



SQL Anywhere 11 and Microsoft .NET

A whitepaper from Sybase iAnywhere

Date: 2008-10-20

**This whitepaper was written in the context of SQL Anywhere 11.
However, its content may be applicable to previous and future releases.**

CONTENTS

Introduction	3
Understanding the .NET Framework	3
SQL Anywhere integration points with .NET	3
XML and Relational Databases	4
Importing XML Into Relational Databases	4
Exporting XML from Relational Databases	5
Storing XML in Relational Databases	5
Querying XML	5
XML Web Services	5
ADO.NET and .NET Data Providers	6
The ADO.NET Programming Interface	6
CLR Stored Procedures and Functions	8
MobiLink .NET support	8
MobiLink Synchronization Scripts	8
Direct Row API	8
MobiLink .NET API for DBMLSync	8
Summary	9

INTRODUCTION

The purpose of this whitepaper is to outline the integrated Microsoft .NET support features of SQL Anywhere. While not compulsory, this paper assumes familiarity with the Microsoft .NET framework and its components.

UNDERSTANDING THE .NET FRAMEWORK

The .NET Framework is A programming model, which is designed to simplify application development in distributed environments. There are two main components to the .NET Framework:

- **Common Language Runtime:** A managed, protected application execution environment. The runtime manages code when it is executed and handles tasks such as memory and thread management. A virtual machine executes the supported .NET languages, including C#, Visual Basic, and C++.
- **Common Class Library:** A library of common classes that is available across all supported languages. Previous to .NET, there were different Windows APIs for each language.

The .NET Compact Framework from Microsoft is a subset of the .NET Framework for smart devices. It delivers managed code to smart devices, and also allows you to run secure, downloadable applications on Microsoft supported devices such as personal digital assistants (PDAs) and mobile phones.

When it comes to data management software such as SQL Anywhere, "supporting .NET" really means two things:

- Programmability using the .NET Framework and programming environments, and ADO.NET (the common database API in .NET) in particular.
- XML support. In particular, making data available in XML format and making functions available as Web services.

SQL ANYWHERE INTEGRATION POINTS WITH .NET

The 11.0.0.x releases of SQL Anywhere provide a wide range of support for .NET. Supported features include:

- SQL Anywhere allows you to export XML and store XML data in your database.
- SQL Anywhere allows you to expose your data via an XML web service.
- SQL Anywhere supports the ADO.NET interface using the SQL Anywhere .NET Data Provider, as well as the standard .NET OLE DB and ODBC Data Providers.

- SQL Anywhere allows you to use any supported .NET language (version 2.0) to write your stored procedures and functions.
- MobiLink allows you to use any supported .NET language to write your synchronization scripts.
- MobiLink exposes a .NET API for its synchronization client (DBMLSync).

The following sections outline these features in more detail, as well as explaining features and options that may be added to later releases of SQL Anywhere to enhance its .NET support.

XML AND RELATIONAL DATABASES

Extensible Markup Language (XML) is a markup language for representing structured data in a text format. XML is designed to be a simple markup language, like HTML, but also to be flexible, like SGML. There is no specific format for data as XML, and the legal tag names and structure can be re-defined for each application. XML documents use a document type definition (DTD) or XML schema to define the structure, elements, and attributes that are used in an XML file.

.NET uses XML as a universal data-exchange format. The XML-based Simple Object Application Protocol (SOAP) is used by .NET as a mechanism for inter-process communication.

There are several ways in which XML and relational databases interact:

- Importing XML into relational databases
- Exporting XML from relational databases
- Storing XML documents in the database
- Performing XML queries
- Creating XML views
- Accessing Web services

Current releases of SQL Anywhere support exporting XML and storing XML documents in your SQL Anywhere database. There are also several procedures and functions provided for reading XML documents and importing their contents.

IMPORTING XML INTO RELATIONAL DATABASES

One way to import XML into your relational database is to parse the XML in an application and then generate SQL statements to insert, delete, or update tables. There are currently several parsers and tools available that allow you to do this, such as the built-in OpenXML procedures/functions.

The ADO.NET DataSet object allows you to read the data and/or schema from an XML document into a DataSet. The ReadXml method populates a DataSet from an XML document that contains both a schema and data, while the ReadXmlSchema method reads only the schema from an XML document. Once the DataSet is filled with the data from the XML document, you can update the tables in your database with the changes from the DataSet.

Both of these methods can be used with current releases of SQL Anywhere to import data from an XML document into your database.

EXPORTING XML FROM RELATIONAL DATABASES

XML can be exported directly in a SELECT statement, by using the FOR XML option.

