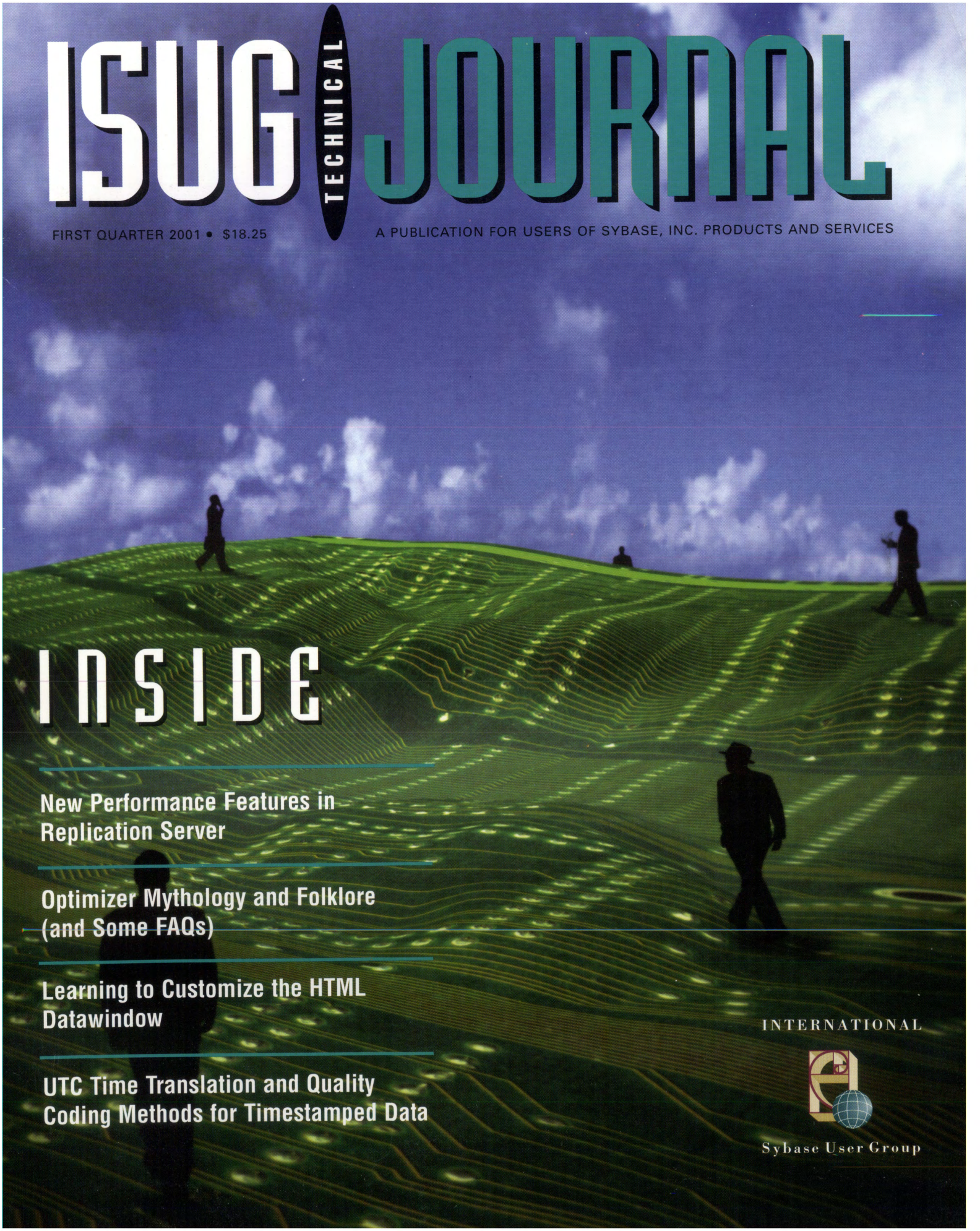
ISUG TECHNICAL JOURNAL

FIRST QUARTER 2001 • $18.25

A PUBLICATION FOR USERS OF SYBASE, INC. PRODUCTS AND SERVICES



INSIDE

New Performance Features in Replication Server

Optimizer Mythology and Folklore (and Some FAQs)

Learning to Customize the HTML Datawindow

UTC Time Translation and Quality Coding Methods for Timestamped Data

INTERNATIONAL

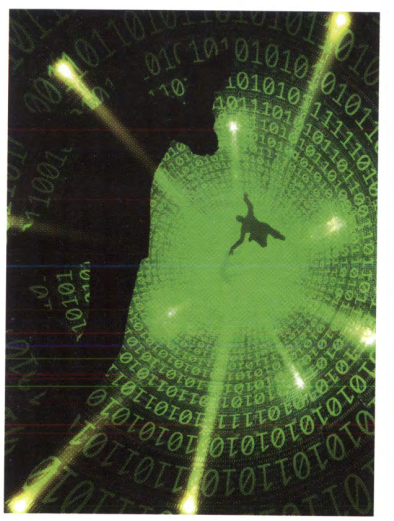
Sybase User Group

PRESIDENT'S MESSAGE

Dear ISUG Members,

Just one year ago I was welcoming everyone to the new millennium. Whether you believed it was last year or this year, it is definitely upon us now. With the recent downturn of the stock market, and the collapse of a significant number of the "dot coms," we are finally seeing the return of some older values and judicious amounts of common sense. In a recent speech made by Lou Gerstner, Jr. of IBM, he stated that it was time to drop the use of "e" and get back to making certain that sound business strategies are at the core of everything we do.

Sybase continues to be well positioned to provide these fundamental principles. Last year we saw the launch of iAnywhere, and during our fourth quarter board meeting we officially recognized this spin-off company and its president, Terry Stepien, by making iAnywhere a permanent ISUG partner.



During the same meeting, we also outlined several new initiatives that you will be hearing about during the coming months. These include:

1. Meeting-in-a-Box. ISUG will be working to put together several kits of materials that can be sent out for use in user group meetings. These kits will include presentations, general and frequently asked questions, marketing materials, and handouts.
2. Value Recognition for the *ISUG Technical Journal*. We are planning to have a price associated with the journal, thereby demonstrating real value for the membership.
3. The ISUG Innovation Tour 2001 in Europe.
4. Increased enhancement process activities with the Internet Applications Division.

For more on these and other initiatives, see the Board of Directors report by Kathy Ridley on page 34.

ISUG also ended 2000 with the election of its new Board of Directors for 2001. Information about the new directors can be found on page 35, with a directory on page 38. I would like to thank last year's members for their dedication and efforts which have contributed so much to the success of ISUG in the past, and hope that their continued membership will help to take our organization to new levels.

We also welcome some new board members who bring with them fresh ideas and concepts. I would like to recognize Philip Brantley as our new Enhancements co-director (Database); Bryan Enochs as our new Special Interest Group co-director (Tools); North America regional co-directors Richard Brooks and Brian Pollard; and Ramesh Babu Ve as the new co-director for the Pacific-Asia region. In addition, Cynthia Gill, who has done such great work as the ISUG conference director, is now the director of the *ISUG Technical Journal*.

Remember, this year there will not be an Australasian Conference. However, the North American conference has been moved to the West Coast and will take place on August 12–16, 2001 at the San Diego Convention Center. Sybase and the ISUG Board of Directors are hoping that positioning the conference in San Diego will make it convenient for more of our Australasian members to participate. Be sure to register early for this exciting event, which has the theme of "e-Business. Anytime. Anywhere."

Although no specifics are currently worked out, the Hong Kong office may also once again be sponsoring a conference in China or the Asia-Pacific area. Keep checking the Sybase corporate and Hong Kong websites for more details, as well as the ISUG website.

We on the ISUG board look forward to continuing to work with you, our members, providing the benefits, the opportunities, and the knowledge you need to make your relationships with Sybase and each other pleasant and profitable. All the best for 2001!

All the best for 2001,
Thom Lamb
ISUG President

Table of Contents

FEATURES

New Performance Features in Replication Server 2
By Irfan Khan

Optimizer Mythology and Folklore (and Some FAQs) 7
By Eric Miner

Learning to Customize the HTML Datawindow 12
By Jay Hunt

UTC Time Translation and Quality Coding Methods for Timestamped Data 18
By Mark Gearhart

ISUG Welcomes Board of Directors for 2001 35

DEPARTMENTS

Server Views *Ian Smart* 26

PowerBuilder Tips & Techniques 30
Thomas Lamb

Board of Directors Report 34

READER INFORMATION

ISUG Membership Application 37

ISUG Board Directory 38

Sybase User Group Directory 39

ISUG Technical Journal

ISUG Technical Journal Director
Cynthia Gill
CGI
713.868.5537 x1409
cynthia_gill@drthou.com

Managing Editor
Mary Freeman, Freeman Communications
510.525.4863
510.528.6958 Fax
Mary_Freeman@compuserve.com

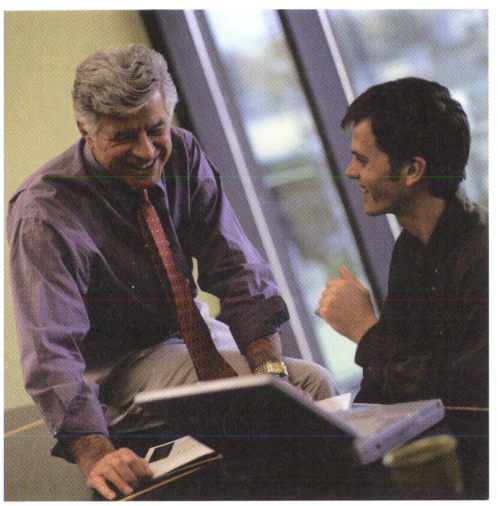
Art Direction
Jerry Jager, JagerCreative
jagercreativ@earthlink.net

Senior Technical Editor
Al Huntley, EDS

Technical Editors
Teresa Larson, Vetcentric.com
Thomas Lamb, Accenture
David Straiton, Sybase, Inc.

New Performance Features in Replication Server 12.1

By Irfan Khan



The upcoming release of Replication Server enhances monitoring and disk affinity features

Irfan Khan is a senior technical architect for Sybase's enterprise solutions division, with a special focus on Replication Server. He can be reached at irfan.khan@sybase.com.

The first few months of 2001 are poised to become a landmark period for Sybase's Replication Server. In the past, many users have commented on the great functionality and flexibility Replication Server provides for heterogeneous environments. However, amid these accolades there still remains a desire in the user community for functionality that is not currently available: to be able to better understand what is happening inside the Replication Server and to be able to apply this knowledge to providing better levels of transactional throughput.

In Replication Server 12.1, the primary focus is performance. In this article I will outline a couple of the areas of new functionality that are squarely aimed at addressing these desires of Sybase Replication Server users—Monitors, Counters, and Disk Affinity.

Monitors and Counters

Sybase Replication Server now includes a set of performance counters that provide metrics about what the Replication Server is doing and how it is performing. These are gathered by the Replication Server during performance of its normal runtime operations. The counters are always active and therefore available to be viewed at any time, using extensions to the existing **admin statistics** command.

The sampling command interface is displayed below.

As can be seen, the new performance infrastructure utilizes the pre-12.1 command **admin statistics**. However, this command has been further enhanced to allow users to simulate the functionality that is already available to users of ASE through the *sp_sysmon* stored procedure. This is perhaps most easily seen in relation to the first example in the above table, which highlights the use of the literal *sysmon* that can be further supplemented with a sample interval.

The following sections describe in greater detail the various types and clarifications of counters, in addition to the user interface that has been introduced for the persistent storage of Replication Server counters.

Command Syntax	*Description*
Admin statistics, {module_name, "all_modules" | sysmon [, sample_time]}	Display counters of one or all modules.
Admin statistics, flush_status	Display current flush settings (persistent storage).
Admin statistics, intrusive_counter_status	Display current intrusive counter setting and list available intrusive counters.
Admin statistics, reset	Reset all counters.

Counter Classification

As has been stated, the Replication Server now performs active sampling of counters.

The various classifications of counters that are sampled are described in the following table.

Counter Type	*Description*
Total	Accumulative total of the sampling data.
Last	Last value of the sampling data.
Max	Maximum value of the sampling data.
TotalRate	Total values of the sampling data divided by the thread or module duration. Duration is measured in one-hundredth of a second.
Current Rate	Total values of the sampling data divided by the sampling duration. Sampling duration is measured in one-hundredth of a second.
Duration	Time duration measured in one-hundredth of a second.

Counter Types

The most common form of counter is classified as "generic" in functionality, that is to say, no adverse performance degradation is expected. To provide a greater level of performance analysis, a further set of counters can be called upon, termed "intrusive counters." These counters provide additional insight into the performance profile of a running server, but they can carry a processing penalty. Sampling is not performed for these counters by default, and therefore they only provide information from the point in time at which they were enabled.

Counter Type	*Description*
Intrusive	Counters that may impact Repserver performance. By default, intrusive counters are not configured to sample.
Internal	Counters that are used by calculated counters. Internal counters are not visible to end-users.

