-supplying table in `R.x *= S.x`, and participates in another join.

```
-- invalid query
SELECT *
FROM R, S, T
WHERE R.x *= S.x
AND S.y = T.y
```

When to upgrade your database

There is no need to upgrade your Version 5 database itself when you start using the current release of the software. You do need to upgrade the database if you wish to take advantage of many of the features introduced since Version 5.

☞ For more information on upgrading databases, see “Upgrading Software and Databases” on page 175.

Running more than one version of the software

The SQL Anywhere Studio software is designed so that both separate versions can be run if you install them in separate directories. This is the default behavior of the SQL Anywhere Setup program.

Review of SQL Anywhere Version 5 architecture

This section reviews the architecture for SQL Anywhere Version 5 applications connecting to a SQL Anywhere Version 5 database.

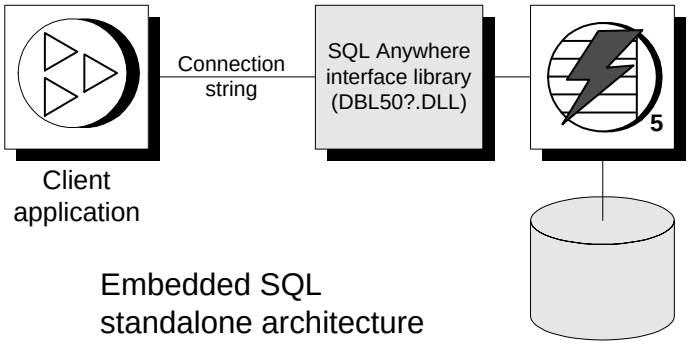
This information helps you to understand the changes needed when upgrading to more recent versions. If you are familiar with SQL Anywhere Version 5 architecture, you do not need to read this section.

Standalone components for Version 5 (embedded SQL)

If you are using SQL Anywhere Version 5 as a personal server, with an embedded SQL client application, you are using the following components on your machine:

- A SQL Anywhere Version 5 database.
- The SQL Anywhere Version 5 database engine (personal database server).
- The SQL Anywhere Version 5 interface library.
- A SQL Anywhere Version 5.0 connection string.

The following figure illustrates how these pieces fit together.



Embedded SQL
standalone architecture

Here, the question mark in *dbl50?.dll* represents a single character indicating the operating system. The interface library is named *dlb50t.dll* on Windows operating systems.

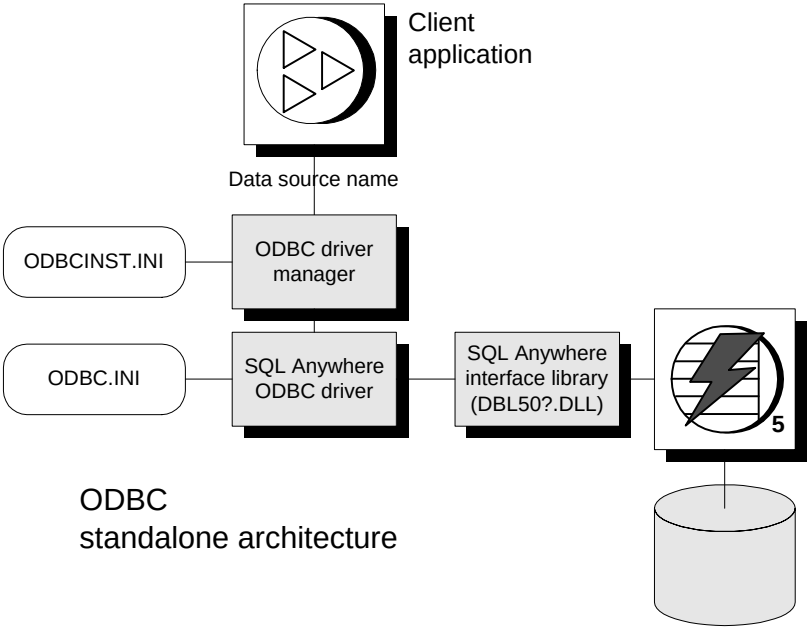
Embedded SQL client applications make calls to the interface library. The interface library is referenced by name, so that the name *dbl50t.dll* is a part of the client application.

Standalone components for Version 5 (ODBC)

If you are using SQL Anywhere Version 5 as a personal server, with an ODBC client application, you are using the following components on your machine:

- A SQL Anywhere Version 5 database.
- The SQL Anywhere Version 5 database engine.
- The SQL Anywhere Version 5 interface library.
- The SQL Anywhere Version 5 ODBC driver.
- The ODBC driver manager.
- A SQL Anywhere Version 5.0 connection description. This may be an ODBC data source, or a connection string from an application.

