SYBASE®
An SAP Company

WHITE PAPER

# Sybase® Advantage Database Server® 11: A First Look

by Cary Jensen

**TABLE OF CONTENTS**



## OVERVIEW

Advantage 11 introduces a wealth of new and enhanced features that make it an essential upgrade for existing users, and increases the value of Advantage for those looking for a better database server. While significantly increasing Advantage's already abundant connectivity options, Advantage 11 adds additional value to many of its already advanced features.

The Advantage Database Server is a high-performance, low maintenance database server that has earned the affection of small-to-medium sized businesses and vertical market developers. It combines speed and high-end features with an ease-of-use that is welcome but unexpected. And now, with the release of Advantage Database Server 11, a whole new world of features and enhancements make Advantage even more powerful and easier to use and manage.

This paper is designed to introduce you to the many exciting new features introduced in Advantage 11. It begins by looking at the Advantage Web Platform, a RESTful service that makes your data available from anywhere, anytime. Next, we will look at the new online maintenance features, improvements to replication, and the many enhancements to Advantage's support for SQL, the structured query language. Later in this paper, you will learn about the new and enhanced connectivity options, general performance updates, and improved error reporting.

## ADVANTAGE WEB PLATFORM

One of the most anticipated updates in Advantage 11 is the new Advantage Web Platform. The Advantage Web Platform is a RESTful Web service that provides you with the ability to interact with your Advantage databases over the World Wide Web.

There are two distinct capabilities that the Advantage Web Platform enables. The first is the ability to read, update, and delete data from your Advantage data dictionaries using a RESTful interface. This feature is implemented through the OData protocol, an industry standard for accessing data using a REST interface. This interface also provides you with access to metadata, stored procedure and query execution, as well as offline storage. The second capability is that it permits you to manage and query your databases from a Web browser, permitting you to access your data from anywhere, using any browser, even from a smart phone.

When you install the Advantage Web Platform, you are actually installing a lightweight version of Apache Web server, and it is through this Web server that the Advantage Web Platform provides access to your data. This service is configured by default to listen to HTTP (hypertext transfer protocol) requests on port 6272, and HTTPS (secure HTTP) on port 6282. These port numbers, and other aspects of the Apache Web service can be configured. Please see the Advantage help details.

The following sections describe the two principle features provided by the Advantage Web Platform.

### Data Anywhere Using the Advantage Web Platform and OData

OData is a standards-based protocol for accessing data using a RESTful Web interface. REST, which stands for REpresentation State Transfer, is a description of the basic request and response interaction that drives the World Wide Web. A RESTful Web service is one that you can interact with using basic HTTP commands over the Internet.

A significant feature of REST is that you call the methods of your Web service using the values that you pass in your URI (uniform resource identifier), and sometimes pass additional data in the HTTP message body. These methods, when they return a value, return a string. Using the OData protocol, this string will either be in AtomPub (Atom Publishing Protocol), an XML-based format, or JSON (JavaScript Object Notation), and you get to choose which format you want.

Because OData uses REST, any client that can communicate across the Internet, and which can issue HTTP requests, can interact with the Advantage Web Platform to retrieve, update, and manage data in your database. Importantly, because OData relies on HTTP, the client from which the RESTful Web service is accessed does not need the Advantage client API installed locally.

OData can only be used to interact with Advantage data dictionaries that have been configured for access through the Advantage Web Platform, and only through the Advantage Database Server (the Advantage Web Platform cannot be used with the Advantage Local Server). For information on configuring a database for access through the Advantage Web Platform, please see the Advantage help.

Fortunately, nearly every modern client development tool has the capability to communicate using HTTP, including .NET, JavaScript and jQuery, Java, C++, Objective-C, and Delphi, to name just a few. This means that the Advantage Web Platform provides access to your Advantage data from almost any imaginable platform, including Windows, Linux, the Mac, Unix, mobile devices, iOS and Android tablets, and Web pages.

OData consists of a sophisticated collection of standards and capabilities. For more information, including examples of how to use OData to communicate to the Advantage Web Platform, see the Advantage help. For more information on OData, visit http://www.odata.org/.

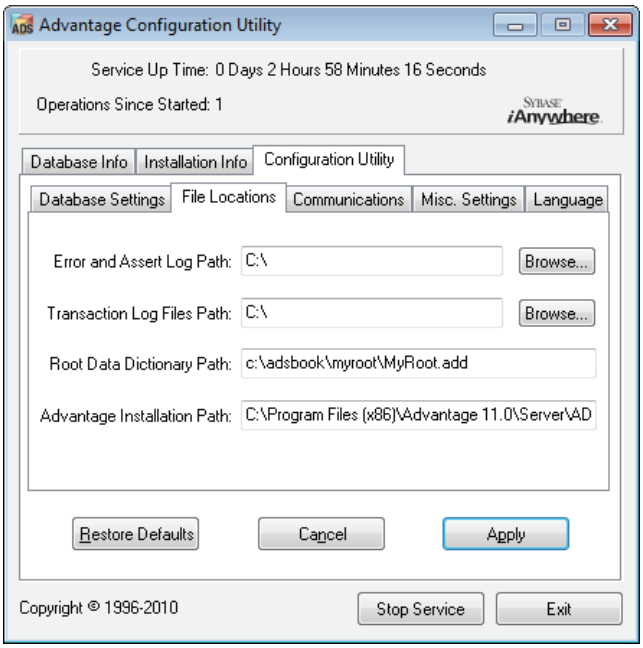
### The Advantage Web Administrator

The Advantage Web Administrator is a Web application that gets installed along side of the Advantage Web Platform. This Web application provides you with a wide range of options, including the ability to view information about connected users, active queries, configuration parameters, communications statistics, and more.

Based on how you configure the Advantage Web Administrator, you may also be able to run ad-hoc queries, interact with multiple data dictionaries, and even change configuration parameters (though many of these configuration changes made within the Advantage Web Administrator require a subsequent re-start of the Advantage Database Server service).

To provide an entry point for communication to Advantage through the Advantage Web Administrator, a new concept, the *root dictionary,* has been introduced. Each server can have at most one root dictionary, and it is the dictionary to which the Advantage Web Administrator initially connects.