Sampling Overhead

A series of internal benchmarks of the new counter functionality has revealed that no noticeable performance degradation has arisen as a result of the use of the generic counters. This may sound optimistic, but if you consider that Replication Server essentially comprises of a pipeline that utilizes a component-based architecture in the shape of various modules (e.g., REPAGENT, SQM, SQT, etc.) and if you then bear in mind that active sampling only introduces additional CPU cycles into that pipeline-constrained architecture, then it can be seen that in the case of the Replication Server this potential performance issue is removed through the rigidity of the pipeline. Since some modules are inherently faster than others, introducing sampling for the faster modules acts as a break in the pipeline, allowing the throughput to remain balanced.

Persistent Storage of Sampled Counters

Replication Server monitor counters can optionally be flushed (or saved) to the RSSD with minimal performance overhead. The process of flushing counters to the RSSD is managed by a new Replication Server daemon thread called *dStats*. There is no overhead on the ASE tempdb, as *dStats* uses physical memory to stage the flushing process.

dStats Daemon

The *dStats daemon* is a self-contained thread that is constantly active whenever the Replication Server is running. Normally the dStats is in a sleeping state. However, at a configured time interval it is awakened to perform the configured activity. If a module is configured to flush its counters to the RSSD, the dStats thread will walk through the module counters, making a copy into its own memory space before initiating the flushing process to the RSSD.

Once the counters have been flushed to the RSSD, users will have the ability to interpret the profile of an executing Replication Server. The current intention is that within the Replication Server 12.5 timeframe, the Replication Server Manager will be enhanced to provide a graphical interface to sampled counters. However, in the initial release, this information and analysis will be provided through the use of a set of stored procedures that are executed within ASE against the new RSSD tables.

New RSSD Tables

There are three new RSSD tables for recording, sampling, and storing available counters. These tables are not referenced directly by Replication Server, but are used by external scripts for calculating performance statistics:

- *rs_statcounters* is for counter descriptions. One row per counter.
- *rs_statrun* is for sampling period (run) descriptions. One row per run.
- *rs_statdetail* is for counter values. One row per run and configured counter.

By default, Replication Server does not automatically flush statistics to the RSSD. This was a conscious design decision, since prolonged periods of flushing introduce risks of filling the data or log segments of the RSSD. Consequently, the counter configuration is designed to limit the output sent to the RSSD. New commands and configuration parameters provide control over how enthusiastically this flushing of metrics to RSSD tables takes place. As a further optimization, Replication Server does not flush counters that have a value of zero.

There is a hierarchical configuration structure that limits the flushing of any counters to RSSD. This is:

New Configuration Parameters

Parameter	*Description*
stats_sampling	Enable sampling counters. Default: on.
stats_flush_rssd	Enable flushing counters to RSSD. Default: off.
stats_reset_afterflush	Enable reset counters after flushing. Default: on.
stats_daemon_sleep_time	Number of seconds between RSSD flushes. Default: 3600.

As a primary activity, the users must first enable the flushing functionality within the Replication Server using the new configuration parameter *stats_flush_rssd*, followed by enabling the desired specific module(s) flushing state.

Replication Server Modules

Replication Server manages three types of modules:

- **Multi-threaded modules**: modules that may have multiple thread instances. Examples of these would include: SQM, SQT, EXEC, DIST, DSI, and RSI.
- **Single-threaded modules**: modules that have only one thread instance. All Replication Server daemons are single-threaded modules.
- **Non-threaded modules**: modules that have no thread instance. Examples of these would include MD, TD, SRE, STS, CM, and MEM.

Relationship between Modules and Counters

Each and every counter in replication server is unique. Therefore, its function is tightly bound to a particular module. For multi-threaded modules, each thread has one instance of a module counter. Single-threaded modules and non-threaded modules have only one instance of module counters. Some non-threaded (e.g., SRE) counters are collected by threaded modules (e.g., DIST). In this case, the non-threaded (SRE) counters are a part of the collecting module (DIST). These counters do not belong to the non-threaded modules (SRE).

New User Interface for Configuring Flushing

For multi-threaded modules, users must specify individual thread flushing. All flush settings are dynamic and do not retain their values when Replication Server is recycled. To configure the process of individual threads flushing their counters to the RSSD, a set of new RCL commands has been introduced to control flushing. These are:

Task	*Command Syntax*
Module flush control	**Admin stats_config_module,** {**module_name**|"all_modules"}, {"on"|"off"}
Thread flush control	**Admin stats_config_connection,** {**data_server, database**|"all_connections"}, {**module_name**|"all_modules"}, ["inbound"|"outbound"], {"on"|"off"} **Admin stats_config_route,** {**rep_server**|"all_routes"}, {**module_name**|"all_modules"}, {"on"|"off"}
Intrusive counter control	**Admin stats_intrusive_counter,** {"on"|"off"}

Demonstration of User Commands

To demonstrate how a user would use this new feature, the following example outlines the necessary steps:

1. Configure Replication Server to flush counters:

```
configure replication server:
set 'stats_flush_rssd' to 'on'

configure replication server
set 'stats_reset_afterflush' to 'on'

configure replication server
set 'stats_daemon_sleep_time' to '600'
```

The *stats_flush_rssd* is the highest level control that dictates if the Replication Server is configured to flush its counters to the RSSD. By default this parameter is "off," implying no persistent storage of counters.

The *stats_reset_afterflush* determines whether counters are reset to zero after flushing. If this parameter is set to "on," all counters are reset. The default is "off." By resetting counters after flushing, you can reduce the number of counters that are saved on subsequent flushes. Replication Server only flushes counters that have a non–zero value. Even if the counters are not reset, reaching a maximum value is unlikely – all counters are 32 bit INT (signed) and the only threat for overflowing a counter is if MAXINT is reached. Since Replication Servers are typically recycled more frequently than ASE, this is not much of a threat.

The *stats_daemon_sleep_time* determines how often the counters are saved to the RSSD. This parameter is configured in seconds. Smaller values cause more frequent flushing and more data to be saved to the RSSD. The default is 120 seconds.

2. Restart the Replication Server.
3. To enable the flushing of the counters for all modules, associated with both routes, and database connections:

```
admin stats_config_module, all_modules, 'on'

admin stats_config_connection, all_connections,
all_modules, 'on'

admin stats_config_route, all_routes,
all_modules, 'on'
```

To enable the flushing of only the inbound SQM thread managing the PDS.pdb connection:

```
admin stats_config_module, SQM, 'on'

admin stats_config_connection, PDS, pdb, SQM, 'on', inbound
```

4. Check the status to confirm flags are set.

```
admin statistics, flush_status
```

Examples of Available Counters

There are many new counters that have been added to the Replication Server. Unfortunately, there is not sufficient space here to list them all. However, just to give a flavor of the types of monitoring that are possible, here are a selection of counters for a small subset of modules.

DSI-EXEC Module

If a connection has been configured to use the *parallel_dsi* option, the counters listed below are captured for each configured DSI executor thread. This is particularly useful if you need to determine the effectiveness of the parallel property.

Counter	*Module*	*Description*
DSI executor grouped tps current	DSI-EXEC	The current number of transactions being processed per second
Last xact process time	DSI-EXEC	The time taken in milliseconds to process the last transaction
Maximum xact processing time	DSI-EXEC	The maximum time taken in miliseconds to process a DSI transaction
DSI rs_writetext total bytes written	DSI-EXEC	The total number of bytes written by the rs_writetext command

DIST Module

The Replication Server starts a DIST thread for each primary site. To allow users to understand the individual parts of the DIST thread, (namely SRE, TD, and MD), performance counters have been added to determine the profile of subscription migration and SRE processing.

Counter	*Module*	*Description*
SRE **resolve insert** statement	DIST-SRE	The total number of DIST resolved insert commands
SRE **resolve delete** statement	DIST-SRE	The total number of DIST resolved delete commands
TD **begin transaction** commands	DIST-TD	The total number of **begin transaction** commands propagated
DIST rs_locater updates	DIST	The number of updates performed against the rs_locater table by the DIST threads. The DIST thread will perform an explicit synchronization each time a SUB RCL command is executed.

SQT Module

A subset of the SQT counters that are available are:

Counter	*Module*	*Description*
Last transaction command count	SQT	Total number of commands scanned in previous transaction
Current SQT memory usage	SQT	The Current SQT memory utilization. Each command structure allocated by SQT, i.e., SQT_TRAN and SQT_STMT is freed when a transaction context is removed. For this reason, if no transactions are active in SQT memory, SQT cache memory usage will be zero.
Last transaction memory usage	SQT	Total number of bytes used from SQT memory by previous transaction
SQT transaction removed count	SQT	Since SQT thread commenced processing, the number of xacts whose constituent messages have been removed from memory, causes of xacts being removed is most commonly caused by a single xact exceeding the available sqt_max_cache_size.

Disk Affinity

An often-requested functional capability for the Replication Server is a data placement framework. In its 12.1 release, the Replication Server provides this functionality using the new Disk Affinity feature. This makes it possible to specify to which disk partition Replication Server should allocate a segment associated with a stable queue. By choosing the stable queue placement, it is possible to achieve better enhanced load balancing and read/write distribution.

When performing a fresh install of Replication Server, **rs_init** assigns Replication Server's initial partition. Since a single partition is limited to 2GB, it quite possible that more partitions will be added as users increase the number of databases and remote sites to which the Replication Server distributes messages.

Replication Server's default allocation mechanism assigns queue segments to the first partition in an ordered list of partitions. With the next allocation, the first partition becomes the last partition, and the queue segment is allocated to the new first partition. This rolling allocation of segments is automatic and does not require any manual intervention.

You can now choose the segment allocation using the **alter connection** and **alter route** commands. The syntax is:

```
alter connection to dataserver.database
set disk_affinity to [ 'partition' | 'off' ]

alter route to replication_server
set disk_affinity to [ 'partition' | 'off' ]
```

where partition is the logical name of the partition to which you want to allocate the next segment for the connection or route. Each allocation directive is called a "hint" because Replication Server can override the allocation if, for example, the allocated partition is full or has been dropped.

When Replication Server overrides the hint, it allocates segments according to the default mechanism described above. Each hint is stored in a new RSSD table called *rs_diskaffinity*. Currently, users are only able to associate a single hint per stable queue. Replication Server checks for an allocation hint each time it allocates a new segment for a queue. When a stable queue is being migrated from one partition to another as a result of an allocation hint, Replication Server displays an informational message confirming the migrating queue allocation; this records both the old and new partition.

There is only a minimal overhead for configuring disk affinity. Since the table *rs_diskaffinity* is cached in repserver memory, the cost of checking for a hint is one logical I/O every time a new segment is to be allocated. One segment equals only 1MB, so that the duration for a typical segment allocation, depending on throughput, is just 500ms-5000ms.

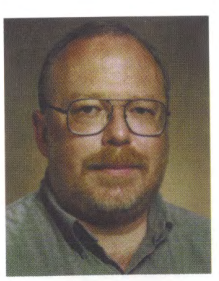
Other Upcoming 12.1 Release Features

There are a number of other performance features included in the 12.1 release. Look out for the next issue of the *ISUG Technical Journal* for additional new features in Sybase Replication Server 12.1. Note that at TechWave this year, there will also be a presentation on "Performance Tuning for Replication Server," which will cover the monitors and counters interface in considerable detail. ■

Optimizer Mythology and Folklore (and Some FAQs)

By Eric Miner

Dispelling the fables that lurk in the darkest areas of ASE optimizer



Eric Miner has been with Sybase since 1992, working with the optimizer team in engineering, as well as focusing on optimizer issues for technical and product support engineering. He can be reached at eric.miner@sybase.ocm.

Now that we've entered the 21st century, let's take a look at some leftovers from the 20th. We've discussed a number of aspects of the optimizer over the past year. Let's take a look at some shorter but equally important topics.

Over the years, mythology and folklore have developed around a number of areas of ASE. One area that has more than its share of these myths is the optimizer. Strange stories spread far and wide, sometimes very quickly. Sometimes they get repeated so many times they're unquestionably accepted as truth. Sometimes they're even put into Sybase training materials. In this article, I'll try to set the record straight on the more common ones.

Why discuss myths and folklore? They need to be explained in order to provide you with a better understanding of how the optimizer works and its true behavior. This article also includes answers to some optimizer-related FAQs.

Myth #1: The 20% Rule

"If more than 20% of a table is returned by a query, the only choice the optimizer has is to table scan."

This is a very old myth. As the story goes, many years ago a senior engineer answered a question at an ISUG conference. The question itself is lost in the mists of time. The off-the-cuff answer was something like, "Hmm, well, if about 20% of a table (read column) will be returned by a query, a table scan will be done. The percentage varies at times, but it is always around 20%." This myth was picked up quickly. It was quoted by Technical Support, consultants, and even put into the manuals.

Since we don't know the exact question that was asked, it's hard to know exactly what the former engineer was referring to. However, an educated guess would be that he was talking about the pessimistic costing of non-clustered indexes in pre-11.9.2 versions of ASE. Basically, this costing assumed that every read from the leaf of the index to a datapage would cost one I/O. In other words, for every qualifying row in the index, one I/O would be done. This, of course, was not the best assumption to make because it didn't take into account that more than one qualifying row may exist on the same datapage.