SQL Anywhere version 11 also allows you to export XML from Interactive SQL. The OUTPUT statement supports an XML format that outputs query results to a generated XML file. The generated XML file is encoded in UTF-8 and contains an embedded DTD. In the XML file, binary values are encoded in character data (CDATA) blocks with the binary data rendered as 2-hex-digit strings.

There are two other ways you can export your data as XML in current releases of SQL Anywhere: use the ADO.NET DataSet object, or assemble the result set into an XML document within your application.

The ADO.NET DataSet object allows you to save the contents of the DataSet in an XML document. Once you have filled the DataSet (for example with the results of a query on your database) you can save either the schema or both the schema and data from the DataSet in an XML file. The WriteXml method saves both the schema and data in an XML file, while the WriteXMLSchema method saves only the schema in an XML file.

Alternatively, you can execute a SQL query and then assemble the results in an XML document in an application.

STORING XML IN RELATIONAL DATABASES

In current versions of SQL Anywhere, you can use the XML data type to store an XML document in your SQL Anywhere database. The XML document is stored as a string.

QUERYING XML

Once you have XML stored in the database, you may want to query it. This can be done using the built-in functions and stored procedures. More information can be found in the product documentation.

XML WEB SERVICES

A Web service is programmable application logic that can be accessed by other applications in different locations via the Internet using a set of underlying standard Web protocols such as HTTP, Simple Object Access protocol (SOAP), and XML. Web services are designed to be platform and language independent.

Web Services Description Language (WSDL) is an XML-based language that is used to describe a Web service and how to access it. Universal Description Discovery and Integration (UDDI) is a registry for Web services; it's similar to a search engine, but for Web services. Neither WSDL nor UDDI are required for a Web service, but they make it easier for programs to find (using UDDI) and use (using the WSDL information) a Web service.

Web services in SQL Anywhere are supported via an integrated HTTP server and a SOAP request manager in the database server. This allows you to send SOAP requests to SQL Anywhere via HTTP, and then SQL Anywhere returns a response to the requesting HTTP client.

Furthermore, there are several data formatting options available, which include the .NET data payload format. This formats the data in such a way that data-typing is preserved through transmission, and can be used properly from within your application.

ADO.NET AND .NET DATA PROVIDERS

ADO.NET is the latest data access API from Microsoft in the line of ODBC, DAO, RDO, OLE DB, and ADO. It is the preferred data access component for the Microsoft .NET Framework. .NET data providers are tools that provide access to data stores so that applications can retrieve and modify data from the data source. In ADO.NET, data providers (also called managed providers) are used to facilitate the integration of data with .NET applications.

The .NET Framework, as shipped by Microsoft, currently includes three data providers: the SQL Server .NET Data Provider, the OLE DB .NET Data Provider, and the ODBC .NET Data Provider. The SQL Server .NET Data Provider allows you to connect to Microsoft SQL Server version 7.0 or later databases, while the OLE DB .NET Data Provider allows you to access relational database systems (including SQL Anywhere), as well as data sources for which there is an OLE DB provider. The .NET Framework also supports Microsoft's ODBC .NET Data Provider for connecting to any ODBC data source.

The OLE DB driver included with SQL Anywhere .NET Data providers have been implemented to supplement the three already included in the framework. This allows you to develop applications in any of the supported .NET languages using the standard ADO.NET interface for all Windows platforms including Windows Mobile.

Alternatively, the standard non-managed SQL Anywhere Data Provider can be accessed via the OLE DB, and ODBC data providers.

THE ADO.NET PROGRAMMING INTERFACE

Each managed provider implements the following classes, which are the standard objects for database manipulation and management:

- **Connection:** connects to a data source
- **Command:** executes a command (SQL statements or stored procedure) against a data source
- **DataReader:** provides forward only, read-only access to results of a command
- **DataAdapter:** fills a DataSet and handles updates to data
- **Parameters:** passes parameters to a Command object
- **Transaction:** provides COMMIT and ROLLBACK functionality
- **Error, Exception:** handles and collects error and/or warning messages

One of the primary objects in ADO.NET is the DataSet. The DataSet is a disconnected store for data retrieved from a database. It is a collection of DataTable, DataRow, DataColumn, and DataRelation objects. The DataSet is a generic object provided by Microsoft as part of the ADO.NET architecture. Using a provider's DataAdapter, you can fill a DataSet, modify the data in the DataSet, and then apply the changes to the database through the DataAdapter. The DataSet is independent of any managed provider or database driver and can be used to read and write either data or schema information in XML.

Caution should be used when updating your database from a DataSet. Any changes you make to a DataSet are made while you are disconnected. This means that your application does not have locks on these rows in the database. Your application must be designed to resolve any conflicts that may occur when changes from a DataSet are applied to the database.