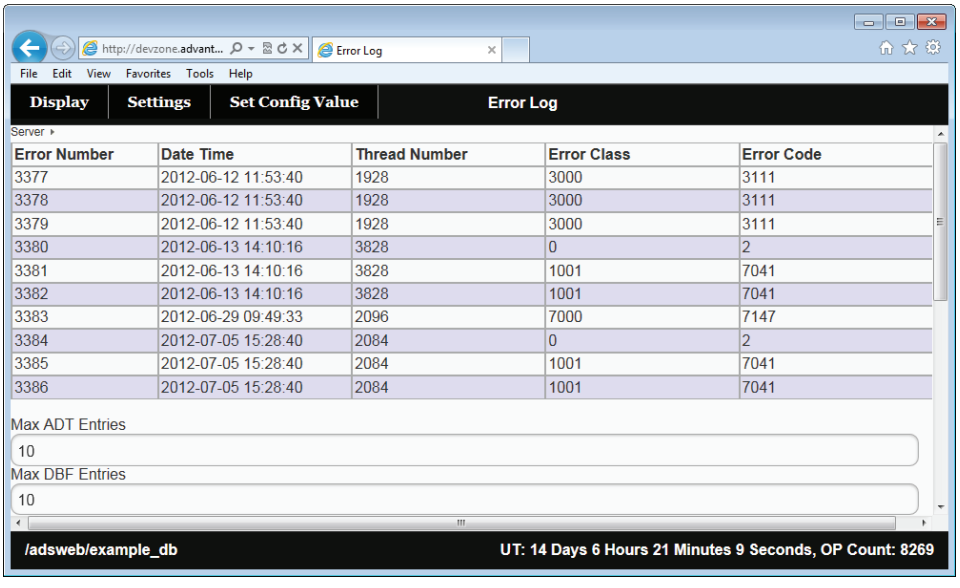
You configure the root dictionary for a particular server using the Advantage Configuration Utility. The Advantage Configuration Utility, showing the new Root Dictionary file location, is shown in the following figure.




You can select one of your existing data dictionaries to serve as your root dictionary. Alternatively, you can create a new, empty data dictionary, and use that as your root dictionary. This second option has the benefit of providing you with greater control over the data dictionary features that you want to expose using the Advantage Web Administrator.

In addition to defining a root dictionary using the Advantage Configuration Utility, there are several additional configurations that you must perform. You must specify a location for the root dictionary in the adsweb.conf file, located in the C:\Program Files\Advantage 11.0\adsweb\apache\conf directory, as well as set the JavaScript rootDB variable declaration in the configOptions.js file, located in C:\Program Files\Advantage 11.0\adsweb\apache\htdocs\adsconfig. After changing the adsweb.conf file, you must restart the Advantage Web Platform service. For more information on configuring the Advantage Web Administrator, please see the Advantage help.

The Advantage Web Administrator is shown in the following figure, where the contents of the Error logs are displayed



## ONLINE MAINTENANCE

Online maintenance is one of the more popular updates introduced in Advantage 11. Online maintenance lets you pack, reindex, and alter the tables of your database without having to first obtain an exclusive lock on those tables. This feature permits you to make significant changes to your tables without having to limit your users' access to their data.

There are three aspects to online maintenance. The first is how Advantage treats these maintenance operations while users are actively using your database. The second concerns a collection of new system stored procedures introduced in Advantage 11 that permit these operations to be initiated programmatically. Finally, each of the online maintenance functions is also associated with a notification that will trigger when the operation is complete.

### How Online Maintenance Is Applied

Normally, pack, reindex, and alter operations require that you have exclusive access to the table on which you are performing the operation. This often means that these operations have to be performed only after taking the database offline, which, in a production environment, can be difficult, or at least inconvenient.

When performing online maintenance, Advantage begins by creating a copy of the original table, and duplicating both the data and its indexes to the copy. It then uses this copy to apply the requested change. For pack and reindex online maintenance, once any pending transactions have been applied, and all users have released their locks, the new table becomes available. Users will begin using that updated table as soon as they make a request that results in a refresh of their cursor, such as navigating to a new record or setting a filter.

Because an online alter physically changes the structure of the table (or its indexes), users must actually close and reopen that table before the new structure becomes available.

Online maintenance is only supported using the Advantage Database Server. You cannot perform online maintenance from the Advantage Local Server.

### Online Maintenance System Stored Procedures

Advantage 11 introduces three new system stored procedures associated with online maintenance. These are sp_PackTableOnline, sp_PackAllTablesOnline, and sp_ReindexOnline, and they permit you to perform online maintenance programmatically from your custom applications.

Calls to these stored procedures return once the operation has been completed on the copy of the table. However, individual users may continue to use the old version of the table until they reposition their cursor. As a result, a return from sp_PackTableOnline, sp_PackAllTablesOnline, or sp_ReindexOnline does not signal that all users are current, only that all users are in a position to use the new version of the table.

There is no special system stored procedure associated with online alter operations. Instead, you execute a special version of an ALTER TABLE query, ALTER ONLINE TABLE. Other than the ONLINE keyword, the syntax of an ALTER ONLINE TABLE query is the same as for a normal ALTER TABLE statement.

### Online Maintenance Notifications

Notifications, which were introduced in Advantage 9, provide a mechanism by which a client application can request to be informed by Advantage when a particular operation has taken place. For example, a client may ask to be notified when new records are inserted into a particular table, or when updates are made to a table's contents.

There are two parts to a notification. First, a server must issue a specific notification through the execution of the sp_SignalEvent stored procedure. Second, a client must subscribe to a notification and then wait for it. A client subscribes to a notification by calling sp_CreateEvent, and waits for the notification using the sp_WaitForEvent stored procedure. Since the sp_WaitForEvent stored procedure call blocks until the notification is issued, this stored procedure is often called from a worker thread on the client.

In the case of online maintenance operations, Advantage always issues a notification when the online operation has reached a stage where users can begin transitioning to the updated table. In the case of online packing or reindexing, a single notification is issued once users can switch to the new table.

In the case of an online alter table operation, Advantage will issue a notification once users can be transitioned to the new table. In addition, Advantage will continue to issue notifications every minute until all users have closed their connection to the old version of the table.