Even if he was talking about the pessimistic non-clustered index costing, the "20% Rule" still does not hold true. The optimizer's decision of which access to use in order to get the required rows is based on the estimated cost, not on a set percentage of the column. The existence of covering indexes is one way to disprove the "rule." Another is the fact that a clustered index can easily be chosen to return values based on SARGs and joins of columns.

In 11.9.X and above, the Data Row Cluster Ratio is used to measure how well clustered a column's rows are in relation to the data pages. This value is used when estimating the cost of using a non-clustered index. With it, the optimizer can estimate how may qualifying rows can be read in a single I/O.

While the "20% rule" may have seemed to be a reasonable rule of thumb at the time it began, it has never accurately described the optimizer's behavior.

Myth #2: Update Statistics Improves Performance

*"**Update statistics** will give you good performance."*

Update statistics is only guaranteed to result in statistics that are up-to-date as of the end of its run. It does not guarantee good performance. For example, if the values of a column are dense (a gender column is an extreme example), **update statistics** can't change that fact. It can only record the data distribution it sees.

The optimizer depends on the statistics as the only view of its universe. As the statistics change, so can the optimizer's view. Sometimes the change is fine and plans are very good, sometimes not. It's possible that you'll see plans change drastically after **update statistics** is run because the optimizer's view of the data has changes a great deal. This is why it's a good idea to keep a copy of the statistics in the form of an optdiag output file, just in case. It all depends on your dataset, how it changes and what your queries are doing.

Myth #3: When to Run Update Statistics

*"**Update statistics** should be run whenever the data changes by 5-10%."*

This is not necessarily right or wrong. In general, it's a fine guideline. How often to run **update statistics** is completely dependent on changes to the distribution of values in your dataset, the work being done, and the efficiency of the query plans generated by the optimizer. Some datasets whose distribution changes regularly may need to have statistics updated very often, others less often. And, some datasets may never need to have **update statistics** run in order for the optimizer to generate the most efficient query plans. You need to test and monitor performance in order to determine the optimal intervals in which to run **update statistics**.

A rule of thumb such as this isn't harmful, but you may be doing more maintenance work than is necessary.

Myth #4: Update Statistics Does Cleanup?

*"You should always use **update all statistics**" or "**Update all statistics** will 'clean up' all your statistics."*

The **update all statistics** command will gather or update statistics on all columns of the specified table. It will also update partition statistics.

It is not likely that you'll need statistics on all columns of a table. You should conduct complete tests to determine if statistics on all columns of a table are necessary. Adding statistics to columns that previously didn't have them is likely to result in different query plans. You need to verify that the additional statistics are helping the optimizer produce more efficient plans. While it's recommended that you consider adding statistics to inner columns of composite indexes and/or to non-indexed columns, always run tests before putting any new **update statistics** syntax use into production.

Since **update all statistics** must gather statistics on all columns of the table, it will add a great deal of time to your statistics maintenance. If you want to update your partition statistics, use **update partition statistics**.

Myth #5: Deleting Statistics after Upgrades

"You should delete statistics after upgrading to 11.9.X (or above) from an earlier version."

This is a relatively new myth. It began as a solution to a "print bug" in early 11.9.2, which simply resulted in annoying messages in optdiag and trace 302 output saying that the column's statistics were obtained from upgrade. The message was inaccurate because **update statistics** had been run in the newer ASE and the statistics were in the new format. It was found that if the column's statistics were deleted and **update statistics** rerun, the message went away.

The problem with deleting the statistics and rebuilding them is that you lose the number of requested steps that was inherited from the old distribution page. After deleting the statistics, if you don't specify a number of steps to use, the default of 20 steps is be used when rebuilding. This could result in Frequency count cells not appearing in the histogram because the cells are now too wide for highly duplicated values to get a cell of their own. Also, if there are range SARGs (<, <=, >, >=), the smaller number of steps may change the optimizer's cost estimates, because the cells are now wider.

Let's take a quick look at the sample histograms below. Let's say that the old distribution page had 200 steps for this column. That number is then used as the "requested steps" during upgrade and is marked as the step count for the column. Let's suppose we have a search clause in a query:

```
where column between 'July 1, 1996' and 'August 1, 1996'
```

This range of values would fall into cell (step) 4 of the histogram below (a cell is inclusive of its upper bound value and exclusive of its lower). The optimizer would then need to estimate how much of the cell would qualify in order to estimate the cost. When the histogram has 200 cells, they are approximately ten times narrower than when the histogram contains only 20. It will be easier for the optimizer to determine how close the search clause values are to the boundaries of the cell and thus how much of the cell's weight to use in the cost estimate.

200 steps (column contains a little over 300,000 rows):

Step	Weight		Value
1	0.00000000	<=	"May 29 1992 11:53:32:996AM"
2	0.00510867	<=	"Oct 20 1995 9:58:24:000AM"
3	0.00517948	<=	"May 7 1996 8:22:57:000AM"
4	**0.00516177**	**<=**	**"Oct 17 1996 12:32:37:000AM"**
5	0.00514407	<=	"Jan 6 1997 10:44:42:000AM"

Default 20 steps:

Step	Weight		Value
1	0.00000000	<=	"May 29 1992 11:53:32:996AM"
2	**0.05263028**	**<=**	**"Nov 18 1997 5:11:39:000PM"**
3	0.05275065	<=	"Aug 6 1998 12:40:39:000PM"
4	0.05264090	<=	"Feb 26 1999 10:33:32:000AM"
5	0.05263381	<=	"Dec 6 1999 5:37:33:000AM"

The best thing to do after upgrading to 11.9.X or above is to run **update statistics** as you did in the earlier version and then test a set of queries. In the majority of cases, there is no reason to delete the statistics or change the number of steps. See my Q1 2000 *ISUG Technical Journal* article "An Introduction to Utilizing Optdiag Output" for more information on the different types of histogram cell, how they're used, and their advantages.

Myth #6: Trace Flag Sets Optimizer Behavior

"There's a trace flag that will set the behavior of the optimizer back to that of (insert previous ASE version number)."

This folklore has been around for a long time. There is no single trace flag that will set all of the optimizer's behavior back to that of an earlier version, and there never has been one. Over time, a number of trace flags that effect individual optimizer behaviors have been introduced, primarily to address specific bug situations. Sometimes traces are removed in a new version of ASE; sometimes they stay around for many versions. They are usually not supported or documented in any official manner. I will discuss some of these optimizer related trace flags, in detail, in a future article.

Myth #7: Too Many Steps Hurt Performance

"Don't use too many steps when creating an index or updating statistics, because it will hurt performance."

This myth comes from the fact that whenever the optimizer needs to read the cells (steps) of a histogram for costing, it must first place them into procedure cache. Once the query is complete, the cells are removed from cache. The amount of cache used for each cell is minimal (two to eight bytes depending on the datatype). The reading of the cells into cache can indeed add a little time to the overall parse and compile time, but usually only a few milliseconds. These milliseconds are usually a good tradeoff for the more efficient plans that can be generated from the more granular statistics.

Don't be concerned about having a few hundred steps in a histogram (particularly one inherited from an old distribution page during upgrade). You won't notice the difference in parse and compile times, and the optimizer will have more statistics to work with. If you're tempted to request tens or hundreds of thousands of steps, you may want to reconsider why you need this many. The bottom line here is not to be too concerned; there are far more important things to think about when tuning.

Some Optimizer-Related FAQs

1. What is data skew and how can it effect the optimizer?

Data skew is a common term used to describe a column with values that are highly duplicated and other values that are not. In ASE 11.9.2 and above, the presence of both Frequency count and Range cells in the column's histogram indicates some degree of data skew. If all cells in the histogram are one type or the other, there is no data skew in the statistics. The column may be dense, but it's not skewed.

Is date skew bad? No, not really. It's a normal state of many datasets. As far as the cells in the histogram and thus the costing of SARGs are concerned, Frequency count cells are desirable. Because a Frequency count cell represents only one value, the weight of the cell measures exactly how many rows in the column are occupied by that value.

Data skew does have an effect on the column's total density value, which is used to estimate the cost of joins of the column. The total density value is created by using a geometric averaging method. One effect of this is that highly duplicated values in Frequency count cells have a proportionally greater effect on the total density value than those values that are not highly duplicated. Thus, values that may not be participating in the join have an effect on how the optimizer estimates the cost of the join.

By using optdiag, there are a couple of ways to check the distribution of values in a column and measure any data skew that may be present. The first is to compare the Total and Range cell density values. The Total density value is a measure of the average number of duplicates in the entire column. The Range cell density is a measure of the average number of duplicates in cells that are not Frequency count cells. If the values are the same, then there are no frequency count cells

in the histogram and thus no data skew. It is possible that the column may have some degree of data skew, but if there are not enough cells (steps) in the histogram to allow frequency count cells to appear, it will not appear in the statistics. If data skew doesn't appear in the statistics, it won't effect costing.

In this first piece of optdiag output, the two density values are the same. There is no data skew in this histogram:

```
Statistics for column:          "col_A"
Last update of column statistics: Dec 11 2000  1:11:37:693PM

    Range cell density:      0.0000576586096924
    Total density:           0.0000576586096924
```

Here the degree of data skew is due to the large difference between the two density values. Also notice that there is a large frequency count cell for value 1. In this case, it's responsible for the skew.

```
Statistics for column:          "col_B"
Last update of column statistics:   Nov 30 2000  1:58:00:906PM

    Range cell density:      0.0381101996119019
    Total density:           0.6521914581601986

    Step  Weight          Value
    1     0.00000000    < 1
    2     0.79821283    = 1
    3     0.10293456    < 2
    4     0.00565034    < 3
```

Here there is some data skew, but the degree is low. There were two frequency count cells with relatively low weights.

```
Statistics for column:          "col_C"
Last update of column statistics:   Nov 30 2000  1:57:47:890PM

    Range cell density:      0.0086235950083850
    Total density:           0.0762508697619250

Step   Weight          Value
   1   0.00000000    < 16
   2   0.02647445    < 32
   3   0.10705626    = 32
   4   0.00250300    < 33
   5   0.21902847    < 34
   6   0.04812700    < 60
```

What to do about data skew? See the Q3 2000 issue of the *ISUG Technical Journal* in my article, "Where, Why and How To Add Or Modify Optimizer Statistics."

2. Why Does Update Index Statistics table_name Take Longer Than update statistics table_name?

Let's take a look at the behavior of the "standard" syntax first: **update statistics table_name [index_name]**. In pre-11.9.2 versions of ASE, statistics were an attribute of an index and could only exist for the leading column. When **update statistics** was run, the index was read and the statistics gathered from there. This was done because the index is sorted in the leading column order.

In 11.9.2 and above, the column-level statistics are no longer associated with an index, but are now an attribute of a column. This means that they can exist on any column whether or not it is a member of an index.

In 11.9.2 and above, if you run **update statistics table_name**, it will behave the same as in earlier versions: it will scan the index to gather the statistics because the rows are in order. In 11.9.2 and above, **update statistics** has been extended to allow you to place statistics on columns other than the leading column of an index. There are three ways to do this:

- **update statistics table_name (column_name)**
  Will create or update statistics on the specified column
- **update index statistics table_name [index_name]**
  Will create or update statistics on all columns of all indexes in the table or the specified index.
- **update all statistics table_name**
  Will create or update statistics on all columns of the table, and updates partition statistics.

When the column that is having statistics either created or updated is not the leading column of an index, the values must be read from the column and sorted before statistics can be gathered. A table scan is done to read the values from the column. The sort will be done in a worktable in tempdb. Both of these, of course, add I/O to the process, and this is the bulk of the extra work that needs to be done.

If you use **update index statistics**, you may want to increase the size of tempdb. This is especially true if you use **update all statistics**. In most cases, the more efficient plans generated by the optimizer will outweigh the added maintenance cost (except possibly that of **update all statistics**).

3. What is the formatid column in sysstatistics, and what do the values in the column mean?

The *formatid* column in sysstatisics holds a value that describes what type of statistical value is stored in the rest of the row (columns c0-c79).

- foramtid 100: These are values that relate to the column. They include Total density, Range cell density, the number of requested and actual steps used in the histogram, the default selectivity values, the column datatype, length, scale and precision.
- formatid 102: The histogram's boundary values
- formatid 104: The histogram's cell weight values
- formatid 20: These are the simulated table/index level statistics that will be read when *set statistics simulate* is on, and simulated statistics exist.
- formatid 40: These are the simulated server configurations (cache sizes, total memory, max parallel degree, etc.) that will be read when *set statistics simulate* is on, and simulated statistics exist.

While formatid is a tinyint datatype, the columns that contain the values (c0-c79) are all varbinary. Keep this in mind if you want to query the table.

4. How can I tell how many steps were used, or how many steps were requested, for a column and the last time statistics were modified without having to run optdiag?

