WHITE PAPER

An Advantage for Delphi Developers

By: Cary Jensen, Ph.D.

SYBASE IANYWHERE

SYBASE iAnywhere

TABLE OF CONTENTS

This article first appeared on Cary Jensen's blog at http://caryjensen.blogspot.com/2009/05/advantage-for-delphi-developers.html and is reprinted with permission of the author.

Here's a trivia question: What was the name of Delphi during its original beta test?

The answer is Delphi (but you probably knew that already). But why did the Delphi development team pick such an odd name for their ground breaking, component-based heir to their Pascal development tool? The answer is related to databases.

Delphi is name of both the city and temple in Greece where people would travel to speak to the Oracle. And although Delphi can work with just about any database you can think of, the point was that it was designed from the beginning to be a great environment for developing database applications.

It's ironic, then, that while Delphi's name alludes to ORACLE, one of the most popular SQL-based relational database management systems (RDBMS), the foundations of database access in Delphi is navigational, not set oriented. Specifically, the original data access mechanism in Delphi was the Open Database Application Programming Interface, or ODAPI.

ODAPI, which would later be referred to simply as the Borland Database Engine, or BDE, was a direct outgrowth of the Paradox Engine. The Paradox file format, like many of its file-based cousins, is index based. In other words, database tables are treated as an ordered sequence of records (rows), whose order can be changed by changing which index is used to access the records. These indexes can also be used to filter records of a single table, as well as perform joins between tables having compatible indexes.

DATA ACCESS IN DELPHI

That Delphi's database roots are well grounded in the navigational model is immediately obvious if you consider the TDataSet class, which is the base class for all data access components in Delphi. This class includes properties and methods for a variety of navigational operations. For example, for selecting indexes (IndexName), setting ranges (SetRange), finding records based on indexed values (FindKey, FindNearest), creating relational joins using indexes (MasterSource and MasterFields), and navigating records in index order (First, Next, Prior, MoveBy, and so forth).

While the benefits of SQL-based data access are well-known, navigational data access has its advantages as well. Specifically, the navigational model better suits the development of intuitive and user-friendly interfaces.

For example, with a navigational interface, it is possible to display all of the records from a database table (or a query result set), and permit the user to navigate these records freely. This navigation may even include incremental search (always an end user favorite). And, these features are available whether the underlying table has a couple of hundred records, or millions.

By comparison, nearly all set-based databases, such as MS SQL Server and ORACLE, are not designed to provide navigational access. As a result, searching for a record in a result set means refining an underlying WHERE clause in a SQL query to reduce the resulting result set to a manageable size. The idea of opening a result set with millions of records and permitting a user to freely browse it is unthinkable.

Here is where the Advantage Database Server really shines. Not only is it a high-performance, low maintenance database server that supports optimized SQL queries, its index-based architecture permits it to provide data access options normally only found in a file-based database (such as Paradox, dBase, or MS Access), while providing all the benefits of a transaction-based remote database server.

But there is more - a lot more. To begin with, database developers creating native Delphi applications have access to Advantage Delphi Components, a component set that implements the native TDataSet interface for access to ADS. (For Delphi developers using Delphi 7 and earlier, the Advantage TDataSet Descendant provides the same features.)

These components provide a near seamless replacement for BDE components, but go even further. For example, while the BDE supports filters, they are not optimized. With Advantage Delphi Components, so long as your tables have the appropriate indexes, filters are fully optimized. It is possible to set a filter on a 20 million record table and have the result in a fraction of a second.

For Delphi for .NET, as well as for Delphi Prism developers, Advantage offers the Advantage Data Provider for .NET. This ADO.NET provider not only supports all features defined by ADO.NET, but it even supports optimized, server-side cursors. Specifically, while all .NET data providers support a forward navigating, read-only DbDataReader descendant, the AdsExtendedReader class, which is also a DbDataReader descendant, supports bi-directional navigation, optimized server-side filters, read/write support, as well as pessimistic locking. You cannot find better .NET data provider anywhere.

GENERAL OVERVIEW OF THE ADVANTAGE DATABASE SERVER

The Advantage Database Server is a high-performance, relational database server that simultaneously supports optimized set-based SQL, as well as a blinding-fast, index-based navigational operations. Its low per-seat cost, ease of deployment, and very low maintenance requirements makes it particularly well suited for vertical market applications, especially those where the deployed sites lack the IT infrastructure required to maintain normal database servers.

There is another reason why Advantage is so popular with vertical market developers. In addition to the Advantage Database Server, Sybase iAnywhere publishes the Advantage Local Server (ALS), a free, file-server based, five-user database that sports an interface identical to ADS, providing a seamless upgrade path to ADS. For those customers for whom cost is an issue, you can deploy your applications using ALS. Later, if the needs of the customers grow, or they want the stability of a transaction supporting, remote database server, migrating can be as simple as installing ADS (and this installation takes less than five minutes in most cases).