If you want your client applications to be informed once an online maintenance operation has reached the point where a transition to the new table is possible, your client applications must subscribe to, and listen for, these notifications.

A client subscribes to an online maintenance notification by calling the sp_CreateEvent system stored procedure, passing in the first parameter, an event name using a specific pattern. The event name prefix begins with two underscore characters, following by OP, OR, or OA, which corresponds to online pack, online reindex, and online alter, respectively. The middle section of the event name is FinalStage, and the suffix is a single underscore character followed by the name of the table upon which the maintenance was performed.

As a result, if you want to subscribe to a notification about the packing of a table named archive, you will use the following string:

```
__OPFinalStage_archive
```

By comparison, if you want to subscribe to a notification about the alter table operation on a table named customers, you will use the following string:

```
__OAFinalStage_customers
```

To learn more about notifications in general, and online maintenance notifications in particular, please see the Advantage help.

## REPLICATION

Replication, which was first introduced in Advantage 8, provides a mechanism for duplicating changes made to one database to one or more databases. For example, replication can be used to copy changes made at regional data centers to a centralized data center, where the combined data can be analyzed and reported. Similarly, replication can be implemented in a bi-directional fashion, permitting two databases to share changes made to the individual databases.

There are a number of requirements that must be met before you can implement replication. First of all, replication requires that the source database and the target database both use data dictionaries. Second, both the source data dictionary and the target data dictionary must be accessed by the Advantage Database Server (ADS). Replication cannot be implemented using Advantage Local Server. Finally, an additional replication license must be installed on the source server, and sometimes the target server as well, depending on how you configure your replication.

Replication is an extremely valuable mechanism for those developers who need it. Fortunately, there are several significant capabilities added to Advantage's replication introduced in Advantage 11. These include replication to older Advantage servers, additional system stored procedures to examine the replication queue, and the ability to more easily edit the replication queue. Each of these capabilities is described in the following sections.

### Replication to Older Advantage Servers

A significant update, one that I know will delight many developers, is the ability to replicate between the new Advantage 11 server and those servers running older versions of Advantage. In Advantage 11, replication to Advantage 9 and later servers is supported. In all previous implementations of replication, an older Advantage server could replicate to a new server, but a newer server could not replicate to an older one.

What makes this update so valuable is that it permits an organization to upgrade one or more of their servers to take advantage of the many new features in Advantage 11 without having to upgrade every server involved in replication. For example, a company may want to upgrade the server at their main headquarters so that they can use the Advantage Web Platform, the new SQL capabilities, and online maintenance. If that server is involved in replication with satellite offices, those offices can continue to use their Advantage 10 servers for the meantime, upgrading at a later date when it is convenient.

Replicating to an older server from Advantage 11 requires some setup. See the Advantage help for a detailed description of these steps.

### Viewing the Replication Queue

The replication queue, which is a table in your data dictionary that you designate to hold the records that will be replicated, now displays its data in a more meaningful order. Previously, the replication queue records were displayed in the natural order. Due to efficiencies in how Advantage reuses records when they are deleted, which happens to records in the replication queue once a record has been replicated, the natural order often fails to preserve the order in which the records will actually be replicated. In Advantage 11, the order of records in the replication queue now more accurately reflect the order in which the records will be replicated.

In addition, two new stored procedures assist in your work with the replication queue. You can call the system stored procedure sp_GetReplicationEntryDetails to retrieve detailed information about a record in the replication queue, including the text of the update, insert, delete, or merge statement that Advantage has generated to perform the replication on the target database.

The sp_TestReplicationConnection permits you to test whether an attempt to replicate to a replication subscription will be successful. This stored procedure is useful since it permits you to test your subscription configuration without actually having to initiate the replication.

### Editing the Replication Queue

Under normal conditions, replication works fine so long as your servers involved in the replication can maintain their connection, relying on the replication queue for temporary storage for those periods of time when a connection is not possible. However, sometimes anomalies in the data of a record or collection of records prevent the replication from being successful. When this happens, it is necessary to manually delete one or more records from the replication queue in order to allow the replication to proceed normally.

Advantage 11 introduces a new system stored procedure that helps in this process. You can now call sp_DeleteReplicationEntry to remove a problem record from the replication queue. Importantly, if this record was involved in a transaction, sp_DeleteReplicationEntry deletes all records associated with that transaction from the replication queue, thereby maintaining the integrity of the data.

## SQL ENHANCEMENTS

Some of my favorite enhancements introduced in Advantage 11 are associated with Advantage SQL, Advantage's implementation of the structured query language. Among these changes are improvements in parameter access in stored procedures and triggers, an option for variable output parameters from SQL stored procedures, new SQL functions, new system variables, a modulo operator, several new SQL literals, and the SQL command line utility. Each of these features are described in the following sections.

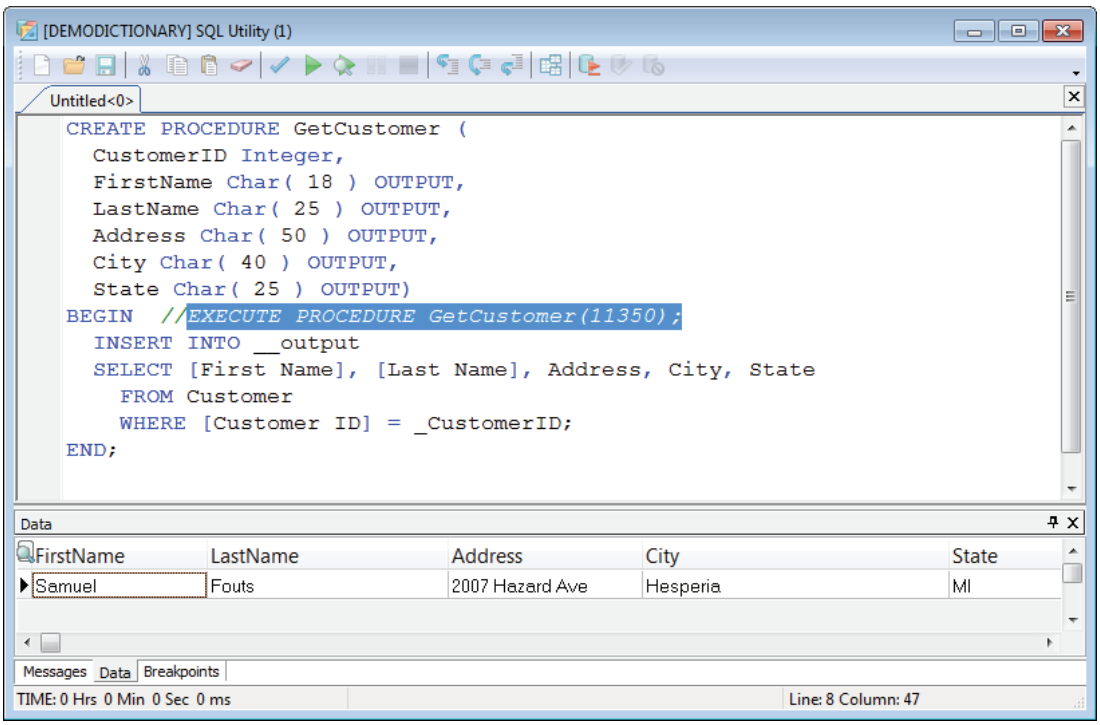
### Improved Parameter Access in SQL Stored Procedures and Triggers