MobiLink provides a solution for resolving conflicts in a replication environment. MobiLink is a session-based relational database synchronization system that is intended for two-way synchronization of data between a central, consolidated database and a large number of remote databases. It allows you to choose selected portions of data for synchronization and includes features that allow you to resolve conflicts between changes made to data in different remote databases.

The SQL Anywhere .NET Data Provider is a native .NET data provider for SQL Anywhere databases. Unlike the other supported providers, it communicates directly with SQL Anywhere and does not require bridge technology. It also runs on Windows Mobile.

The .NET data provider implements the iAnywhere.Data.SQLAnywhere namespace and allows you to write programs in any of the .NET supported languages, including C# and Visual Basic .NET, and access data from any SQL Anywhere database.

The following are some key features of the SQL Anywhere .NET Data Provider:

- **Application flexibility:** The SQL Anywhere .NET Data Provider supports all classes, including DataReader and DataAdapter.
- **Support for .NET languages:** The SQL Anywhere .NET Data Provider can be used with all .NET supported languages, including C# and Visual Basic .NET.
- **Support for connection pooling:** The SQL Anywhere .NET Data Provider supports connection pooling, which allows your application to reuse existing connections rather than repeatedly creating a new connection to the database. This may improve performance.
- **Superior performance:** The SQL Anywhere .NET Data Provider is a direct implementation, and does not require OLE DB or ODBC bridges. It is faster than the .NET OLE DB Data Provider.
- **Broad Microsoft Windows support:** The SQL Anywhere .NET Data Provider supports both Windows and Windows Mobile platforms. Currently, Microsoft does not supply an OLE DB or ODBC managed provider that allows you to access data on Windows Mobile. The SQL Anywhere .NET Data Provider is the only way you can access SQL Anywhere using the .NET Compact Framework on Windows Mobile.

CLR STORED PROCEDURES AND FUNCTIONS

New in version 11 of SQL Anywhere, is the .NET external environment. It can be used to execute any CLR (2.0 or lower) code. This allows you to reference a function or procedure written in C#, VB.NET, or any other .NET language that resides in a class library (.dll). The database will load the library, and call the function in place of a SQL stored procedure. This allows you to manipulate data in any .NET language, thus preserving and re-using your business logic.

MOBILINK .NET SUPPORT

MobiLink is a session based synchronization system that allows two-way synchronization between a main database, called the consolidated database, and many remote databases.

MOBILINK SYNCHRONIZATION SCRIPTS

Synchronization scripts are used to control the actions of the MobiLink synchronization server. Generally, the synchronization scripts are written as stored procedures in the SQL language of the consolidated database, or in Java.

SQL Anywhere also allows you to use any of the supported .NET programming languages to write MobiLink synchronization scripts. This gives you full access to all the functionality of the .NET Common Language Runtime, and you can write scripts in C#, Visual Basic .NET, or any other supported .NET language.

Using .NET synchronization logic allows you to perform operations across database platforms, and provides portability across RDBMSs. With .NET synchronization logic, you can use MobiLink to access data from application servers, Web servers, and files. You can use iAnywhere classes in your synchronization logic to access data on the consolidated database's synchronization connection. For example, you can write a .NET script to use an external server to validate a user ID and password in the server's `authenticate_user` event.

Scripts also allow you to access and manipulate uploaded data in the consolidated database before it is committed. For example, you could reject a change before it is committed so that other remotes would not receive it. If you use an external program to access the data on the consolidated database, you cannot view or undo the update until it has been committed.

This is also supported for use within UltraLite databases. UltraLite is a smaller footprint alternative to SQL Anywhere used to build and deploy relational database applications on small devices.

DIRECT ROW API

Alternatively, the Direct Row API passes data directly through the .NET environment, allowing you to manipulate it before committing it to the database.

MOBILINK .NET API FOR DBMSYNC

DBMSync is the synchronization client that initiates client-side synchronization. As of version 11, an API is made available for use in your .NET applications. The `iAnywhere.MobiLink.Client` namespace is

implemented, and can be used from any .NET language. Refer to the documentation for examples and references.

SUMMARY

SQL Anywhere currently provides a wide range of .NET support, and future releases will provide even greater support.

The SQL Anywhere .NET Data Provider provides native access to your .NET applications, including applications running on Windows Mobile. In addition, you can currently import, export, and store XML in your SQL Anywhere database.

MobiLink gives you full access to the functionality of the .NET Common Runtime Languages by allowing you to write your synchronization scripts in any of the supported .NET languages.

Copyright © 2008 iAnywhere Solutions, Inc. All Rights Reserved. Sybase, Afaria, SQL Anywhere, SQL Anywhere, MobiLink, UltraLite, and M-Business Anywhere are trademarks of Sybase, Inc. All other trademarks are property of their respective owners.