The following figure illustrates how these pieces fit together. The client application passes a data source name to the ODBC driver manager. The ODBC driver manager looks up the appropriate driver in ODBCINST.INI. The driver looks up the connection information in ODBC.INI and, via the interface library, connects to the SQL Anywhere Version 5 database engine.



ODBC standalone architecture

Client/server components for Version 5

If you are using SQL Anywhere as a network server, you have the following components on your server machine:

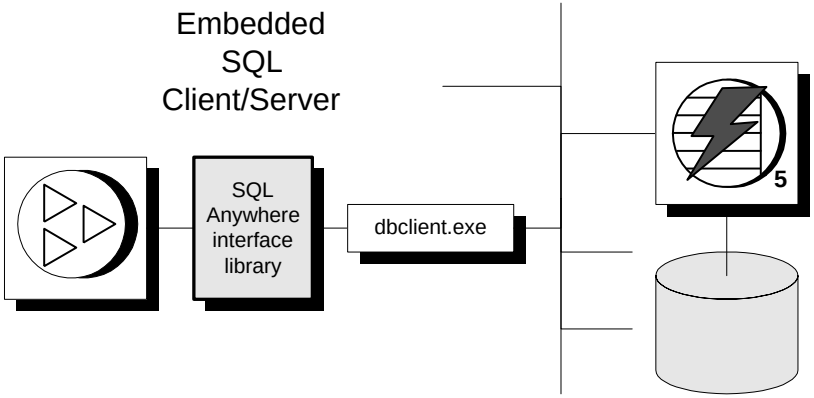
- A SQL Anywhere Version 5 database
- The SQL Anywhere Version 5 database server

You have the following components on your client machine:

- The SQL Anywhere Version 5 Client executable
- The SQL Anywhere Version 5 interface library.
- The SQL Anywhere ODBC driver, if your application uses ODBC.
- A SQL Anywhere Version 5.0 connection description. This may be an ODBC data source, or a connection string from an application.

  The data source may contain connection information in the start option, corresponding to a SQL Anywhere Client command line. You may also have connection strings in your application, and batch files that start a client with a particular set of parameters and options.

The architecture of a Version 5 embedded SQL client/server connection is illustrated in the figure. For ODBC applications the ODBC driver manager and ODBC driver stand between the application and the interface library.



Upgrading embedded SQL applications

You can upgrade a SQL Anywhere Version 5 standalone application to use newer versions of the database server by upgrading the database server and the interface library. You do not need to upgrade the database or the client application itself.

The upgrade procedure uses the **compatibility library**. The compatibility library is a dynamic library (a DLL on PCs, shared library on UNIX) that enables embedded SQL applications to work with both Version 5 and newer versions of the database server.

☞ For a description of the compatibility library, see "Using the compatibility library" on page 208.

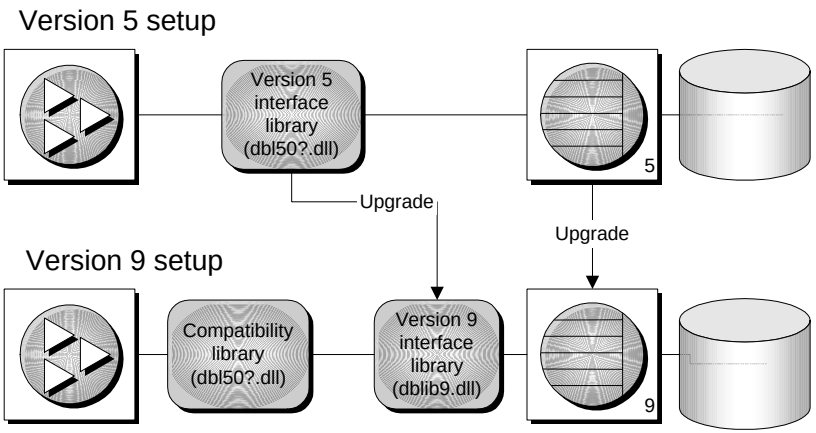
StartLine connection parameter

If your application connection string uses a StartLine parameter that explicitly provides a *dbclient.exe* or *dbeng50.exe* command line, and this is hard coded into your application, there are additional upgrade considerations.