It's not always possible or practical to use optdiag to see the statistics. Also, if you're interested in automating things, optdiag is not going to be terribly useful. If you can't or don't want to use optdiag to determine the requested and actual step count, the query below will get the information for you. It will return the table name, the column name the number of actual steps used in the column's histogram, and the date of the last modification to the statistics.

```
select object_name(s.id) tablename,
c.name column_name,
convert (int, c4) actual_steps,
convert (int, c5) requested_steps,
moddate
from sysstatistics s, syscolumns c
where formatid = 100
and s.id = c.id
and colid = convert(tinyint, substring(colidarray,1,1))
and s.id >100
and s.c4 != NULL
and s.c5 != NULL
order by 1,2
```

5. How can I Read The Cluster Ratios Without Going to Optdiag?

The cluster ratio values you see in optdiag output under "Derived statistics" are values that have been computed internally. The exact formula for doing this is not publicly available. However, there is a way to view the cluster ratios without going to optdiag.

The "raw" numbers, or CR counts, that are used to derive the cluster ratios are stored in systabstats. Think of each as a count of "jumps." For the Data Page and Index Page Cluster Ratio, the measure is how many jumps between extents will have to be done in order to read the data or index pages in order. For the Data Row Cluster Ratio, the value is measuring how many jumps between data pages will need to be done to read rows in index leaf order.

While the database is static, run **update statistics**. Then run the query below. sp_flushstats will copy the most recent table/index level statistics from the in-memory copy to systabstats. In most cases, you won't need to run sp_flushstats, because Housekeeper will flush the values to systabstats as part of its regular work. This will give you a baseline to use when monitoring the values. From time to time, rerun the script below. As the values change, so do the cluster ratios. If the values increase the cluster ratio decreases (becomes less clustered) and vice versa if they decrease.

Note: If you want to get the table level CR counts and either the table is DOL or is APL and doesn't have a clustered index, drop the clause **and t.indid !=0**.

```
sp_flushstats table_name
go
select
i.name,
t.dpagecrcnt "DPCR Raw",
t.ipagecrcnt "IPCR Raw",
t.drowcrcnt "DRCR Raw"
from sysindexes i, systabstats t
where t.id = object_id("table_name")
and t.id = i.id
and t.indid = i.indid
and t.indid !=0
go
```

Learning to Customize the HTML Datawindow

By Jay Hunt

The last of a series of articles on using the datawindow to build applications for the Web.

In my last article on Jaguar, PowerBuilder, and PowerDynamo, the concept of customizing the HTML datawindow by adding method calls was introduced (Q4, 2000, *ISUG Technical Journal*). It mentioned that Sybase has delivered a pre-built component in anticipation of the developer's need. In concluding the last article, I suggested that it would be beneficial for developers to review this code, which can typically be found at C:\Program Files\Sybase\PowerBuilder 7.0\Code Examples\HTMLDW (Figure 1), in the non-visual object *nv_remote_datawindow*.

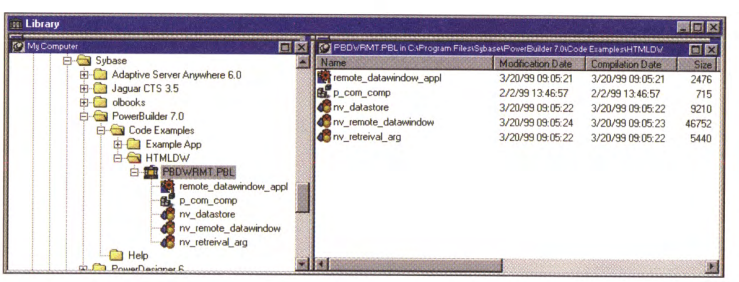


Figure 1

There are roughly 20 methods in this object—some which are critical for processing, some for public access, others protected for internal processing, and a few providing "nice-to-have" features. It is the intention of this article to present information on these methods, how they are called from PowerDynamo, and suggestions for adding or customizing methods.

Start by navigating from PowerBuilder to the Code Examples directory, specifically, the HTMLDW directory. Activate the remote datawindow application object (*remote_datawindow_appl*) to gain access to the objects, and open the object *nv_remote_datawindow*.

Here we discover some instance variables, including the *ids_datastore object*. At this point we have a link to our pasts as PowerBuilder developers: a datastore. It is this datastore to which we will assign a dataobject, retrieve data, and present that data via an HTML representation.

We could accomplish everything with a single method call; however, this does not make much sense. First, the method call would require a great number of arguments. Second, it severely limits the flexibility obtained by calling only the methods that we need. With that said, we realize that this instance variable must exist from one method call to the next. This is not a strange request for an instance variable in normal PowerBuilder. However, HTML development makes this request special. In the application server arena, variable persistency should always be considered. The use of stateful variables—that is, the ability to retain a single value from one method call to another—is the approach we need to take in utilizing the HTML datawindow component from Sybase.

"Borrow" From the Sybase Example

Developers learn very early not to re-invent the wheel. If it were not for cut-and-paste or save-as, how much code would there really be? As previously

Jay Hunt is a solution architect with Parametric Technology Corp. He can be reached at djayhunt@charter.net.

mentioned, Sybase has graciously provided an example for us to build upon. I have been using isugexample.pbl, so I copied *nv_datastore*, *nv_remote_datawindow*, and *nv_retrieval_arg* from HTMLDW (from Figure 1) to my project (Figure 2). Reactivate the application object and do a full rebuild.

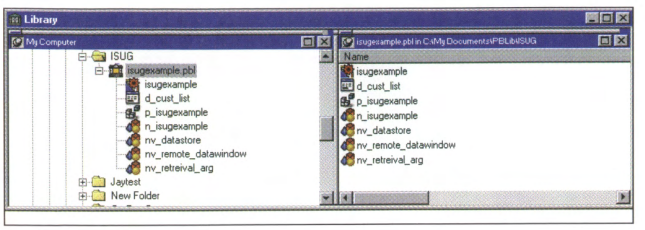


Figure 2

We will introduce two new methods to the remote datawindow. After opening the object *nv_remote_datawindow*, create a new function called "of_setdebug" with the following information (Figure 3).

```
Name:          of_setdebug
Access:        public
Return Type:   none
ARGUMENTS:
Pass By:       Type          Name
value          Boolean       ab_switch
Code:
ib_trace = ab_switch
return
```

Figure 3

The variable ib_trace has already been defined. This is used in the encapsulated processing to determine if log information should be produced to a text file. The text file is called srv.log and is normally stored in the bin directory of the Jaguar installation directory. The default value for this variable is "false," meaning log information is not produced.

Take a moment to review the other functions in this object at this time. Note the function "retrieve," which we will be using later, and notice the references to ib_trace. It is used in reference to three functions: *of_log_enter*, *of_log*, and *of_log_exit*. These functions are designed to isolate and present grouped information in an indented form. They will be very helpful in debugging activities, taking the place of ad hoc messageboxes.

Although the HTML datawindow has a method to handle the transaction object, we will add our own function. In this example, it is difficult to see the value of creating our own function, but in more complicated scenarios this approach will prove invaluable. Create another new function with the information displayed in Figure 4. In this example, we are going to take advantage of a connection cache. The information for the connection cache will be set up when we move to work in the Jaguar manager.

```
Name:          Settrans2
Access:        public
Return Type:   none

ARGUMENTS:
Pass By:       Type      Name

Code:
//--------------------------------------------------------//
//     Declare Needed Variables
//--------------------------------------------------------//
       Integer   li_rc
//--------------------------------------------------------//
//     Process Request
//--------------------------------------------------------//
       of_log ("Turning On the Base Transaction")
       sqlca.DBMS          = "ODBC"
       sqlca.DBParm        = "CacheName='EASDemo'"
       li_rc = ids_datastore.SetTrans(SQLCA)
       of_log ("SetTrans Statement for ids_datastore = " +
String(li_rc))
       if li_rc = 1 then ib_transactionSet = true
       of_continueWork (TRUE)
       of_log_exit("of_SetTrans2() = " + String(li_rc))
       return li_rc
```

Figure 4

From PowerBuilder to Jaguar

We have now finished working on the *nv_remote_datawindow* object and must ensure that this object is deployed to Jaguar. In my previous articles, I discussed the setup of the deployment project. We will use the same project in this article as well.

Open the project *p_isugexample* and add our new object to the project. This can be accomplished by selecting "Edit – Select Objects…" from the menu or by clicking on the "Select Objects" icon. Verify that the object *nv_remote_datawindow* is checked and close the dialog box (Figure 5).

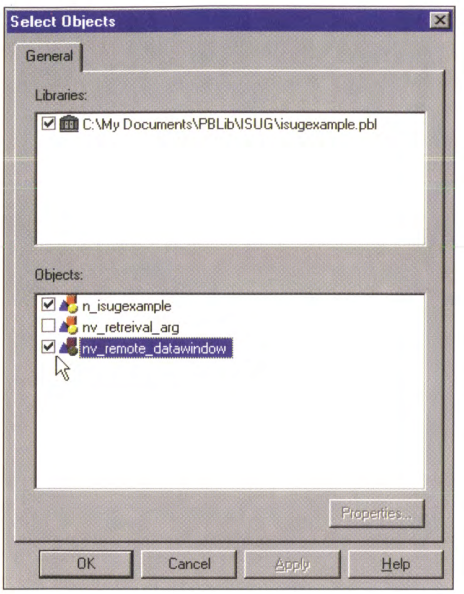


Figure 5

Open the Properties dialog box and select the Components tab. Verify that the information matches the display in Figure 6.

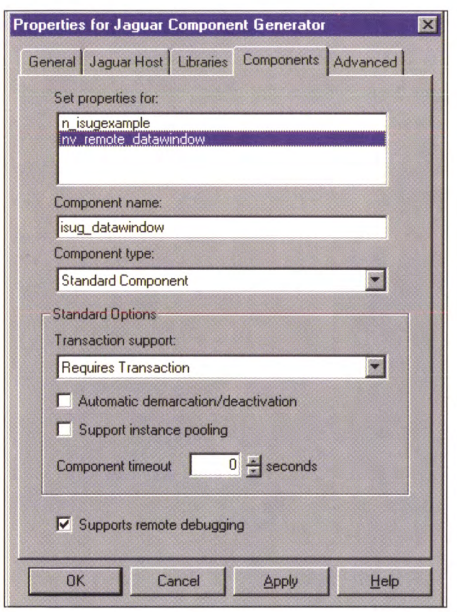


Figure 6

To keep things simple, the component name has been changed to "isug_datawindow," and this is the name we will eventually use to communicate with the object from PowerDynamo. This object does require a transaction, allowing the operation to be rolled back should a problem occur. In fact, this logic has been built into the HTML datawindow we have been working on. "Automatic demarcation/deactivation" is not checked nor is "Support instance pooling."

Build the project, open, and connect to the Jaguar manager. Navigate the treeview to expand the installed packages—our new object should be visible. Additional navigation should display the methods in this object (Figure 7).

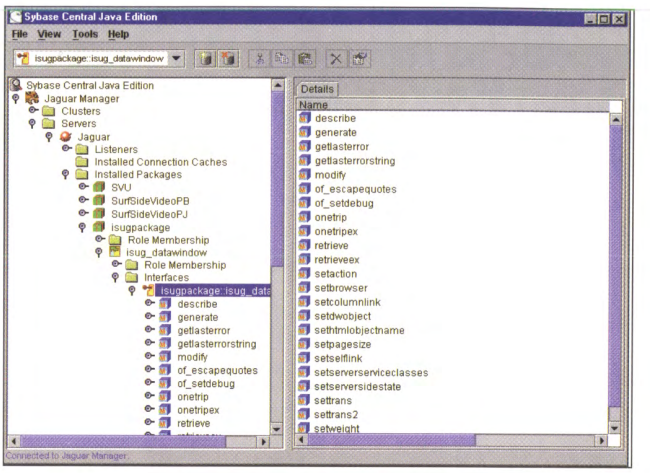


Figure 7

Because we have introduced a new function to the object, we must generate the Java stubs and skeletons, and compile them again. Select the isugpackage, right-click and select "Generate Stub/Skeleton..." (Figure 8).

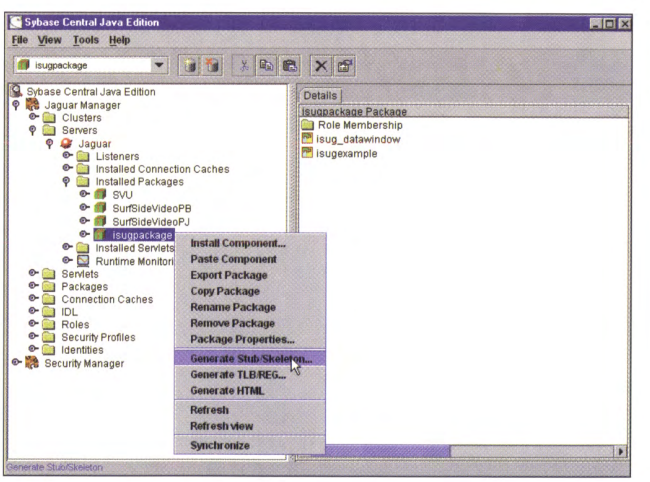


Figure 8

Select the "Generate Stubs" box, verify that the "Generate Java Stubs" box is also selected and push the "Generate" button. This information should be familiar to you now, as the process has been covered in previous articles. Exit to DOS and compile the stubs and skeletons. The normal location for

this information is "C:\Program Files\Sybase\Jaguar CTS 3.5\html\classes." Once in this directory, there should be a subdirectory with the same name as our object, "isugpackage." Change to this directory and issue the Java command javac *.java.