In previous versions of Advantage, it was your responsibility to specifically read your input parameters from the temporary, virtual __input table. This could be done using a cursor, which you needed to open, fetch, and close, or by querying the __input table directly (this table's name is input preceded by two underscore characters). For example, consider the following simple SQL stored procedure:

```
CREATE PROCEDURE GetCustomer (
  CustomerID Integer,
  FirstName Char( 18 ) OUTPUT,
  LastName Char( 25 ) OUTPUT,
  Address Char( 50 ) OUTPUT,
  City Char( 40 ) OUTPUT,
  State Char( 25 ) OUTPUT)
BEGIN
  DECLARE @Cur CURSOR as SELECT CustomerID FROM __input;
  OPEN @Cur;
  TRY
    FETCH @Cur;
    INSERT INTO __output
    SELECT [First Name], [Last Name], Address, City, State
      FROM Customer
       WHERE [Customer ID] = @Cur.CustomerID;
  FINALLY
    Close @Cur;
  END;
END;
```

There are a total of four statements associated with the retrieval of the input parameters using this technique, which involves a cursor. And yes, in this case, we could have used a SELECT CustomerID FROM __input statement instead (one statement instead of four), but the cursor approach would have been less verbose if we had five or more input parameters.

Advantage 11 introduces a new option, which requires no extra lines of code. All you need to do is to reference the name of the input parameter, prefixed by a single underscore character. Using this technique we can write this same stored procedure using the syntax shown in the following figure.



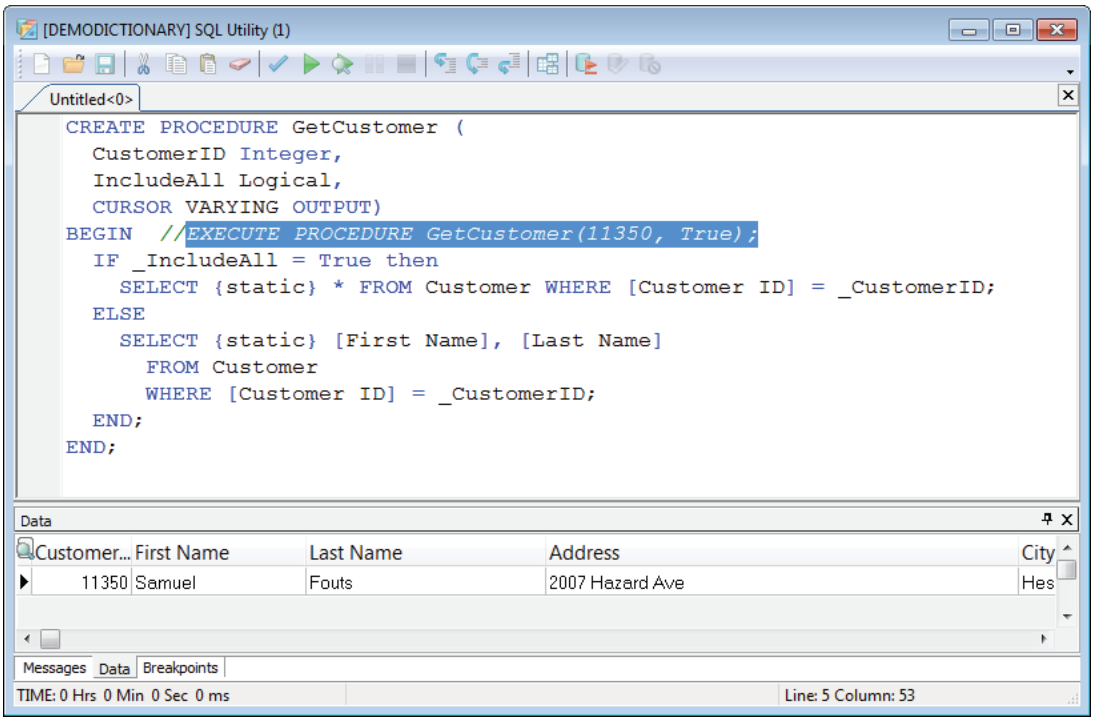

This technique can also be used in triggers. Instead of having to use a cursor or a SELECT statement to retrieve the fields of the __new table, you can use this same technique. Simply prefix the field names of the new table with a single underscore character to read the values directly.

**Variable SQL Stored Procedure Output**

Previously, you had to explicitly declare each of the output parameters of your SQL stored procedure, using the OUTPUT keyword, as seen in the preceding two queries. Furthermore, the output of your SQL stored procedures was necessarily fixed, in terms of the number of fields in the result table.

In Advantage 11, you can specify that the output is variable, permitting you to omit the output parameters altogether. When you do this, the number of fields produced by your stored procedure execution depends on the number of fields in the last SELECT statement in your stored procedure.

This can be seen in the figure on the following page, which includes a stored procedure that supports variable output parameters.