For information on upgrading applications that use StartLine connection parameters, see "Start parameters and the compatibility library" on page 212.

Components upgraded

The following figure illustrates the changes you need to make in your setup in order to upgrade:



☞ For a description of the compatibility library, see "Using the compatibility library" on page 208.

Upgrade procedures for embedded SQL applications

This section provides step-by-step procedures for different kinds of embedded SQL applications:

- Applications using an embedded database are standalone applications using the personal database server (*dbeng50.exe* for Version 5, *dbeng9.exe* for Version 9).
- Client/server applications connect across a network to the network database server. In Version 5, these applications use the *dbclient.exe* executable.

❖ To upgrade embedded SQL applications that use an embedded database

1. **Install the current version of the software** Newer versions of the software contain components that enable Version 5 applications to continue working. You can either install into a separate directory or over the top of your Version 5 software.

   The installation places the new executable directory ahead of the Version 5 executable directory in your system path.

2. **Ensure that your application is using the compatibility DLL** If necessary, copy the compatibility library *dbl50?.dll* from your Adaptive Server Anywhere executable directory to a place where your application will locate it.

   For example, you could copy the compatibility library to the same directory as the module of your application that loads it. The file *dbl50?o.dll* is installed into your Adaptive Server Anywhere executable directory. This directory must be in your path as this library is required by the compatibility library.

   At this stage, your Version 5 application should continue to work as before. However, it will be connecting to your database through the compatibility library rather than directly through the Version 5 interface library.

   ☞ If you have any problems at this stage, you need to check how your application locates the interface library. For information, see "File locations and the compatibility library" on page 211.

3. **Create a new connection description** If your application obtains its connection parameters from configuration files, batch files, or the system registry, you should prepare a new description that uses the current version of the database server. For example, Sybase Central stores connection descriptions in the system registry.

- If you store a connection string that uses the DBF parameter to start the default database server, then the new database server is started automatically by the compatibility library, instead of the Version 5 database server. In this case, no new connection description is needed.
- If you store a connection string that uses a Start Line parameter specifying *dbeng50.exe*, you must replace this with one specifying *dbeng9.exe*. If your application contains a hard-coded connection string, you need to take extra steps at this point.

  ☞ For more information on upgrading StartLine parameters, see "Start parameters and the compatibility library" on page 212.
- If the database server is started in some other way, such as by a batch file or using a Windows NT service, you must reconfigure this so that the new version of the database server is started instead.

4. **Use the new connection description** With this step, you are using all current software and have completed your upgrade. The database itself does not need to be upgraded to work with existing applications.

❖ To upgrade embedded SQL client/server applications

1. **Prepare for the upgrade** This step must be carried out at each client machine. You prepare for the upgrade by installing the current version of the software.

   Ensure that the installation places the current executable directory ahead of the Version 5 executable directory in your system path.

2. **Ensure that your application is using the compatibility DLL** This step must be carried out at each client machine. If necessary, copy the compatibility library *dbl50?.dll* from your Adaptive Server Anywhere executable directory to a place where your application will locate it.

   For example, you could copy the compatibility library to the same directory as the module of your application that loads it. The file *dbl50?o.dll* is installed into your Adaptive Server Anywhere executable directory. This directory must be in your path as this library is required by the compatibility library.

   At this stage, your Version 5 application should continue to work as before. However, it will be connecting to your database through the compatibility library rather than directly through the Version 5 interface library.

   ☞ If you have any problems at this stage, you need to check how your application locates the interface library. For information, see "File locations and the compatibility library" on page 211.

3. **Create a new connection description** This step must be carried out at each client machine. If your application obtains its connection parameters from configuration files, batch files, or the system registry, you should prepare a new description that uses the newer database server. This description is for use when the server is upgraded.

   If you store a connection string that uses a StartLine connection parameter specifying *dbclient.exe*, you must replace this with a new one. The new connection description should contain all the dbclient information as a set of parameters. If your application contains a hard-coded connection string, you need to take extra steps at this point.

   ☞ For more information on upgrading StartLine parameters, see "Start parameters and the compatibility library" on page 212.

   ☞ For more information about creating connection descriptions that capture the dbclient command information, see "Capturing dbclient command information" on page 214.

4. **Upgrade the database server** This step must be carried out at the server machine.
   - As with any software upgrade, back up your database before upgrading.
   - Install Adaptive Server Anywhere on the server machine.
   - Start the new database server on the database.