Advantage is a Full-Featured Database

While the benefits of ADS mentioned so far in this article provide a compelling case for building database applications with ADS and Delphi, there is much more to the story. The remainder of this article provides you with a brief overview of the many features that makes ADS a strong contender for just about any database development environment. All of these features, with the exception of replication, online backup, and support for class 4 Java drivers, are also supported by ALS.

At the time of this writing, the current version of ADS is version 9.1. While some of the features defined below have been available since version 6.0, many have been either introduced or enhanced in later versions.

Advantage Comes in Many Flavors

There is not just one Advantage Database Server, there are several. For Linux and Windows servers, there are both 32-bit and a 64-bit versions of ADS. If you are running Novel, there is an Advantage Database Server NLM (Netware Loadable Module). The 64-bit versions of ADS were introduced in ADS 9.0.

An Easy to Use Data Architect

While you can control all aspects of your database design and management through Advantage SQL (or even the Advantage Client Engine), most of the time, you will want to design and configure your Advantage databases and data dictionaries using the Advantage Data Architect. The Advantage Data Architect is shown in Figure 1.

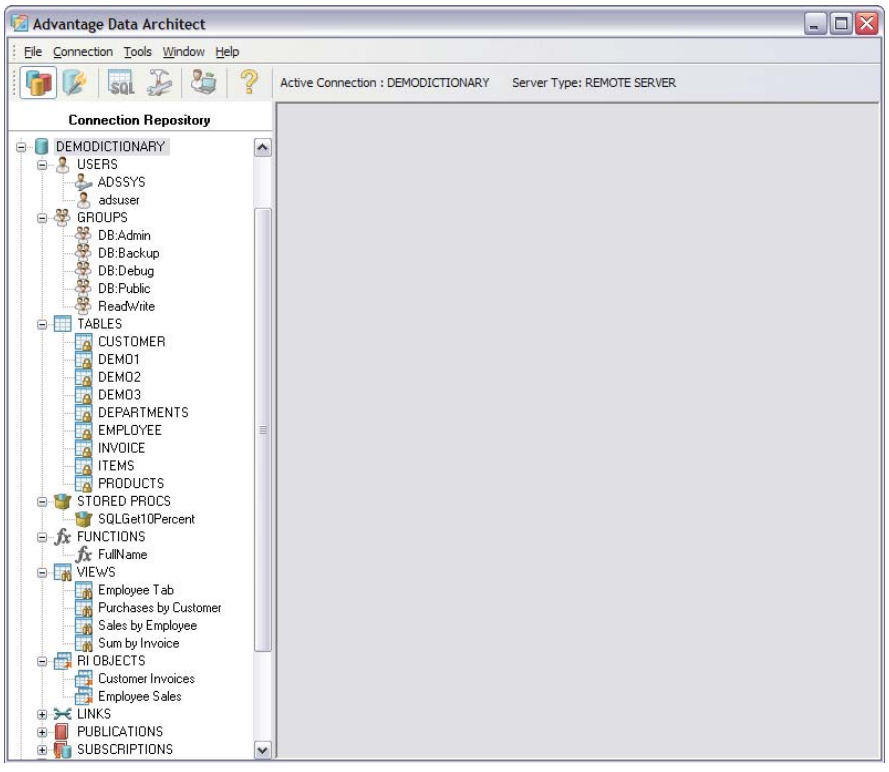

Figure 1: The Advantage Data Architect

The Advantage Data Architect provides you with a rich graphical interface for working with Advantage objects. For example, you can use the Advantage Data Architect to create data dictionaries and database tables, create and configure users and groups, define stored procedures, create views, write user defined functions, and execute queries against your database objects.

The Advantage Data Architect also comes with an invaluable SQL debugger. This debugger helps you step through your SQL queries, SQL stored procedures, SQL triggers, and user defined functions. It supports inspection of local variables, multiple breakpoints, live editing of your SQL queries, and more.

A Multitude of Data Access Options

Advantage supports one of the widest arrays of data access mechanisms in the industry. In addition to the Advantage Delphi Components, TDataSet Descendant, Advantage Data Provider for .NET, and Java class 4 drivers already mentioned, Sybase iAnywhere publishes a range of additional drivers. These include an ODBC driver, an OLE DB Provider, a PHP driver, a DBD (Perl) driver, a Clipper Replaceable Database Driver (RDD), and a Crystal Reports driver.

In addition, some third party vendors provide their own Advantage drivers. For example, Alaska Software publishes the ADS-Database Engine for their Xbase++ language, an object-oriented language based on Clipper.