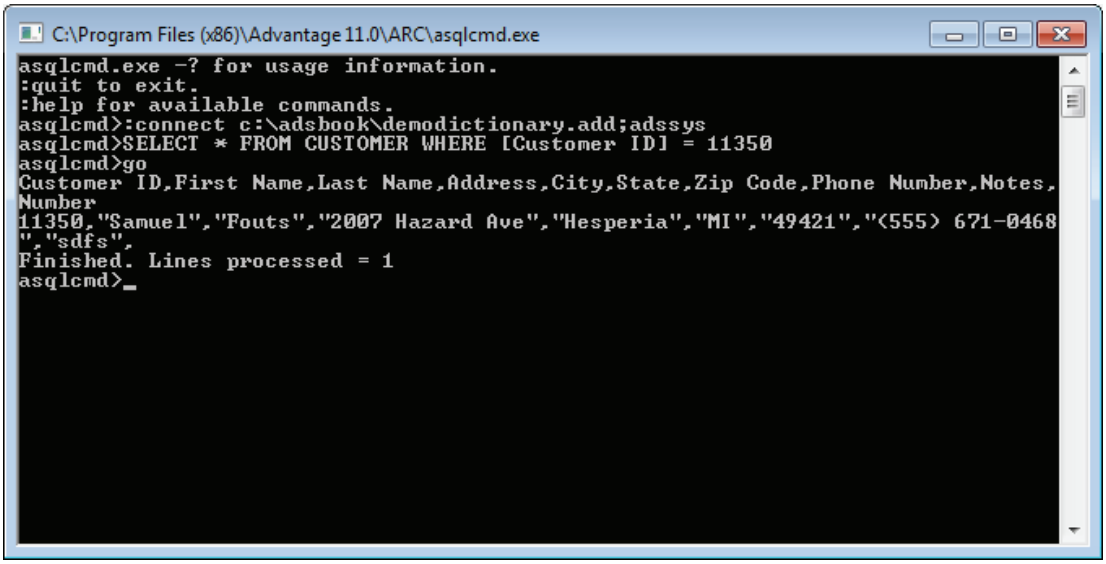



**SQL Command Line Utility**

Another highly anticipated addition to Advantage's support for SQL is the new SQL command-line utility. The SQL Command Line Utility provides two distinct capabilities. First, it can act as a command line window, similar to the Windows command prompt, from which you can connect to a database and then execute ad hoc queries. In this mode you are presented with a prompt from which you can enter queries, as well as a small set of interactive commands that permit you to connect to databases, list queries you previously executed, flush the cache of previously executed queries, and exit this interactive mode.

The second capability that it provides is the ability to create batch operations that execute single SQL statements, or execute SQL scripts that perform multiple operations. One important application for this usage is to create regularly scheduled batch operations, such as initiating backups or performing maintenance operations.

The SQL Command Line Utility is installed with many of the Advantage packages, including the Advantage Data Architect. To run the SQL Command Line in interactive mode, run asqlcmd.exe from a command prompt, or run it from the Window Explorer. The figure on the following page shows the SQL Command Line Utility running in interactive mode.



In batch mode, you will execute asqlcmd.exe from a batch file, or by scheduling a task using the Windows Task Scheduler. When run in this mode, you will pass command line parameters to the SQL Command Line Utility to provide it with connection string information, or direct it to read from one or more files where it will find connection information, queries to execute, or SQL scripts.

Here is a typical scenario of how the SQL Command Line Utility might be used. Imagine that your database is part of an interactive shopping Web site. During the day users might browse your inventory, and create virtual shopping carts, which are records in your database that identify the items users have selected. While many of your customers will continue to checkout, after which you can delete the contents of the temporary shopping carts, moving those items to an invoice table, other users will lose interest and leave your site without checking out.

Using the Windows Task Scheduler, you might run a batch file late each evening. This batch file could invoke the SQL Command Line Utility, instructing it to run a SQL script that looks for shopping carts that never made it to checkout, and which have remained unused for more than four hours or so. Those shopping carts could be considered abandoned, and the SQL script could delete those unnecessary records from their tables.

**New SQL Engine Functions**

SQL engine functions are subroutines that you can use in your SQL queries, whether they are used in stored procedures, SQL scripts, user defined functions, or triggers.

The SQL engine includes four new functions. Three of these provide information about the contents of strings. StartsWith returns True if a specified string begins with a specified substring. Likewise, EndsWith returns true if a string ends with a particular substring. Finally, SubstringOf returns true if the specified substring is contained in the specified string.

The fourth function is LastRowID. This interesting function returns the unique row id value for the most recently added row in a table. You can use this function to identify the most recently added record to a table, without knowing any of the particulars of the table's structure, such as its primary key, or without having to rely on auto increment fields, which introduce a whole slew of problems on their own.

The new SQL engine functions are listed in Table 1.

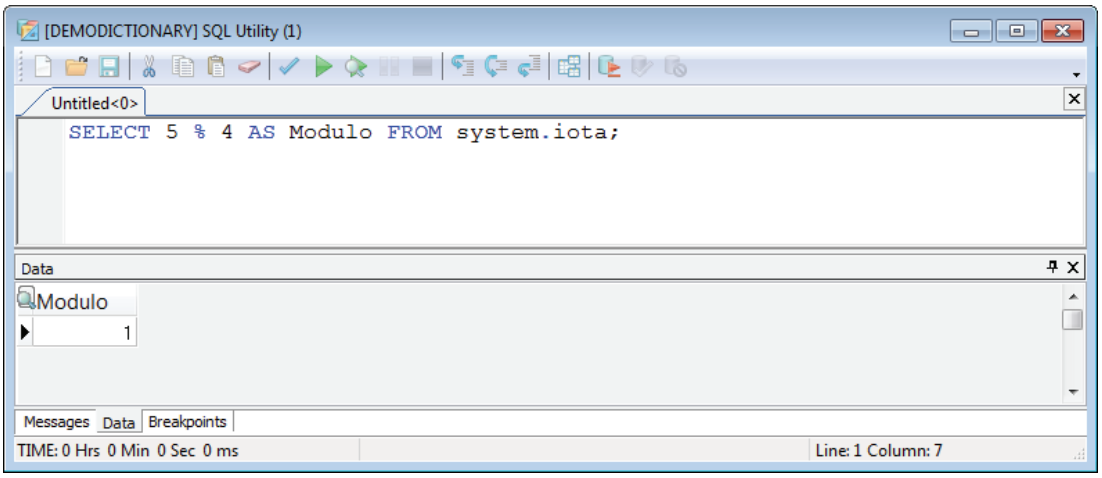
| SYSTEM VARIABLE NAME | DESCRIPTION |
|---|---|
| EndsWith() | A Boolean True if a string ends with a given substring. |
| LastRowID() | A string containing the row id of the most recently added row. |
| StartsWith() | A Boolean True if a string begins with a given substring. |
| SubStringOf() | A Boolean True if a string contains a specified substring. |

**Table 1.** The New SQL Engine Functions

**The Modulo Operator**

Advantage SQL supports a new operator, the modulo operator (%). The modulo is the remainder following a division operation on two numbers. For example, if you divide 5 by 4, the remainder, or modulo, is 1. If you divide 6 by 2, the modulo is 0.