5. **Use the new connection description** This step must be carried out at each client machine. You need to use the new connection description to connect to the newer version of the database server. With this step, you are using all current software and have completed your upgrade. The database itself does not need to be upgraded to work with existing applications.

Using the version 5 utilities with Adaptive Server Anywhere

For the Version 5 database utilities, connection strings are supplied interactively. The Version 5 database utilities such as ISQL are embedded SQL applications that search for the interface library in the following order:

1. The current directory
2. The Version 5 executable directory
3. The system path

For these applications, even though the compatibility library is ahead of the Version 5 interface library in the system path, the Version 5 compatibility library is located.

❖ To use Version 5 utilities with Adaptive Server Anywhere Version 9

1. Make a backup copy of your interface library file.
2. Copy the compatibility library from your Version 9 executable directory to your Version 5 directory. For example, on Windows 95 and Windows NT, copy the file *dbl50t.dll* from the *win32* subdirectory of your Version 9 installation to the *win32* subdirectory of your Version 5 installation.
3. You can now run your Version 5 utilities against both Version 5 and Version 9 database servers.

Upgrading ODBC applications

You can upgrade Version 5 ODBC applications in the following ways:

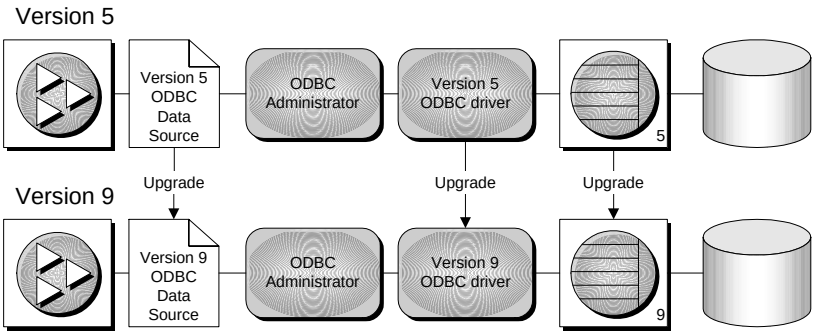
- Replace the Version 5 ODBC data source with a current ODBC data source. This approach is a complete upgrade, and is described in this section.
- Use the compatibility library to connect to a current version of the database server. If you choose this route, your application continues to use the Version 5 ODBC driver, so this is not a complete upgrade. The procedure for upgrading in this way is the same as for embedded SQL applications.

  ☞ For information about upgrading in this manner, see "Upgrading embedded SQL applications" on page 198.

StartLine parameter
If your ODBC data source contains a start line specifying the Version 5 standalone engine (*dbeng50.exe*) or the SQL Anywhere client (*dbclient.exe*), you cannot upgrade using the compatibility library.

Components upgraded

The following figure illustrates the changes you must make when upgrading an ODBC standalone application.



Data sources

The ODBC data source specifies which ODBC driver to use. When an ODBC data source is created as an Adaptive Server Anywhere Version 9 data source, it uses the Version 9 ODBC driver.

Data source must be current version
ODBC applications require the version of the ODBC data source to be the same as that of the ODBC driver.

Upgrade procedures for ODBC applications

This section provides step-by-step procedures for different kinds of ODBC applications:

- Applications using an embedded database are standalone applications using the personal database server (*dbeng50.exe* for Version 5, *dbeng9.exe* for Version 9).
- Client/server applications connect across a network to the network database server. In Version 5, these applications use the *dbclient.exe* executable.

Some applications allow you to change the ODBC data source name you use. Other applications use a fixed data source name. You can upgrade either kind of application.

❖ To upgrade ODBC applications that use an embedded database

1. **Install the current software** The current version of the software contains components that enable Version 5 ODBC applications to continue working. You can either install into a separate directory or over the top of your Version 5 software.

   Your application should be unaffected by installation of the current version of the software.

2. **Create a Version 9 ODBC data source** The changes you make depend on the connection parameters you use in the data source.
   - If you start the default database server using the DBF parameter, you can use the same connection parameters in your new data source as your old one.
   - If you store a connection string that uses a Start Line parameter specifying *dbeng50.exe*, you must replace this with one specifying *dbeng9.exe*.

     ☞ For more information on upgrading StartLine parameters, see "Start parameters and the compatibility library" on page 212.
   - If the database server is started in some other way, such as by a batch file or using a Windows NT service, you must reconfigure this so that the newer version of the database server is started instead.