Data Dictionaries with Database Wide Security

Advantage first introduced data dictionary support in 2000 with the release of Ads 6.0. Data dictionaries provide a number of valuable features, some of which are associated with security. For example, with data dictionaries you can define users and groups, which permit you to define which database objects (including tables, fields, stored procedures, views, and so forth), individual users, or groups of users, have access to. This access can include full access, read/write access, readonly access, or no access.

Furthermore, data dictionaries enable table encryption to be applied across the entire database, without having to encrypt and decrypt tables on a table-by-table basis. In addition, encryption can be extended to network and Internet communications. Specifically, Advantage data can be transparently encrypted using 160-bit encryption before being moved across your networks, or even the Internet. All you need to do is configure your data dictionary for the encryption option you want. There is no need to manually perform encryption and decryption in your client applications.

Stored Procedures, User Defined Functions, and Views

Stored procedures and user defined functions are routines that you create for your data dictionaries that can be used by any applications that access your Advantage data. For stored procedures, these routines can be written using DLLs (a Delphi project template is supplied), COM objects, or .NET managed assemblies. Stored procedures can also be written using SQL Scripts (SQL persistent stored modules, or PSMs). SQL Scripts are the only option for writing used defined functions (UDFs).

Views are also based on SQL. However, views are SQL SELECT statements that you define in your data dictionary, and then call, as though they were a database table, from your client applications.

Support for Constraints

Advantage supports a range of constraint options. The most prominent of these is referential integrity (RI) constraints. Though they should be used judicially, RI constraints provide you with a mechanism to ensure the relational integrity between two or more related tables.

Advantage also provides both table-level and field-level constraints. These constraints define rules that ensure that data entered into your tables meets criteria that you specify, without your having to embody these rules in each of your client applications.

Online Full and Incremental Backup

In version 8.0 Advantage introduced online back. Online backup permits you to create a copy of your data, which you can use in the case of a catastrophic failure of your server.

Advantage supports two modes of online backup. A full backup creates a complete backup of your data each time it is run. Incremental back, by comparison, maintains a single backup, but updates it with the incremental changes that were detected since the previous back was completed.

Database Replication

Advantage also introduced database replication in ADS 8.0. With replication, changes to your database are propagated to one or more servers. Replication in Advantage uses a publish/subscribe model. As a result, replication can be performed between two or more Advantage servers using either unidirectional or bi-directional modes.

Replication in ADS 8.0 was performed on a whole record basis. In ADS 9.0, it became possible to filter replication in order to replicate only certain fields of a table.

Triggers and Notifications

Triggers, which were added in ADS 7.0, and notifications, which were added in ADS 9.0, provide you with the option to react to changes that occur in your databases programmatically. Triggers, which, like stored procedures, can be written as DLLs, COM objects, .NET managed code, or SQL scripts, are routines that execute in response to a change occurring to a record in an underlying table, such as an insertion, modification, or deletion. In addition, triggers can be defined to trigger before the operation (permitting you to prevent it), during the operation (permitting you to implement it), or after the operation (allowing you to respond to it).

Notifications provide a mechanism for communicating to a client that something has occurred in the database. For example, notifications can be used to inform a client application that data in a critical table has changed. The client application can then use this information to refresh its view of that table, providing the end user with the most current data. Or, a notification might be used to signal a client that a record has been added to a special table created for the purpose of communicating messages from the system administrator to the end users. The client can then read the latest message and display it within its interface.

CONCLUSION

The Advantage Database Server stands alone in the world of relational database servers (RDBMs). It not only supports the navigational model, but also provides optimized support for SQL operations. This navigational support makes it a perfect match with Delphi's data access model. This, combined with its advanced features, very low maintenance, high performance, and ease of deployment, make it an ideal database server for a wide range of applications.

ABOUT THE AUTHOR

Cary Jensen is President of Jensen Data Systems, Inc., a Texas-based company that provides software training, development, consulting, and mentoring. He is an award-winning, best-selling co-author of twenty books, including Advantage Database Server: A Developer's Guide (Sybase), Building Kylix Applications (McGraw-Hill), Delphi In Depth (Osborne/McGraw-Hill), and Programming Paradox 5 for Windows (Sybex). He is a popular speaker at conferences, workshops, and training seminars throughout North America and Europe. Cary has a Ph.D. in Human Factors Psychology from Rice University, specializing in human-computer interaction.

You can contact Cary at cjensen@JensenDataSystems.com, and visit his company Web site at www.JensenDataSystems.com.

Contact Us

iAnywhere Solutions, Inc.
Worldwide Headquarters
United States
One Sybase Drive
Dublin, CA 9
4568-7902 U.S.A.

For General Information:
AdvantageInfo@sybase.com
North America
T 1-800-235-7576
1-208-322-7800

For Specific Regional Product Inquiries
North America
1-800-235-7576
(208)-322-7800
Germany
+49 (0)7032/ 798-200
United Kingdom
+44 (0)117 315 3900

www.AdvantageDatabase.com