Here is also a piece of advice: Create a batch file to automatically complete this mundane task. I have included the batch file I use in Figure 9. The %1 is a placeholder for the command line parameter passed in at run time.

```
cls
cd \program files\Sybase\jaguar cts 3.5\html\classes
cd %1
cls
dir
```

Figure 9

The connection cache should be set up and verified at this time. From the Jaguar Manager, navigate the treeview to "Servers, Jaguar, Installed Connection Caches." In the Detail list on the right, look for EASDemo. If it is there, select it, right-click, and look at the Connection Cache Properties. Select the third tab, Cache, and verify that the Enabled cache-by-name access check box is checked.

If there is no entry for EASDemo, we will have to create it by right-clicking on Installed Connection Caches and selecting Install Connection Cache. This will display another dialog box. Click "Create and Install a New Connection Cache." Supply a description and enter "EAS Demo DB V3" in the Server Name field. This is the name of the System ODBC entry for the EAS Demo database that ships with PowerBuilder 7.0.

Supply the proper User Name and Password, activate the Driver tab, and enter ODBC32.DLL in the DLL or Class Name field while selecting ODBC. On the Cache tab, make sure both check boxes are checked and press OK to save the cache. Prior to pinging this cache to test it, you may have close, shut down, and restart Jaguar.

The HTML World

We now transfer our attention to PowerDynamo to set up the HTML portion of this example. Using the procedures discussed in my previous articles, create and open a new template. The one I will use in this example is called "isugdw_q1.htm." Add the code from Figure 10.

```
<HTML>
<TITLE>isugdw_Q1.htm</TITLE>
<BODY>
<H1>HTML DW example for Q1 Journal</H1>
<!--SCRIPT
  s_comp = "isugpackage/isug_datawindow";
  s_jag  = "iiop://localhost:9000";
  s_uid  = "jagadmin";
  s_pw   = "";

  s_serverbo = "";
  l_pagesize = 50;
  s_browser =
    document.GetServerVariable ("HTTP_USER_AGENT");

  s_slash  = "/";
  s_pblpath  = "C:" + s_slash;
  s_pblpath += "My Documents" + s_slash;
  s_pblpath += "PBLib" + s_slash;
  s_pblpath += "ISUG" + s_slash;

  s_pbl    = s_pblpath + "isugexample.pbl";
  s_dwo    = "d_cust_list";

  jagcomp = java.CreateComponent (s_comp,s_jag,s_uid,s_pw);

  if(jagcomp == null )
  {
    document.writeln ("<P>Unable to Load module.</P>");
  }
  else
  {
    jagcomp.of_setdebug(true);
    jagcomp.setdwobject(s_pbl, s_dwo);
    jagcomp.setserverserviceclasses ( s_serverbo );
    jagcomp.setpagesize(l_pagesize);
    jagcomp.setbrowser (s_browser);
    jagcomp.setweight (true, true, true, true, true) ;
    jagcomp.sethtmlobjectname ("dw_client");
    jagcomp.settrans2();
    retval = jagcomp.retrieve( );
    if (retval < 0)
    {
      document.writeln("Retrieve may have failed");
    }
    else
    {
      ls_html = jagcomp.generate();
      document.writeln( ls_html );
    }
  }

-->
</BODY>
</HTML
```

Figure 10

Notice some code similarities to my previous article? Last time we created a custom function called *of_setbrowser*, and this new code makes use of a similar function called *setbrowser*. Because our new HTML datawindow uses a datastore to get and display data, we need to associate a datawindow object to it. This is accomplished with the function *setdwobject*. In my previous article, this was hard-coded in the PowerBuilder object and did not provide any reusability. We also can control the debugging on a page-by-page basis by making the proper call to *of_setdebug*.

The code we are working on is intended to handle display processing—attention to business processing rules should be avoided. The processing of business rules is accomplished using a simple PowerBuilder NVO and associating it with the HTML datawindow object. This is done by the *setserverserviceclasses* function call. This example passes a string variable called *s_serverbo*, a server business object, to the HTML datawindow. There are no business rules in this case, so I have left this information blank.

The function *setpagesize* limits the number of rows the HTML datawindow will return for display. In this example we will return 50 rows. However, if needed, this number could be reduced by making a new assignment to the variable *l_pagesize*.

The function called *setweight* modifies certain HTML generation properties associated with the datawindow object being worked with. Using PowerBuilder, open *d_cust_list*, which we created last time. Investigate the properties, looking in particular at the HTML Generation tab (Figure 11).

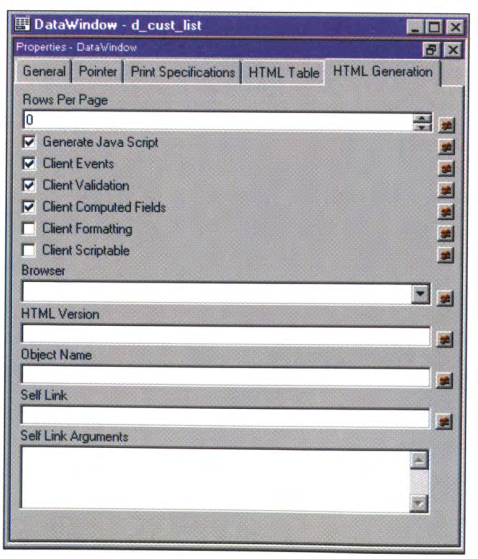


Figure 11

The functions we are calling from PowerDynamo adjust this information in run time, by calling the *setbrowser* function. Rows Per Page is adjusted by calling the function *setpagesize*. The *setweight* function controls the on/off status of the check boxes displayed in Figure 11.

In order to communicate with the HTML datawindow on the client side by making Java script calls, a name should be provided to the Object Name property displayed in Figure 11. This can be done via our HTML datawindow and the function call *sethtmlobjectname*.

The Result

Before the HTML datawindow is tested, verify that a mapping exists for the website and that the personal web server is running. Open the browser and navigate to the web page. The result is displayed in Figure 12.

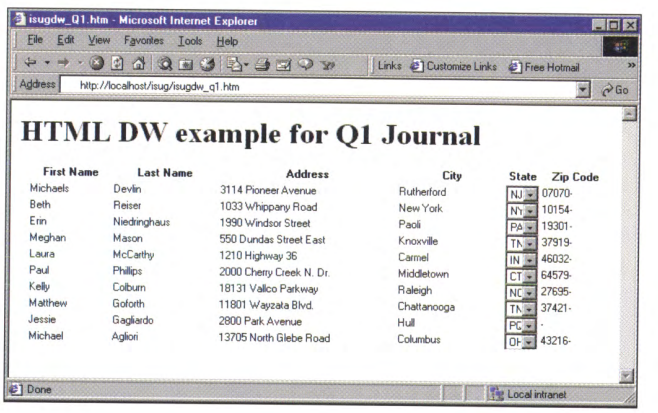


Figure 12

Although the result of this exercise is the same as in my previous article, the approach provides for greater flexibility and does not couple the processing to a particular business need. Further review of the HTML datawindow code in PowerBuilder will reveal several additional function calls to experiment with. Note that some functions are not public and cannot be called from PowerDynamo.

I should also address the issue of debugging since I mentioned the log file earlier. I have enclosed an excerpt from the log file produced when I ran the example (Figure 13 on the next page).

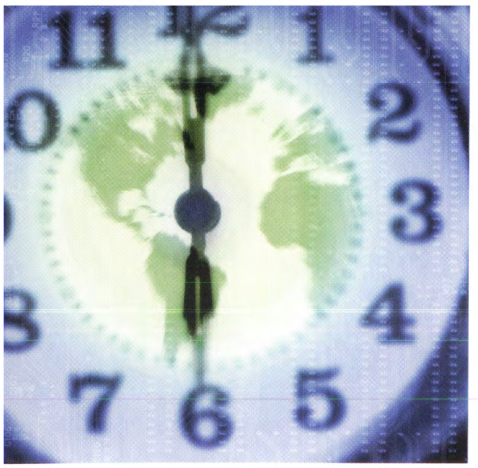
```
Dec 31 14:24:27 2000: HTMLDW: Entering SetDWObject('C:/My
Documents/PBLib/ISUG/isugexample.pbl','d_cust_list')
Dec 31 14:24:27 2000: HTMLDW:   Doing PBL/PBD
Dec 31 14:24:27 2000: HTMLDW:   SetLibraryList(C:\Program
Files\Sybase\Jaguar CTS 3.5\Repository\Component\isugpack-
age\isug_datawindow\C5\isugexample.pbd,C:/My
Documents/PBLib/ISUG/isugexample.pbl) = 1
Dec 31 14:24:28 2000: HTMLDW:   Entering
Modify('DataWindow.htmldw='yes'
DataWindow.NoUserPrompt='yes'')
Dec 31 14:24:28 2000: HTMLDW:   Exiting Modify() = ''
Dec 31 14:24:28 2000: HTMLDW:   EnableCommit()
Dec 31 14:24:28 2000: HTMLDW:   Entering
of_initializeFromDeclarative()
Dec 31 14:24:28 2000: HTMLDW:   Exiting
of_initializeFromDeclarative() = 1
Dec 31 14:24:28 2000: HTMLDW: Exiting SetDWObject() = 1

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Entering
SetServerServiceClasses('')
Dec 31 14:24:28 2000: HTMLDW: Exiting
SetServerServiceClasses() = 1

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Entering SetPageSize(10)
Dec 31 14:24:28 2000: HTMLDW: Exiting SetPageSize() = ''

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Entering
SetBrowser('Mozilla/4.0 (compatible; MSIE 5.0; Windows NT;
DigExt)')
Dec 31 14:24:28 2000: HTMLDW: Exiting SetBrowser() = ''

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Entering SetWeight()
Dec 31 14:24:28 2000: HTMLDW: Exiting SetWeight() = ''

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Entering
SetHTMLObjectName('dw_client')
Dec 31 14:24:28 2000: HTMLDW: Exiting SetHTMLObjectName() =
''

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Turning On the Base
Transaction
Dec 31 14:24:28 2000: HTMLDW: SetTrans Statement for
ids_datastore = 1

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Exiting of_SetTrans2() = 1

Dec 31 14:24:28 2000: HTMLDW: Entering Retrieve()
Dec 31 14:24:28 2000: HTMLDW: Exiting Retrieve() = 126

Dec 31 14:24:28 2000: HTMLDW: EnableCommit()

Dec 31 14:24:28 2000: HTMLDW: Entering Generate()
Dec 31 14:24:28 2000: HTMLDW: Exiting Generate(), length =
78722

Dec 31 14:24:28 2000: HTMLDW: SetComplete()

Dec 31 14:24:28 2000: HTMLDW: Entering Deactivate()
Dec 31 14:24:28 2000: HTMLDW: resetting state
Dec 31 14:24:28 2000: HTMLDW: Exiting Deactivate()
Dec 31 14:24:28 2000: HTMLDW: Destructor()
```

Figure 13

I edited the file to insert blank spaces to logically break up the code, but this can be accomplished by modifying the PowerBuilder code in *nv_remote_datawindow*. A quick review of this information allows us to see the results of the function calls we made. For example, we called *setbrowser* with a variable, but we can look at the log file and see the value of the variable. The same is true of the call to *setlibrarylist*. By looking for the "retrieve," we see that the datawindow has 126 rows.

Summary

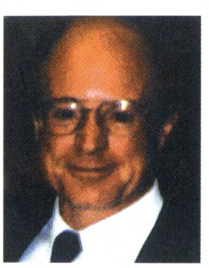
We have introduced the pre-written Sybase HTML datawindow, how to interface with it from PowerDynamo, and how to create custom functions. In addition, some debugging concepts were introduced. Too often, I have been told stories of organizations throwing away PowerBuilder code to develop an identical system using Java, just because the press says, "Java Java Java Java." By using the concepts relayed in this series of articles, PowerBuilder code and the datawindow can continue to be a benefit. ■



Time Translation and Quality Coding Methods in the Sybase Database

By Mark Gearhart

The second of a two-part series on translating time zones and displaying temporal data across Daylight Savings Time intervals



Mark Gearhart is an independent systems engineer specializing in product development for energy management systems. He can be reached at mark@mgearhart.com.

Historical energy accounting applications for electric utilities require the ability to manage data which is collected over time. In our applications, we use Sybase ASE to sample data for energy sales and purchases, transmission line voltages, circuit breaker statuses, power plant outputs, and power system apparatus on a periodic or event-triggered basis and store it as timestamped, quality-coded numeric values.