     ☞ For information on creating data sources, see "Working with ODBC data sources" [*ASA Database Administration Guide*, page 53].

3. **Use the new data source** With this step, you are using all current software and have completed your upgrade. The database itself does not need to be upgraded to work with existing applications.

Some applications may have the data source name hard wired. In this case, you need to replace the Version 5 data source with a Version 9 data source of the same name. It is recommended that you rename, rather than delete, your Version 5 data source.

❖ To upgrade ODBC client/server applications

1. **Prepare for the upgrade** This step must be carried out at each client machine. You prepare for the upgrade by installing the current version of the software.

   Your application should be unaffected by installation of the current version of the software.

2. **Create a Version 9 ODBC data source** This step must be carried out at each client machine. The changes you make depend on the connection parameters you use in the data source.
   - If you start the default database server using the DBF parameter, you can use the same connection parameters in your new data source as your old one.
   - If you store a connection string that uses a StartLine connection parameter specifying *dbclient.exe*, you must replace this with a new one. The new connection description should contain all the *dbclient* information as a set of parameters. If your application contains a hard-coded connection string, you need to take extra steps at this point.

     ☞ For more information on upgrading StartLine parameters, see "Start parameters and the compatibility library" on page 212.

     ☞ For more information about creating connection descriptions that capture the dbclient command information, see "Capturing dbclient command information" on page 214.
   - If the database server is started in some other way, such as by a batch file or using a Windows NT service, you must reconfigure this so that the newer version of the database server is started instead.

     ☞ For information on creating Version 9 data sources, see "Working with ODBC data sources" [*ASA Database Administration Guide*, page 53].

3. **Upgrade the database server** This step must be carried out at the server machine.
   - As with any software upgrade, back up your database before upgrading.
   - Install Adaptive Server Anywhere on the server machine.
   - Start the Version 9 database server on the database.

4. **Use the new data source** This step must be carried out at each client machine. You need to use the new connection description to connect to the newer version of the database server. With this step, you are using all current software and have completed your upgrade. The database itself does not need to be upgraded to work with existing applications.

Upgrade notes for PowerBuilder and InfoMaker users

Users of Sybase PowerBuilder and InfoMaker should make some changes in order to obtain full functionality with Adaptive Server Anywhere Version 9.

The pbodb80.ini file

PowerBuilder and InfoMaker use a file named *pbodb80.ini* to hold ODBC data source information. The 80 in the file name may be different, depending on the version you have. For each ODBC driver it provides such things as DDL syntax, default DBParm options, valid function names and special data types.

If your *pbodb80.ini* file does not have a Adaptive Server Anywhere section, PowerBuilder and InfoMaker default to a core syntax. This limits the operations you can carry out using these tools. For example you cannot create, alter, or drop primary and foreign keys.

Upgrading your pbodb80.ini file

To obtain complete functionality with PowerBuilder and InfoMaker, you need to upgrade your *pbodb80.ini* file.

❖ **To upgrade your pbodb80.ini file**

1. Make a backup copy of your existing *pbodb80.ini* file.
2. Add an Adaptive Server Anywhere section to the working copy of the file containing the same information as the existing Sybase SQL Anywhere section:

```
[Adaptive Server Anywhere]
PBSyntax='WATCOM50_SYNTAX'
PBDateTime='STANDARD_DATETIME'
PBFunctions='WATCOM_FUNCTIONS'
PBDefaultValues='autoincrement,current date,current
        time,current timestamp,timestamp,null,user'
PBDefaultCreate='YES'
PBDefaultAlter='YES'
PBDefaultExpressions='YES'
DelimitIdentifier='YES'
PBDateTimeInvalidInSearch='NO'
PBTimeInvalidInSearch='YES'
PBQualifierIsOwner='NO'
PBSpecialDataTypes='WATCOM_SPECIALDATATYPES'
IdentifierQuoteChar='"'
PBSystemOwner='sys,dbo,rs_systabgroup'
PBUseProcOwner='YES'
SQLSrvrTSName='YES'
SQLSrvrTSQuote='YES'
SQLSrvrTSDelimit='YES'
ForeignKeyDeleteRule='Disallow if Dependent Rows Exist
        (RESTRICT),Delete any Dependent Rows (CASCADE),Set
        Dependent Columns to NULL (SET NULL)'
TableListType='GLOBAL TEMPORARY'
```

Using the compatibility library

The compatibility library is a dynamic library (a DLL on PCs, shared library on UNIX) that enables embedded SQL applications to work with both Version 5 and Version 9 database servers. This section describes how the compatibility library works.