The figure on the following page shows the use of the modulo operator in a query and the result that it produces.




The modulo operator performs essentially the same task as the MOD() SQL engine function.

**New System Variables**

System variables are symbols that are automatically available within your SQL queries and SQL scripts. Unlike the custom variables that you must declare in your SQL scripts, system variables are predefined, and can be used in expressions simply by referencing them.

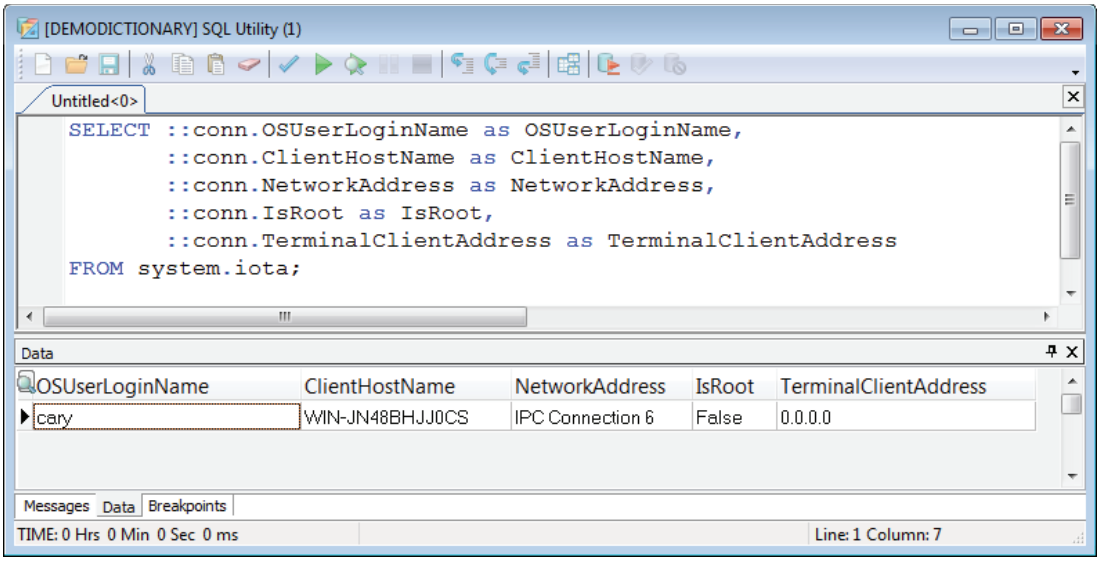
Five new system variables are introduced in Advantage 11, and all of them are connection-level system variables. Connection-level system variables are prefixed by two colons, the letters conn, followed by a period and the variable identifier. An example of one of these new connection-level system variables is ::conn.OSUserLoginName.

Table 2 contains a list of the five new system variables, along with a brief description of their associated values.

| SYSTEM VARIABLE NAME | DESCRIPTION |
|---|---|
| ::conn.ClientHostName | A string with the name of the computer on which Advantage is running. |
| ::conn.IsRoot | A Boolean value indicating if the connection is to the defined root dictionary. |
| ::conn.NetworkAddress | A string with the IP address of the machine on which Advantage is running. |
| ::conn.OSUserLoginName | A string with the currently logged in user name. |
| ::conn.TerminalClientAddress | A string with the IP address of the terminal server on which Advantage is running. |

**Table 2.** The New System Variables

The figure on the following page shows a query that includes the five new system variables, as well as the resulting output. This particular query was executed from a virtual machine.




**New SQL Literals**

Four new SQL literals have been introduced for Advantage SQL. Three of these literals can be used to represent Timestamp, Date, and Time values in queries or SQL scripts, and supplement existing literals for those types. For example, in addition to previously available literals for Timestamp values, you can now use a literal with the following format:

```
datetime'yyyy-mm-dd hh:mm:ss'
```

For example, the following query will select all records where the FromTimestamp field is less than 3:00am on January 1, 2012:

```
SELECT * FROM [Accounts] WHERE [FromTimestamp] < datetime'2012-01-01 03:00:00';
```

Date and Time literals can now also be represented by time'hh:mm:ss' and date'yyyy-mm-dd', respectively.

The fourth new literal type represents binary data. You define a binary literal by preceding a string with the letter x (upper or lower case). The string must contain a hexadecimal value consisting of 0-9 and a-f and A-F. Spaces can be used for readability, but are not required. There is no limit to the number of characters that can appear in the string.

The following two queries are equivalent, and both test the Key binary field for a very small binary value:

```
SELECT * FROM Products WHERE Key = x'29 a2 f3 0e 16 76 ee';
SELECT * FROM Products WHERE Key = X'29a2f30e1676ee';
```

**New System Stored Procedures**

System stored procedures are built-in routines that you can execute using the SQL EXECUTE PROCEDURE syntax. These stored procedures are typically used to control or interrogate your databases, permitting you to add intelligence to the various operations that you need to perform.