The first half of this article (Q4 issue, *ISUG Technical Journal*) described how we went about establishing the parameters of this historical database and setting up global conversion to local time zones. Having two distinct design issues to deal with—time and quality—presented an opportunity to develop the methods presented for both temporal and bit-wise database operations. Taken together, our solution provides the mechanisms needed to present data for any time zone, in a useful time and data format, and with a quality which describes the validity of the data. We successfully implemented these solutions on several multi-gigabyte, geographically dispersed energy accounting systems.

The second half of this article covers attaching quality codes to data, accessing data through views, selecting data into multiple columns, using quality codes to compute aggregates, and using UTC time for inserts and updates.

Attaching Quality Codes to Data

In our application, all data values contain a quality. The quality is a set of codes attached to each value intended to represent the characteristics of the sample. It is stored in the database as an *integer* data type and is initialized at the beginning of the hour to the quality of "U," meaning "meter inoperable or reading unavailable." The inserted row is later updated by metering or via manual entry as the hour progresses.

The quality is interpreted as a bit mask. For example, a quality of 10 is interpreted as 1010 [binary]. For display, this bit mask is converted to the characters that correspond to each bit. For the value 1010 [binary], bit 2 is given the meaning U, and bit 4 is given the meaning "M," or "value has been manually entered." The choice of codes is based on the type of application for which the database is intended. For a simplified set of codes noting the quality of data received hourly from remote locations, their settings might be configured as follows:

Quality	Meaning	Priority	Bit	Character
Manual	Value has been manually entered.	High	4	M
Questionable	Value is questionable (range error).		3	Q
NonUpdate	Meter inoperable or Reading unavailable.		2	U
Estimated	Value estimated from previous hour.	Low	1	E

Figure 8. *Quality code descriptions*

An integer datatype can hold up to 32 different codes. If the convention were to display all quality characters for a given bit mask, then we could end up using more real estate on the screen than can be afforded. Therefore, we also use a abbreviated form of the mask which shows just the highest priority quality. To alleviate the overhead of examining each bit at runtime, we pre-configure a *QTran* relation to contain all possible combinations of qualities:

Quality	QMask	QChar
0	""	""
1	E	E
2	U	U
3	UE	U
4	Q	Q
5	QE	Q
6	QU	Q
7	QUE	M
8	M	M
9	ME	M
10	MU	M
11	MUE	M
12	MQ	M
13	MQE	M
14	MQU	M
15	MQUE	M

Figure 9. *QTran relation*

Therefore, to access data quality, we join the *Quality* attribute in *HourData* to the *Quality* attribute in *QTran* in order to show both the full quality mask as well as the highest priority quality.

Accessing Data through Views

A view is created on behalf of the user in order to encapsulate the mechanics of both time translation and quality coding. This also provides the abstraction to add other types of translations as needed. For example, one extra translation which we have successfully implemented is the ability to view results in different conversion units; for example, *Kilowatts-days* instead of *Megawatt-hours*.

The view *V_HourData* returns the base attributes for the HourData, QTran, and TimeTran relations as well as the derived attributes for *LocTime*, *LocTimeEnd*, and *HourEnd*.

```
create view V_HourData as
select
 UTCTime   = hh.UTCTime,
 LocTime   = dateadd(mi,Offset,UTCTime),
 LocTimeEnd = convert(char(9),dateadd(ms,-
2,dateadd(mi,Offset,UTCTime)),1) +
        right("0"+convert(varchar(2),datepart(hh,dateadd(
        ms,-2,dateadd(mi,Offset,UTCTime)))+1),2) + "00",
 HourEnd   = right("0"+convert(varchar(2),datepart(hh,dateadd(
        ms,-2,dateadd(mi,Offset,UTCTime)))+1),2),
 DST       = TimeTran.DST,
 ZoneAbbr  = TimeTran.ZoneAbbr,
 VarNum    = hh.VarNum,
 Value     = hh.Value,
 Quality   = hh.Quality,
 QChar     = QTran.QChar,
 QMask     = QTran.QMask
from HourData hh,TimeTran,UserTimeZone,QTran
where UTCTime >= isnull(UTCStart,UTCTime)
and  UTCTime <= isnull(UTCStop,UTCTime)
and  TimeTran.Zone = UserTimeZone.Zone
and  ServerLogin = suser_name()
and  hh.Quality = QTran.Quality
```

Figure 10. *V_HourData view*

Note that LocTime returns the direct translation from UTC time in a native Sybase data type, whereas LocTimeEnd returns a time-ending format as a displayable character string, with the hour part shown as "00" through "24". Also, since this view returns both native local and UTC time, it would seem reasonable to allow selection against either time:

```
select HourEnd,DST,Value,QChar
from V_HourData
where UTCTime > "10/29/00 05:00"
and  UTCTime <= "10/30/00 06:00"
and  VarNum = 2953
order by UTCTime

select HourEnd,DST,Value,QChar
from V_HourData
where LocTime > "10/29/00 00:00"
and  LocTime <= "10/30/00 00:00"
and  VarNum = 2953
order by UTCTime
```

Figure 11. *Equivalent selections using UTC time or local time*

Both of these queries will in fact operate correctly. However, the performance degradation when using local time is significant. The reason for this is due to the fact that the Sybase query optimizer does not utilize the indexes defined for the base relation, when the derived attribute LocTime is used as a search argument. Instead, it performs a full table scan. If HourData contains just one hourly variable with a retention of one year, selection of only one row can mean the difference between three I/Os using UTC time versus 1,950 I/Os using local time.

We admit that it is unrealistic to expect users to know the UTC day-ending time for their time zone for any day of the year. Therefore, to facilitate the use of local time when formulating queries, a set of stored procedures is used to convert between local time and UTC time.

By using these stored procedures, one can take displayable local time such as 10/29/00 24:00:00, formatted as a *varchar* datatype, and call a procedure which returns the corresponding native UTC time of 10/30/00 06:00:00, formatted as a *datetime* datatype. This time can then be used as a search argument or in further date arithmetic.

Other procedures are devised for operations such as converting between UTC and local time; for finding the UTC beginning of a day; and for adding or subtracting one or more time units to a UTC time. UTC time arithmetic is unusual because, depending upon the day of the year, we must add (or subtract) 23, 24, or 25 hours to arrive at midnight of the next (or previous) day. Shown below are the most common stored procedures which, when used together, provide the means to perform time conversion.

The *sp_DspToLoc* procedure takes a fixed-format displayable time and DST indicator as input and returns the corresponding UTC time as output.

```
create procedure sp_DspToUTC @Loc varchar(33), @DST
as
declare @AdjLoc datetime
select @AdjLoc = case
 when substring(@Loc,10,2) = "24"
  then
 else convert(datetime,@Loc)
 End
exec sp_LocToUTC @AdjLoc,@DST,UTC output
return
```

Figure 12. *Stored procedure sp_DspToUTC*

The *sp_LocToUTC* procedure takes the local time and DST indicator as input and returns the corresponding UTC time as output. For the spring Daylight Savings Time interval, we consider the local 2 a.m. hour as being non-existent in the database. Therefore, if this hour is passed in, the UTC time is returned as NULL.

```
create procedure sp_LocToUTC @Loc datetime, @DST char(1),
@UTC datetime output
as
select @UTC = dateadd(mi,-Offset,@Loc)
from TimeTran, UserTimeZone
where UTCStart < dateadd(mi,-Offset,@Loc)
and UTCStop >= dateadd(mi,-Offset,@Loc)
and DST = @DST
and TimeTran.Zone = UserTimeZone.Zone
and UserTimeZone.ServerLogin = suser_name()
return
```

Figure 13. *Stored procedure sp_LocToUTC*

The *sp_UTCToLoc* stored procedure is the reverse of *sp_LocToUTC*. It takes UTC time as input and returns the corresponding local time and DST indicator as output.

```
create procedure sp_UTCToLoc @UTC datetime, @Loc datetime
output,
 @DST char(1) output
as
select @Loc = dateadd(mi,Offset,@UTC), @DST = DST
from TimeTran, UserTimeZone
where @UTC between UTCStart and UTCStop
and TimeTran.Zone = UserTimeZone.Zone
and UserTimeZone.ServerLogin = suser_name()
return
```

Figure 14. *Stored procedure sp_UTCToLoc*

The *sp_UTCAdd* procedure performs UTC time arithmetic by taking a UTC time, time unit, and increment as input and returning the new UTC time as output. For adding hours and minutes, we apply the dateadd() function directly against the input UTC time. For days, weeks, months, and years, we first convert to local time, apply the *dateadd()* function, and convert back to UTC time, thus accounting for the 23, 24, and 25 hour days in our result. For example, if we add one day to 10/29/00 05:00 (i.e., midnight Central time) the result is 10/30/00 06:00, which is 25 hours, not 24, in the future.

```
create procedure sp_UTCAdd @OldUTC datetime, @Unit char(2),
@Inc integer,
 @NewUTC datetime output
as
declare @Loc datetime, @DST char(1)
if @Unit = "mi"
 select @NewUTC = dateadd(mi,@Inc,@OldUTC)
else if @Unit = "hh"
 select @NewUTC = dateadd(hh,@Inc,@OldUTC)
else
begin
 exec sp_UTCToLoc @OldUTC,@Loc out, @DST out
 select @Loc  = case
  when @Unit = "dd" then dateadd(dd,@Inc,@Loc)
  when @Unit = "wk" then dateadd(wk,@Inc,@Loc)
  when @Unit = "mm" then dateadd(mm,@Inc,@Loc)
  when @Unit = "yy" then dateadd(yy,@Inc,@Loc)
  end
 exec sp_LocToUTC @Loc, @DST, @NewUTC out
end
return
```

Figure 15. *Stored procedure sp_UTCAdd*

Having constructed both the procedures and the view necessary to translate data from HourData, a production query can now be constructed:

```
declare @UTCStart datetime, @UTCStop datetime
exec sp_DspToUTC "10/29/00 24:00:00", "", @UTCStop out
exec sp_UTCAdd dd,-1,@UTCStop, @UTCStart out
select HourEnd,DST,ZoneAbbr,Value,QMask
from V_HourData
where UTCTime > @UTCStart
and  UTCTime <= @UTCStop
and  VarNum = 2953
order by UTCTime
```

Figure 16. *Query used to select data from V_HourData*

Comparing this query to our initial query in Figure 2 from the previous article, we note that selection of hourly data for "Plant Inverness, Unit 1" can now be executed correctly, with the results shown as follows:

HourEnd		DST	ZoneAbbr	Value	QMask
01		CDT	520		
02		CDT	411	ME	
02	2	CST	409		
03		CST	224		
04		CST	217	Q	
05		CST	274	Q	
06		CST	205	Q	
07		CST	596	Q	
08		CST	677		
09		CST	507		
10		CST	569		
11		CST	542		
12		CST	541		
13		CST	527		
14		CST	624		
15		CST	766		
16		CST	714		
17		CST	683		
18		CST	664		
19		CST	550		
20		CST	549	M	
21		CST	750	MQ	
22		CST	643		
23		CST	444		
24		CST	350		

Figure 17. *Results of the V_HourData selection*

The revised query, by way of the supporting stored procedures, accepts a local time reference as the originating input. The number of rows returned will vary according to a 23, 24, or 25 hour day, and the second Daylight Savings Time hour will return the denoting indicator. The user will also see the correct hour-ending reference.

Selecting Data into Multiple Columns

Now that we have succeeded in selecting data for a single variable, we can extend the implementation to select multiple variables simultaneously. *Table pivoting* can be used to translate a long narrow relation into a short wide one. The need for such formatting occurs frequently in practice, primarily because, while the narrow representation for HourData is better for data manipulation, the wide form is better for presenting data by common subject.

For example, to show all four generating units for Plant Inverness Generation, the query needs to return data for four variables. If we conform to the format of HourData, we would

have 4*24 rows of results, and we would need to identify each row by the variable number. In a pivoted format, the data and quality for each variable are presented in adjacent columns, along with the local hour indicator. In order to obtain this format, a point characteristic function is used so that the presentation attributes for value and quality are pivoted by the base attribute, UTCTime. The query needed to produce this result is given as in Figure 18:

```
select LocTimeEnd,DST,ZoneAbbr,
 Unit1Value = sum(Value*(1-abs(sign(VarNum-2953)))),
 Unit1Qual = max(substring(QMask,1,4*(1-abs(sign(VarNum-
2953))))),
 Unit2Value = sum(Value*(1-abs(sign(VarNum-2954)))),
 Unit2Qual = max(substring(QMask,1,4*(1-abs(sign(VarNum-
2954))))),
 Unit3Value = sum(Value*(1-abs(sign(VarNum-2955)))),
 Unit3Qual = max(substring(QMask,1,4*(1-abs(sign(VarNum-
2955))))),
 Unit4Value = sum(Value*(1-abs(sign(VarNum-2956)))),
 Unit4Qual = max(substring(QMask,1,4*(1-abs(sign(VarNum-
2956))))),
from V_HourData
where UTCTime > "10/29/00 05:00"
and UTCTime <= "10/30/00 06:00"
and VarNum in (2953,2954,2955,2956)
group by UTCTime,LocTimeEnd,DST,ZoneAbbr
order by UTCTime
```

Figure 18. *Selection of hourly data via table pivoting techniques*

Results are returned as in Figure 19 (below).