Who needs to read this section?
You should read this section if you are upgrading SQL Anywhere Version 5 to Adaptive Server Anywhere Version 9, and have existing embedded SQL applications that you need to work with the Version 9 server.

The Version 5 embedded SQL interface library

All client machines running SQL Anywhere Version 5 applications, whether connecting over a network or to a personal server, have a SQL Anywhere Version 5 interface library.

For Windows operating systems, this library is a DLL named *dbl50t.dll.*

How Version 5 client applications locate the interface library

Version 5 client applications locate the interface library in one of the following ways:

- **ODBC applications** ODBC applications connect to a SQL Anywhere database using the SQL Anywhere ODBC driver. The SQL Anywhere Version 5 ODBC driver calls functions in the Version 5 embedded SQL interface library.
- **Embedded SQL applications** Embedded SQL applications for Windows operating systems call into the interface library. In these calls, the interface library is referenced by name—*dbl50t.dll* for Windows 95 and NT.

  The SQL Anywhere Version 5 ODBC driver is an embedded SQL application.

Using the compatibility library

The compatibility library is *optionally* installed as part of the Version 9 client software. It provides support for two interface libraries at the same time. You should check your installation to confirm that it is installed.

The setup program should ensure that your application calls the compatibility library instead of the Version 5 interface library.

The setup program carries out the following steps to ensure that applications call the interface library.

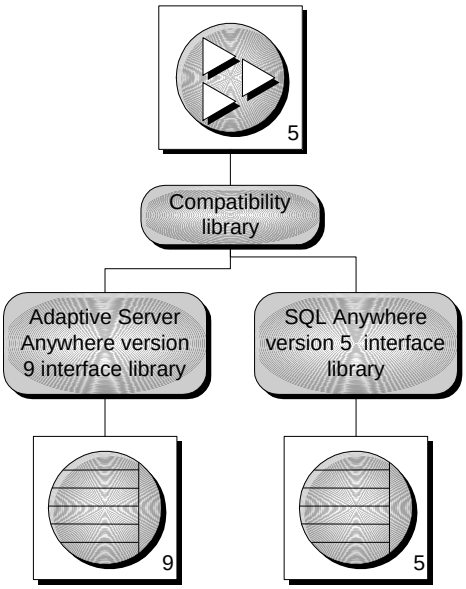
- The compatibility library has the same name as the SQL Anywhere Version 5 interface library. For example, on Windows NT, the compatibility library is named *dbl50t.dll.*
- The compatibility library is installed into the same directory as other Version 9 software.
- The Version 9 installation directory is placed ahead of the Version 5 directory in the system path. This ensures that applications locate the compatibility library ahead of the Version 5 interface library.
- A Version 5 interface library is installed into the same directory as the compatibility library, but with the name *dbl50to.dll.* When the compatibility library is accessing Version 5 servers, it calls this interface library.
- The Version 9 interface library is installed into the same directory as the compatibility library. It has the name *dblib9.dll* on Windows NT. When the compatibility library is accessing Version 9 servers, it calls this interface library.

If you have problems using the compatibility library, you should check the order of the directories in your path, and ensure that the Version 9 location is ahead of the Version 5 location in the path.