In discussing some of the new features introduced in Advantage 11, I have mentioned in passing new system stored procedures that you can use. Table 3 contains a list of all of the new system stored procedures that are introduced in Advantage 11.

| STORED PROCEDURE NAME | DESCRIPTION |
|---|---|
| sp_ChangeCurrentUserPassword | Change the password for the currently connected user. |
| sp_DeleteReplicationEntry | Delete an entry in the replication queue. If part of a transaction, delete all records of the transaction from the queue. |
| sp_GetLinks | Enumerate the dictionary links defined for the root dictionary. |
| sp_GetQueryLoggingResults | Return information about the queries currently being logged. |
| sp_GetReplicationEntryDetails | Get details about a record in the replication queue. |
| sp_mgGetCrashDumpInfo | Get a list of the crash dump files from the root dictionary. |
| sp_mgGetErrorLog | Get information from the ads_err.dbf and ads_err.adt tables from the root dictionary. |
| sp_PackAllTablesOnline | Initiate online table packing for all tables. |
| sp_PackTableOnline | Initiate online table packing for one table. |
| sp_ReindexOnline | Initiate online reindexing on a table. |
| sp_TestReplicationConnection | Test the ability to connect based on a replication subscription. |

**Table 3.** New System Stored Procedures

Many of these stored procedures, like many of those introduced in previous versions of Advantage, require that they be executed from a user account with specific privileges. For example, sp_PackTableOnline is only available if executed from a user account having the DB:Admin role or appropriate effective permissions.

### Other SQL Enhancements

In addition to the SQL features mentioned above, there are several other enhancements to Advantage SQL. For one, there is now support for SQL scripts larger than 32 K in size for SQL-based views, triggers, stored procedures, and user defined functions.

Advantage now also supports transaction logs larger than 4GB in size, as well as SQL intermediate files greater than 4GB (in both 32-bit and 64-bit environments). While few users will be affected by these changes, those users who do need such large operations will no doubt welcome the removal of these limitations.

## CONNECTION ENHANCEMENTS

There are a number of enhancements found in Advantage 11 that are associated with connections. These are discussed in the following sections.

### Connection Pooling

Connections made through the Advantage Client Engine (ACE), which is used by most of the Advantage connectivity options, now support connection pooling. When pooling is enabled, the Advantage Client Engine will not automatically close a connection that has been returned from a client application. Instead, the connection may be kept active for a period of time, making it available if another client requests a connection.

Connection pooling is particularly useful in situations where many different connections are required for short durations. This scenario is particularly common in Internet applications. Making and dropping connections to a database tend to be expensive operations, and connection pooling goes a long way to eliminating a significant bottleneck.

Connection pooling is enabled in the Advantage Client Engine through the new Pooling connection string parameters. When Pooling is set to True, connection pooling is enabled. The default is False. Additional connection string options that apply to connection pooling include Connection Timeout (how many seconds a client will be permitted to wait for a connection), Connection LifeTime (how long a connection in the pool will remain open), Max Pool Size (the maximum number of connections to maintain in the connection pool), and Unused Timeout (how long an unused connection in the pool will exist before being closed).

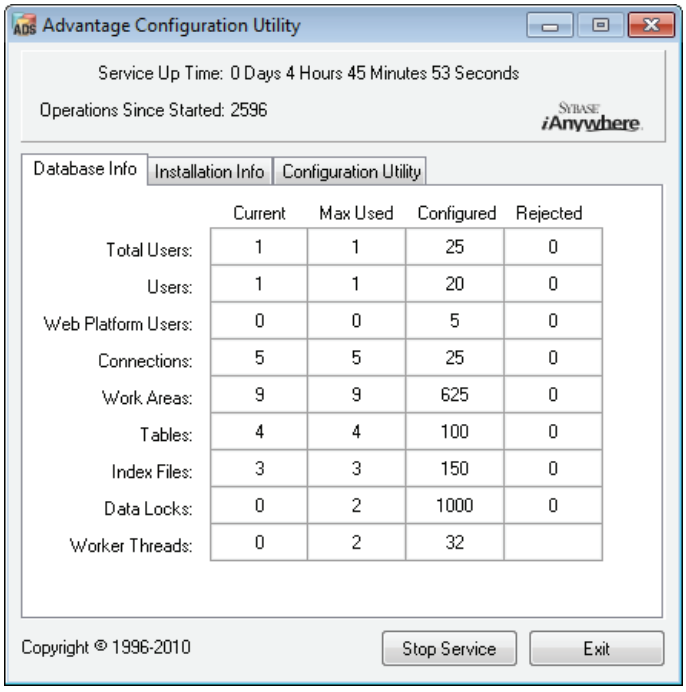
### Other Connection String Options

In addition to connection pooling, the Advantage Client Engine introduces a number of additional connection string options. All of these options are supported by the new AdsConnection101 API. These connection string parameters include Decimals, ShowDeleted, and DateFormat.

### Web Versus Traditional Connections

In previous versions of Advantage, all connections were treated similarly, in that they all consumed one of the available licensed connections. For example, a connection from an Internet-based application consumed a connection that could have also been used by a traditional client application (though in many cases, there were additional licensing issues that either limited or prohibited the use of Advantage from an Internet-based application).

With Advantage 11, you can specifically designate how many connections are allocated for Advantage Web Platform versus traditional connections. This can be seen in the following figure, where the Maximum Web Platform Users option permits you to limit how many of the connections are consumed by Web Platform operations.




Even if you do not configure a Web connection (leaving Web Platform Users set to zero), Advantage allows one connection by the Advantage Web Platform for connecting the Advantage Web Administrator for configuration purposes. If you are also using the Advantage Web Platform for its OData connectivity, you will definitely want to set Maximum Web Platform Users to a positive integer, to accommodate the additional connections through the RESTful service.

One thing to keep in mind when configuring your Maximum Web Platform Users is that clients that connect through the Advantage Web Platform using OData do not share connections from the connection pool in a manner typical of other Internet-based architectures. Instead, these users are treated as normal clients, and each individual connection through OData consumes one user license, and that license remains in use by that connection until that user disconnects, or the connection times out. In other words, you should set Maximum Web Platform Users to the maximum number of OData clients that your system is designed to support.

## DEVELOPER-RELATED ENHANCEMENTS

While Advantage has always been notable for its many connectivity options, this release of Advantage has seen a significant increase in these options. As discussed earlier, the Advantage Web Platform, and its OData support, permits access to your data from almost any development environment.

While this is the case, Advantage's language and environment specific access mechanisms tend to provide direct support that greatly exceeds the "lowest common denominator" approach offered by open specifications such as OData. As a result, it is significant that Advantage has introduced additional connectivity options in Advantage 11.

These connectivity options can be divided into two groups: new connectivity options and enhanced connectivity options. These are discussed separately in the following sections.