Using Quality Codes to Compute Aggregates

When computing aggregates such as max, min, sum, and average, the result must contain a quality which is propagated from the input. In addition, it is necessary to exclude from the result any hours which have failed to receive data. This condition for missing data is denoted by the quality code U (non-update) associated with the hourly value. Therefore, our

LocTimeEnd	DST	Zone	Abbr	Unit1Value	Unit1 Qual	Unit2 Value	Unit2Qual	Unit3Value	Unit3 Qual	Unit4 Value	Unit4 Qual
10/29/00 0100		CDT	520		559	322	E	805			
10/29/00 0200		CDT	411	ME	467	322	E	775			
10/29/00 0200	2	CST	409		460	322		770			
10/29/00 0300		CST	224	Q	454	322	E	773			
10/29/00 0400		CST	217	Q	420	346		772			
10/29/00 0500		CST	274	Q	464	367		815			
10/29/00 0600		CST	205	Q	645	368		842			
10/29/00 0700		CST	596		677	500		843			
10/29/00 0800		CST	677		684	655		0	U		
10/29/00 0900		CST	507		774	690		877			
10/29/00 1000		CST	569		816	817		882			
10/29/00 1100		CST	542		706	850		871			
10/29/00 1200		CST	541		700	842		881			
10/29/00 1300		CST	527		732	M	850		850		
10/29/00 1400		CST	624		801	850		850			
10/29/00 1500		CST	766		804	850	MQ	872			
10/29/00 1600		CST	714		769	837	ME	869			
10/29/00 1700		CST	683		763	M	836	E	870		
10/29/00 1800		CST	664		744	835		876			
10/29/00 1900		CST	550		712	835		877			
10/29/00 2000		CST	549	M	502	833		875	E		
10/29/00 2100		CST	750	MQ	827	850		873	E		
10/29/00 2200		CST	643		698	830		874			
10/29/00 2300		CST	444		500	542		850			
10/29/00 2400		CST	350		502	317		850			

Figure 19. *Hourly results of the table pivot query*

aggregate calculations require the ability to support quality attributes as quantifiers. In practice, we avoid the use of the standard Sybase aggregate functions and instead create our own. For example, to compute the daily summation and average for hour data over a day, a view is written as follows:

```
create view V_DayAggregate as
select
 UTCTime = max(UTCTime),
 VarNum  = VarNum,
 ValSum  = sum(Value*abs(sign(Quality&2)-1)),
 ValAvg  = case
  when sum(abs(sign(Quality&2)-1)) > 0 then
   sum(Value*abs(sign(Quality&2)-1)) /
sum(abs(sign(Quality&2)-1))
  else 0
 end,
 Quality = (sign(sum(Quality&7))*4)|(sign(sum(Quality&8))*8)
from HourData
group by VarNum
```

Figure 20. *Calculation of summation and average values from hourly inputs*

The UTCTime is returned as the maximum hour over the interval. In the case where the query spans one day, the maximum time will return midnight at the end of the day. This date is consistent with the time-ending format of our database model. The quality of each input value is used to control its participation in the result as well as the propagation of its input quality to the result quality. In this query, any input value whose quality is U (non-update), is omitted from the sum and average. These values are omitted via the expression **abs(sign(Quality&2)−1**, which returns 0 if the non-update quality U is set, or 1 otherwise. Thus, the multiplier for Value is either 1 or 0 based on the presence of U in the quality.

The quality is set to "Q" (questionable) if any of the inputs have qualities U, Q, or E (estimated). The quality also set to M (manually entered) if any of the inputs are manually entered. For efficiency, the expression which evaluates U or Q or E (i.e. **Quality&4|Quality&2|Quality&1**) is combined into the equivalent expression **Quality&7**. A homemade average rather than the normal avg() function is constructed in order to exclude non-updated values from the result. The case statement tests the divisor to prevent divide-by-zero.

Other aggregates are possible based on the application. For example, we could include a special maximum and minimum aggregate, and we could even extend QTran to include a new "I" quality to indicate *identical* max/min values over the interval.

The daily aggregate view *V_DayAggregate* in Figure 20 is insufficient for production use since it does not return local time or any displayable quality codes. Therefore, we create a higher-level view called *V_DayData* which includes the extra derived attributes.

```
create view V_DayData as
select
 UTCTime   = dd.UTCTime,
 LocTime   = dateadd(mi,Offset,UTCTime),
 LocTimeEnd = convert(char(9),dateadd(ms,-
2,dateadd(mi,Offset,dd.UTCTime)),1) +
        right("0"+convert(varchar(2),datepart(hh,dateadd(
        ms,-2,dateadd(mi,Offset,dd.UTCTime)))+1),2) + "00",
 ZoneAbbr  = TimeTran.ZoneAbbr,
 VarNum    = dd.VarNum,
 ValSum    = dd.ValSum,
 ValAvg    = dd.ValAvg,
 Quality   = dd.Quality,
 QChar     = QTran.QChar,
 QMask     = QTran.QMask
from V_DayAggregate dd,TimeTran,UserTimeZone,QTran
where UTCTime >= isnull(UTCStart,UTCTime)
and  UTCTime <= isnull(UTCStop,UTCTime)
and  TimeTran.Zone = UserTimeZone.Zone
and  ServerLogin = suser_name()
and  dd.Quality = QTran.Quality
```

Figure 21. *V_DayData view for selection of aggregate data*

We can use this view to select daily aggregates for the UTC day-ending 10/30/00 06:00, for the variables 2953, 2954, 2955, and 2956.

```
declare @UTCStop datetime

exec sp_DspToUTC "10/29/00 24:00:00", " ", @UTCStop out

select UTCTime,LocTime,LocTimeEnd,ZoneAbbr,VarNum,
 ValSum,ValAvg,Quality,QChar,QMask
from V_DayData
where UTCTime = @UTCStop
and  VarNum in (2953,2954,2955,2956)
order by VarNum
```

Figure 22. *Query used to select data from V_DayData*

Results are returned as follows.

```
UTCTime
LocTime
LocTimeEnd     Zone
Abbr  Var
Num   Val
Sum   Val
Avg   Qual
ity
QChar
QMask
10/30/00 06:00          10/30/00 00:00    10/29/00 2400
CST   2953     12956    518      12       M        MQ
10/30/00 06:00          10/30/00 00:00    10/29/00 2400
CST   2954     16180    647      8        M        M
10/30/00 06:00          10/30/00 00:00    10/29/00 2400
CST   2955     15988    639      12       M        MQ
10/30/00 06:00          10/30/00 00:00    10/29/00 2400
CST   2956     20292    845      4        Q        Q
```

Figure 23. *Results of the V_DayData selection*

Alternately, the results could also lend themselves to presentation in a row-wise fashion. That is, a single attribute Value could be shown along with a *ValueType* denoting whether the quantity is a sum or average. Support for this schema would prompt the use of *table folding techniques*. As the number of different types of aggregates grow, this format might be useful. However, since we have only two aggregates, we have chosen the column-wise format. Just as with hourly data, daily aggregates are candidates for table pivoting. With the same *point characteristic function*, two queries can be written for the presentation of daily sum and average:

```
select LocTimeEnd,ZoneAbbr,
        Unit1Value = sum(ValSum*(1-abs(sign(VarNum-2953)))),
        Unit1Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2953))))),
        Unit2Value = sum(ValSum*(1-abs(sign(VarNum-2954)))),
        Unit2Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2954))))),
        Unit3Value = sum(ValSum*(1-abs(sign(VarNum-2955)))),
        Unit3Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2955))))),
        Unit4Value = sum(ValSum*(1-abs(sign(VarNum-2956)))),
        Unit4Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2956)))))
from V_DayData
where UTCTime = "10/30/00 06:00"
and VarNum in (2953,2954,2955,2956)
group by LocTimeEnd,ZoneAbbr
select LocTimeEnd,ZoneAbbr,
        Unit1Value = sum(ValAvg*(1-abs(sign(VarNum-2953)))),
        Unit1Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2953))))),
        Unit2Value = sum(ValAvg*(1-abs(sign(VarNum-2954)))),
        Unit2Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2954))))),
        Unit3Value = sum(ValAvg*(1-abs(sign(VarNum-2955)))),
        Unit3Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2955))))),
        Unit4Value = sum(ValAvg*(1-abs(sign(VarNum-2956)))),
        Unit4Qual  = max(substring(QMask,1,4*(1-
abs(sign(VarNum-2956)))))
from V_DayData
where UTCTime = "10/30/00 06:00"
and VarNum in (2953,2954,2955,2956)
group by LocTimeEnd,ZoneAbbr
```

Figure 24. *Selection of daily aggregates via table pivoting techniques*

The results these two queries are identical to the desired results for our web page in Figure 1.

```
LocTimeEnd     Zone
Abbr  Unit1
Value Unit1
Qual  Unit2
Value Unit2
Qual  Unit3
Value Unit3
Qual  Unit4
Value Unit4
Qual
10/29/00 2400 CST     12956   MQ      16180   M
15988         MQ      20292   Q
10/29/00 2400 CST     518     MQ      647     M
639   MQ      845     Q
```

Figure 25. *Daily results of the table pivot query*

Using UTC Time for Inserts and Updates

One final issue deals with the way in which we originally inserted data and subsequently updated changes to values and qualities. In order to insert or update rows in HourData, the timestamp for each value must be stated as a UTC time. Within a data collection program, we use *transaction time* and *effective time* as two orthogonal time dimensions.

Transaction time is the time when the data is stored in the database. It is consistent with the serialization order of transactions (i.e., it is monotonically increasing) and can be implemented using the *getdate()* function.

Effective time denotes the time when data becomes effective in reality (i.e. data received during the hour is effective hour-ending). If the *getdate()* function were to return a local time, it would not be not possible to convert to UTC time for either the transaction time or effective time without knowing from *getdate()* whether we were in the first or second Daylight Savings Time hour.

Unfortunately, *getdate()* does not return this information, so we have configured the ASE server to operate in UTC time, thus avoiding the problem entirely. This configuration can be done by setting the TZ (time zone) environment variable in the ASE server startup script:

```
#! /bin/sh
DSLISTEN=$DSQUERY; export DSLISTEN
TZ=0; export TZ
$SYBASE/bin/dataserver –d/dev/rMDBlv1 –e$SYBASE/DS.log
2> &1 &
```

Figure 26. *Sybase ASE startup script*

Following the receipt of data, and assuming the Value, VarNum, and Quality are passed in the *@Value*, *@VarNum*, and *@Quality* parameters, the UTC timestamp is computed to the nearest hour-ending as follows:

```
update HourData set Value = @Value, Quality = @Quality
where VarNum = @VarNum
and UTCTime = convert(datetime,
  convert(char(9),dateadd(hh,1,dateadd(ms,-2,getdate())),1) +
  convert(char(2),dateadd(hh,1,dateadd(ms,-2,getdate())),8) + ":00")
```

Figure 27. *Construction of the update statement for hour-ending timestamps*

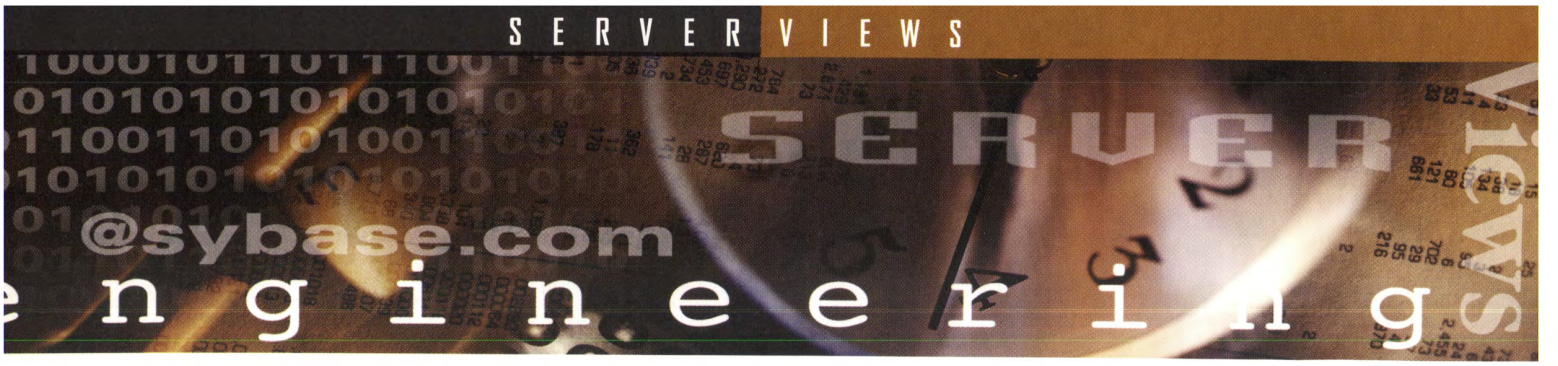
In the event that the update occurs at the exact top of the hour (i.e., the 0th millisecond of the 0th second of the *next* hour), we subtract two milliseconds to bring it into the current hour before taking the *mm/dd/yy hh* parts and appending 0 minutes. This logic produces a timestamp such that receipt of data for any time during the hour will be placed into the database to the nearest hour-ending time.