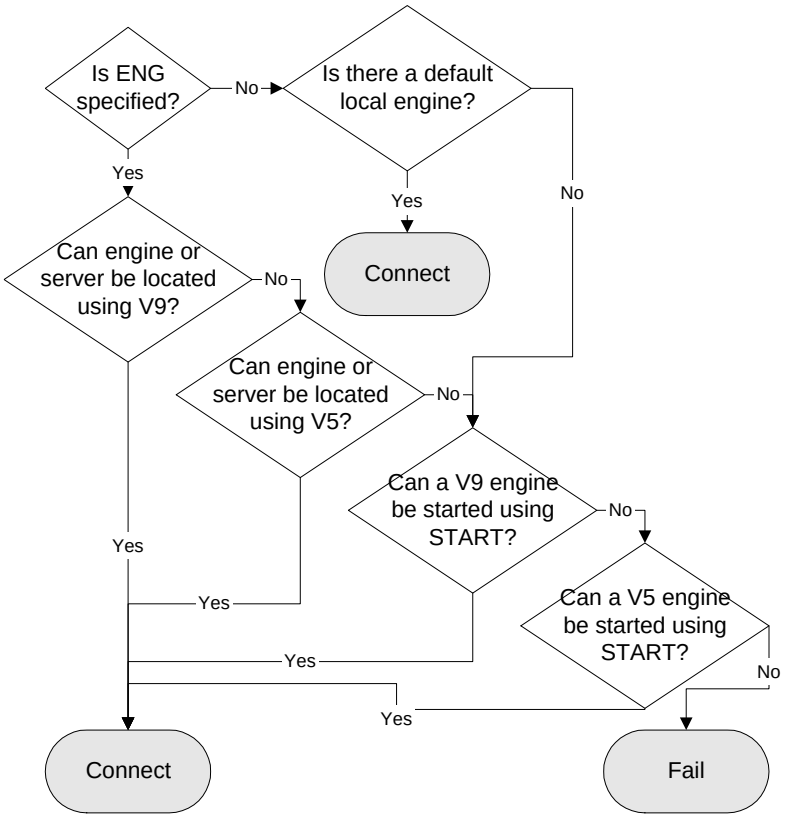
How the compatibility library works

Using the supplied connection string, the compatibility library attempts to connect to an Adaptive Server Anywhere database using the Adaptive Server Anywhere Version 9 interface library. If this attempt fails, it attempts to connect to a SQL Anywhere database using the SQL Anywhere Version 5 library.

The following figure illustrates how the compatibility library enables communications to both a SQL Anywhere Version 5 and an Adaptive Server Anywhere Version 9 database server. The number in the lower right hand corner of the boxes indicates the version of the software component.



The following figure illustrates the algorithm used by the compatibility library to connect to a server:



File locations and the compatibility library

The compatibility library and the Version 5 interface library have the same file name (*dbl50.dll*). For your application to use the compatibility library, it must locate it ahead of the Version 5 interface library when it searches for DLLs.

To ensure that your application locates the compatibility library ahead of the Version 5 interface library you must understand how your application searches for DLLs.

Searching for DLLs

The Version 9 installation program does not necessarily place the Version 9 executable directory ahead of the Version 5 directory in the system path, so any application that uses the path to searching for *dbl50.dll* may have to be changed to find the compatibility library ahead of the Version 5 interface library.

The Version 5 ODBC driver is in the same directory as the Version 5

interface library, and so locates this library instead of the Version 9 library. To enable Version 5 ODBC applications to connect to Version 9 database servers, you can either rename the Version 5 interface library, or copy the Version 9 compatibility library along with *dbl50to.dll* into your Version 5 directory.

Testing to see which library is located

You can test to see which library is located in the following ways:

- Attempt to connect to a Version 9 database server. You cannot connect using the Version 5 interface library.
- Specify a database file parameter (DBF) and no start line in your connection string. If the Version 5 interface library is located, the SQL Anywhere Version 5 standalone database engine is started. If the compatibility library is located, the Version 9.0 personal database server is started.

Start parameters and the compatibility library

Applications using a connection string that includes a StartLine connection parameter face some additional issues in upgrading.

The StartLine parameter provides explicit instructions for starting the database engine or the SQL Anywhere Client executable. Sample StartLine parameters are as follows:

- **Standalone application** A sample StartLine parameter for a Version 5 standalone application is as follows:

  ```
  dbeng50.exe -c 8M
  ```

- **Network client application** A sample StartLine parameter for a Version 5 network client application is as follows:

  ```
  dbclient.exe -x tcpip
  ```

In the current version of the software, the *dbeng50.exe* executable is replaced by the personal database server *dbeng9.exe*. The *dbclient.exe* executable is no longer required.

❖ To upgrade StartLine parameters

1. The procedure depends on where your connection parameters are stored.
   - If your connection parameters are stored outside the application itself, then you need to alter the connection parameters to use the appropriate *dbeng9.exe* executable name instead of *dbeng50.exe*.

     All the information on the dbclient command line can be rephrased in terms of other connection parameters. For information, see

"Connection and Communication Parameters" [*ASA Database Administration Guide*, page 173].

- If your connection parameters are hard-wired into your application, you must relink your application with a new connection string.

  There are many possible configurations of client command lines and connection parameters. Be sure you test any solution thoroughly before deploying.

Capturing dbclient command information

Version 5 client applications that connect to a database server do so via the SQL Anywhere Client, an executable named *dbclient.exe*. The client executable command line contains information needed to locate a server, including the following:

- **Default server name** The server name on the client command line is the default server name. When a client executable is running, the application does not need to supply a server name in order to connect to the default server.
- **Network communications parameters** A listing of network protocols to use together with a set of communications parameters specifies where the client executable is to look as it attempts to locate a server.
- **Client/Server communication tuning** A set of parameters allows the packet size, buffer size, and so on to be tuned for optimum performance.