### New Connectivity Options

In addition to the connectivity options enabled by the Advantage Web Platform's support for OData, Advantage 11 introduces three new developer libraries. Together, these provide access for developers using Ruby, Python, and PHP.

There are actually two APIs (application programming interfaces) introduced in Advantage 11 intended for Ruby developers. The first is the Advantage Ruby API, and it provides Ruby developers access to the Advantage Client Engine API. The second is a package for Ruby developers that uses the Advantage Ruby API, and it implements ActiveRecord, an object-relational mapping (ORM) API commonly associated with Ruby on Rails.

The new Advantage Python Driver gives Python developers access to Advantage. There is also now an Advantage Django Backend, a package that uses the Advantage Python Driver to enable Advantage access from Django, a Python Web framework that supports RAD (rapid application development) for Web sites.

Advantage 11 also introduces a new PHP driver. While Advantage has supported PHP for many years with its Advantage PHP Driver, a new driver, the Advantage PDO (PHP Data Objects), is also available, increasing Advantage's support for PHP development.

### Enhanced Connectivity Options

Some of Advantage's existing connectivity options have been updated with the release of Advantage 11. For example, The Advantage RDD (replaceable database driver) for Visual Objects now supports parameters in the AdsServerObject class. In addition, a 64-bit version of the Advantage TDataSet for Lazarus is now available. Lazarus is a free IDE (interactive development environment) for the Free Pascal Compiler.

The Advantage .NET Data Provider also received an upgrade. The templates for creating Advantage AEPs (Advantage Extended Procedures) and triggers using C# for .NET and Visual Basic for .NET have been improved.

But most of the enhanced data connectivity features can be found in Advantage's support for Delphi. Advantage 11 provides support for Delphi XE2, both 32-bit and 64-bit Windows compilation. This means that you can now build 64-bit AEPs and triggers for 64-bit Advantage using Delphi.

Another improvement introduced recently in Delphi, delayed loading of DLLs, has also now been implemented in the Advantage Delphi Components library. This feature allows the Advantage components to delay loading of the ACE DLLs until they are first needed, as opposed to being loaded statically when your Delphi application is initializing. Delayed static loading of DLLs was introduced in Delphi 2010.

ADS 11 introduced a number of new connection string options, which are represented by the AdsConnection101 interface. These are supported in the TAdsConnection component found in the Advantage Delphi Components package, specifically from the ExtraConnectionString property. The enhanced TAdsConnection component also introduces a new DateFormat property, which permits your application to define the default format for dates on a per connection basis.

Advantage has also improved its support for the TDataSet Locate method. This method now makes more efficient use of available indexes, improving the performance of Locate invocations.

## ABOUT THE AUTHOR

Cary Jensen is Chief Technology Officer of Jensen Data Systems, a consulting, training, development, and documentation company. Since 1988, he has built and deployed database applications in a wide range of industries. Cary is the best selling author of more than 20 books on software development, including three books on the Advantage Database Server. A frequent speaker at conferences, workshops, and seminars throughout much of the world, he is widely regarded for his self-effacing humor and practical approaches to complex issues. Cary has a Ph.D. from Rice University in Human Factors Psychology, specializing in human-computer interaction.

## ABOUT SYBASE

Sybase, an SAP company, is an industry leader in delivering enterprise and mobile software to manage, analyze and mobilize information. We are recognized globally as a performance leader, proven in the most data-intensive industries and across all major systems, networks and devices.

## ADDITIONAL IMPROVEMENTS

The updates, additions, and enhancements that I've mentioned up until now have applied to specific areas of developer concern. But there are additional improvements found in Advantage 11 that I am categorizing as miscellaneous.

One notable update that has been found in all of the preceding releases of Advantage is once again present in Advantage 11, and it is related to performance. These improvements are scattered about the product, and will effect your applications depending on which of Advantage's features you use. But one of these enhancements, in particular, will effect nearly all Advantage installations. Advantage 11 has updated expression parsing, and this provides a significant improvement in the speed of opening tables and indexes. And on some platforms, such as Windows Server 2008 R2 and those Advantage installations under significant concurrent load, the performance improvement is quite large.

Another area of general improvement is related to error messages. A number of error messages have been updated, providing additional information not previously available. For example, some error messages that previously reported a problem with a field now actually include the name of the field where the problem has occurred, a welcome update that will be appreciated by developers.

The detection of corruption in memo fields in Advantage 11 has also improved. And, when a problem is detected, an appropriate exception is raised, alerting you to the issue and helping you to resolve the problem by packing the table, or taking some similar preventative measure.

Another error message-related update is revealed by the new option for database restore operations to suppress warnings associated with automatic table creation. Many developers do not consider these to be actionable errors. Warnings that were previously created when an auto-create table was being created can now be suppressed.

Advantage 11 has also introduced a number of enhancements related to security and encryption. For example, the AdsBackup utility now supports FIPS (Federal Information Processing Standard) and SSL (secure socket layout) options, as do many of the Advantage client libraries. Note that support for FIPS 140-2, a government standard that defines security standards for encryption support, requires the FIPS Encryption Security Option Add-on, which is a separate license available from Advantage.

Finally, there are two new roles that can be applied to users and groups. These are administrative in nature. SERVER:Admin is a role that provides the users to which it is applied the rights to a number of administrative tasks associated with the new root dictionary when connected with the Advantage Web Administrator. Likewise, SERVER:Monitor is a role that provides its users with enhancement information gathering, again when connected to the root dictionary.

See "What's New in Advantage 11" in the Advantage help for a complete list of new and enhanced features found in Advantage 11.

## SUMMARY

Similar to previous releases of Advantage, Advantage 11 comes filled with a wealth of new features that continue the tradition of high performance and ease-of-use. Some of these features, such as the support for online maintenance and the Advantage Web Platform, will fundamentally change the way that some users manage and access their data. Others, such as the default fetching of stored procedure parameters and variable output parameters, will dramatically improve developer productivity. These features, when taken together, make Advantage 11 yet another in a string of must have upgrades.

Sybase, Inc.
Worldwide Headquarters
One Sybase Drive
Dublin, CA 94568-7902
U.S.A
1 800 8 sybase

www.sybase.com



SYBASE®
An SAP Company