In Version 9, this information is held in an ODBC data source along with other connection information. As there is no longer a client executable, there is no longer a client command line. In Version 9, embedded SQL applications can use ODBC data sources as a source of connection parameters.

Client command line scope

Only one Version 5 client executable can be run at a time, and it may be used by more than one application and handle connections to more than one server. The command information is therefore global to the machine.

How to capture client command information

During upgrade to Version 9, you must ensure that Version 5 *dbclient* command information is captured in such a way that the Version 9 ODBC or embedded SQL applications can use it. You can do this in one of the following ways:

- **Place the information in an ODBC data source** If the information can be placed in a data source, you can use it with Version 9 ODBC or embedded SQL applications.
- **Use the SQLCONNECT environment variable** The SQLCONNECT environment variable contains a connection string. It is searched early in the process of establishing which connection parameters to use. You may be able to use SQLCONNECT settings to override connection parameters.

The way to capture client command information depends on where the information is located.

Where command information is located

Your existing command line information may be held in one of the following places.

- **ODBC data source** The ODBC data source contains a START parameter that can hold a client executable command.
- **As a connection string** Your application may obtain client information (for example from an initialization file), and supply it in a connection string as the START parameter.
- **A batch file** You may have a batch file that includes a client executable command line as part of your startup process.
- **Under an icon** You may have a client executable command line under an icon on your desktop.

How to capture the information

- **From an ODBC data source** The ODBC data source upgrade captures the information in an ODBC data source START parameter.
- **From a batch file or under an icon** You can move the *dbclient.exe* parameters into your CommLinks connection parameter.
- **Hard-wired connection strings** Only if you have a hard-wired connection string in your application (that is, one that cannot be edited), you must alter the source of the application and recompile.

Upgrading databases

To use some of the new features of Adaptive Server Anywhere, you need to upgrade your database file.

> ***Caution***
> *Ensure that you back up your database before upgrading.*

Supported versions

You can upgrade your database from any of the following versions of the software to the format of the current version:

- **SQL Anywhere Version 5** Including versions 5.0 and 5.5, all patch levels.
- **Watcom SQL Version 4** All patch levels.
- **Watcom SQL Version 3.2** Only patch level **e** and above.

What the Upgrade [dbupgrad] utility does

The Upgrade [dbupgrad] utility carries out the following tasks:

- Adds new system tables.
- Adds new columns to existing system tables.
- Adds database options.

☞ For instructions on upgrading databases, see “Upgrading Software and Databases” on page 175.

Upgrading version 5 SQL Remote installations

SQL Remote installations include a consolidated database and many remote databases, together with a Message Agent at each site.

At each site, the Message Agent handles the sending and receiving of messages. The messages take the form of SQL statements, and the database server handles the actual execution of those SQL statements.

The upgrade requirements for SQL Remote are as follows:

- **No need to upgrade databases** Although it is not essential to upgrade databases for Version 9, it is recommended for best performance that you 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.

  ☞ For instructions on unloading and reloading the database, see "Unloading and reloading a database participating in replication" [*SQL Remote User's Guide*, page 258]. For instructions on upgrading the database version, see "Upgrading a database" on page 177.

- **Software upgrades can be one site at a time** Version 5 Message Agents can exchange messages with Version 9 Message Agents as long as the COMPRESSION database option is set to a value of -1 (minus one). There is no need to upgrade software throughout the installation simultaneously.

- **Message Agent and server can be upgraded separately** The Message Agent is an embedded SQL application. Therefore, the database server can be upgraded before the Message Agent as long as the compatibility library is used. It is, however, recommended that you upgrade your Message Agent at the same time as the database server for performance reasons.

  The Message Agent cannot be upgraded before the database server, as a new client application cannot work with a Version 5 server.

  Replication is based on the transaction log, and when a database is unloaded and reloaded, the old transaction log is no longer available. For this reason, good backup practices are especially important when participating in replication.

Example

One approach to upgrading is as follows:

1. Upgrade the consolidated database server and Message Agent. Set the COMPRESSION database option to -1 so that all messages are compatible with the Version 5 software at remote sites.

2. Over time, upgrade remote database servers and Message Agents. 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.

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y