Conclusions

Both the UTC time translation and quality coding methods described in this paper have proven effective in satisfying the requirements imposed by our application. Even though we have implemented these methods against a continuous time database model, they can also be adapted to work for an interval database model, where each unit of time would be represented by two attributes, *start time* and *end time*.

Initially, we have applied the design to a collection of sites whose data and users are in a single time zone. However, since data for our application will eventually be replicated from one time zone to another, the reconciliation of timestamps will become far more complex than for a single system. For example, if a midnight value is generated in the USA Eastern time zone with a timestamp of 11/01/00 05:00:00, and this value is then replicated to a site in the USA Pacific time zone, what timestamp should be chosen for the replicate copy? Several options are possible. The timestamp can be changed to 11/01/00 08:00, placing it at midnight USA Pacific. Users in this time zone can then access the value as if it were a local midnight value. Another option might be to preserve the USA Eastern timestamp and install a user at the Pacific site with a USA Eastern time zone.

Furthermore, scenarios in which multiple sites replicate to a central site may present the opportunity for multi-zone queries such as "find the midnight value for data at different locations around the world," or the opposite case: "find all values at this particular local time." These issues continue to pose challenges as larger and more geographically-disperse systems go online. But for now, we have addressed the immediate goals of solving both the Daylight Savings Time and quality coding problems, while providing a design foundation for the future expansion of the system. ■



Lessons Learned from Performance Benchmarking

By Ian Smart

In the ASE SIG at TechWave last summer, there was a discussion about how much performance degradation could be encountered when a user switched on the different types of monitoring, using SQL Monitor and *sp_sysmon*. Being the headstrong type, I seem to remember making a statement like, "Well, I'll go away and test it, and then I'll write it up for the ISUG journal." At the time I was thinking this would only take two days, so I assumed that this would not be a problem. How wrong I was.

This is the promised article and it does contain an answer to the question that was raised. However, it also will document some of the issues that arose whilst trying to get these numbers since, in many ways, they give a very different answer to this question.

The benchmark was run on a medium-size Sun machine that happened to be available. The specification of this machine is as follows:

Host	*Sun E6500*
Memory	9GB total memory on 6 boards
CPU	18 CPUs on 9 boards
CPU Specification	336 Mhz UltraSPARC-II with 4Mb of E-Cache per CPU
Disks	Sun 9.0G disks in a SPARCstorage Array + Sun Fast SCSI disk controller
Network	100 Mbps full-duplex Link Up
Software	Sybase ASE 12.0 Sybase TPC Type B internal test software

Since my intention was just a quick test, I decided to use a TPC type B benchmark, as this is extremely well known and a good simple test of OLTP performance. The other advantage is that, having run this test many times in the past, I already have configuration files for ASE so the tuning would be kept to a minimum. All of this was going to help me run a quick test! The final advantage of this set-up was that I already knew what numbers to expect in terms of the throughput, so if anything was set up incorrectly, this would easily show up. As it transpired, this was extremely useful.

Disk I/O

The host that was being used for the test had no raw devices available and so, in the interest of time, it was decided to use files on one of the file systems, and to have the dsync set on for these devices. The dsync option was added in ASE 12.0 and is defined in the

Sybase ASE 12.0 Systems Administration Guide as follows:

"For devices initialized on UNIX operating system files, the dsync setting controls whether or not writes to those files are buffered. When the dsync setting is on, Adaptive Server opens a database device file using the UNIX dsync flag. The dsync flag ensures that writes to the device file occur directly on the physical storage media, and Adaptive Server can recover data on the device in the event of a system failure.

When dsync is off, writes to the device file may be buffered by the UNIX file system, and the recovery of data on the device cannot be ensured. The dsync setting should be turned off only when data integrity is not required, or when the System Administrator requires performance and behavior similar to earlier Adaptive Server versions.

Note: The dsync setting is ignored for devices initialized on raw partitions, and for devices initialized on Windows NT files. In both cases, writes to the database device take place directly to the physical media."

The Initial Test Results

The results of the initial testing were extremely poor against the expected results. My immediate response was to reach for the *sp_sysmon* output, look at the query plans, and switch on SE, a commonly used Solaris monitoring package. All of the numbers suggested that what was being seen was a TPC type B benchmark that was performing exactly as would be expected, just very slowly. Also of concern was that when pretty major ASE configuration changes were implemented, these were having little, if any, effect.

In a state of confusion, I contacted a colleague of mine, Dhimant Chokshi, whose opinion I value greatly. After reviewing everything, we were pretty much in agreement, until he suddenly made the comment, "Hey, are you using dsync?" To cut a long story short, whenever we perform an I/O to a file system using the OS provided dsync option, while it ensures that the I/O is guaranteed and the database can be trusted, it also has a performance implication associated with it. By switching off the dsync option for the devices that we were using, the throughput rose back to a level that was closer to what was expected.

This made me reflect on a conversation at the Sybase UK User Group a couple of weeks earlier, where a number of people were reporting seeing poor performance after upgrading to ASE 12.0, even though upon investigation everything looked exactly as it should. It made me wonder how many of these cases were as a result of encountering dsync behavior?

While this would not apply to most of those devices which have been placed on file systems, as these have dsync switched off after an upgrade, it would apply to any disk activity that was occurring on a master device, where that had been placed on a file system (in those UNIX environments that support dsync), as the dsync on the master device is on by default. While it should be stressed that from the common sense perspective, this is an extremely undesirable way of installing ASE since, pre-ASE 12.0, this would mean that writes to the master database are not guaranteed, it did make me wonder in how many cases this was what was being encountered.

The UNIX Kernel

The benchmark was now nearly a week behind schedule and, because we were now using buffered writes, they were quite compromised. However, we also had another problem. The results were fluctuating wildly (almost 100%). Once again I reached for my trusty *sp_sysmon* and again the output gave no indication as to why this behavior was being seen. As far as ASE was concerned, the world was simply speeding up and slowing down. Once again, I analyzed the system using SE, but this time it showed me a major problem.

The machine that I was using had been configured expressly to run a particular application server from one of our partners. This had required some specific kernel changes. The way that the flushing had been configured meant that there was both a high overhead due to the amount of file system cache checking that was occurring, and that the hit that was taken by flushing the cache was also extremely high. Depending upon the number of cache flushes and requirements for cache flushing that were encountered, so varied the throughput. By reconfiguring the *tune_t_fsflushr* and *autoup* in the Solaris kernel, coupled with removing all of the changes that had been made to the UNIX message queues, another 30% + was added to the throughput and the results became more uniform. However, they were still too inconsistent to be used.

The Final Solution

Since the results that were obtained were not consistent enough, it was decided that the only option left was to remove some of the file systems and to use the underlying raw devices directly. This would remove any issues associated with file system caching and allow us to get the correct answer to the question that we were trying to answer. This change was made and eventually the proper tests were started. However, the problem now was that I was already very behind schedule and what should have been a two-day benchmark had now taken over two weeks! I therefore decided to try and cut

corners by running multiple tests after each other and not always bouncing the ASE server prior to the test, thinking that, "Hey, that cache should be warm enough." Once again, inconsistency abounded and the results had to be thrown away. Once again, we became further behind schedule. Finally, the full duration tests were set up to run in a consistent, properly set-up environment and the results that were required were obtained. These results are documented in the final section of this document.

Salient Lessons We Learned

There are a number of points that came out of this testing that are fundamental and basic, but which I felt needed to be laid out, as they are important:

- **Always look at the big picture.** So often at user group events we have discussions about the 5% and 10% performance hits associated with different ASE parameters. The reality is that concentrating on these first can result in missing the real major problem.
- **Don't cut corners.** By trying to do this benchmark quickly, it ended up taking almost three weeks and requiring a test run that finished late on Christmas Eve. By spending the time early on to set this up correctly, this would have been avoided.
- **The problem may not be in ASE.** Very often the problems that we encounter are either caused by external or hardware issues. I remember a colleague of mine who spent quite some time trying to work out why a particular query was going slowly when an application had been migrated. It transpired that the wrong type of cable had been used to connect the disk drive that the table being accessed was sitting on, and it was only by chance when he walked behind the machine that he spotted this, two days later. No amount of tuning in ASE would solve this problem.

  I remember another case where we increased the amount of work that we were doing in ASE quite significantly, and yet ASE was telling us that it was still using 50% CPU. It was only when we observed that the engines were being scheduled twice as often as they had been previously that the true picture emerged.

On specific ASE issues:

- Realize that dsync is not exactly the same as async I/O against a raw device. Whilst the net effect is the same, the reality is different and has different performance characteristics.
- When tuning in a scenario where files in file systems have been used as disk devices, realize that tuning for short term bursts of I/O is different from tuning for concerted long term I/O against the device. If your writes are being cached, how are these being de-staged?
- Realize the ASE is executed within an operating system just like everything else, and is as susceptible to a bad kernel configuration as the next process—in fact maybe more so.

And finally always remember the following reality check:

- Never be so arrogant and so stupid that you think you know so much that you can quickly do a benchmark. There is always something you haven't expected.
- Don't always reach for the *sp_sysmon* as your first step in tuning an application. It only knows what is happening from the perspective of what is happening inside ASE. It knows virtually nothing of the outside world. It should therefore be used in conjunction with OS monitoring tools, not in isolation.

The Benchmarking Results

The original purpose of this benchmark was to determine the performance impact of using SQL Monitor and *sp_sysmon*. The reason for this is two-fold. Firstly, there was a concern about the contention between the processes that are updating the monitor counters and the process that is reading them. Secondly, the way that *sp_sysmon* generates its output uses a large amount of temporary table activity that could also have a performance impact. Therefore, as you would imagine, the impact that is observed will vary greatly depending upon the type of load that is being applied. The figures that are described here are from one very specific example, and should not be taken as absolutes to be applied in every case.

The tests themselves consisted of running with 50, 100, and 150 users against an ASE that had two, four, or eight engines. The configuration of the benchmark was such that 150 users would utilize approximately 75% of an eight-engine ASE. Since the *runnable process search count* was 0, the ASE CPU used was close enough to being an absolute as was necessary.

As many of you will be aware, TPC type B benchmarks are mainly I/O-bound on either the transaction log or the smallest table (branch). Therefore, by reducing the number of engines from eight to four to two, this moved the test from being I/O-bound to being very CPU-bound and was intended to see whether the impact varied more in any particular scenario.

However, when the initial eight-engine to four-engine tests were performed, there was very little impact in throughput observed, even though we expected it to move from I/O-bound to being lightly CPU-bound. The reason for this is in the way that the CPU use is calculated. This is done through a counter-based mechanism, where, when the timing interrupt requested from the operating system is detected by the ASE engine, it records what it is doing at that point in time.

Therefore, there may be some "holes" in the timeslice, which actually mean that there is more headroom than may have been thought. Also, since much of the time of a transaction in this case is taken up by the commit of the transaction, then spare capacity within that group commit will also provide some additional capacity.

The results obviously varied depending upon the number of users and the number of engines that were used, and there isn't space in this article to write them up fully. However, the following are the main points that were observed, to answer the question from TechWave:

In the 150-user tests:

- The impact on throughput from using the Monitor Server varied between 4.3% when there were eight engines and the benchmark was I/O-bound, to 52.84% when there were only two engines and a great deal of CPU contention.
- The impact on throughput from using *sp_sysmon* was always under 4.32% in all cases where eight engines were being used.
- The worst case impact on throughput from *sp_sysmon* was in the case when it was being executed at 30-second intervals on the heavily CPU-bound two-engine test, and this caused a 29.72% performance degrade.

In the 100-user tests:

- The impact on throughput from using the Monitor only rose to 29.79% when there were only two engines and there was a great deal of CPU contention.
- The impact on throughput from using *sp_sysmon* was higher than the 150-user impact (which is unexplained) but was always under 11.68% in all cases where eight engines were being used.
- The worst case impact on throughput from *sp_sysmon* was in the case when it was being executed at 30-second intervals on the lightly CPU-bound four-engine test, and this caused a 18.57% performance degrade. Interestingly, the two-CPU test did not degrade so badly.

In the 50-user tests:

- The impact on throughput from using the Monitor only rose to 21.91% when there were only two engines and there was a great deal of CPU contention.
- The impact on throughput from using *sp_sysmon* was always under 3.46% in all cases where eight engines were being used.
- The worst case impact on throughput from *sp_sysmon* was in the case when it was being executed at two-minute intervals on the heavily CPU-bound two-engine test, and this caused a 23.05% performance degrade.

Summary

In conclusion, I would make the following points:

- The impact on throughput in all cases where the ASE is not heavily CPU-bound was a great deal less than the numbers that were being discussed in the session in Orlando.
- While on a lightly loaded ASE, the impact caused by sp_sysmon and the SQL Monitor are fairly similar, in cases of extreme CPU contention, sp_sysmon does cause less of a throughput degradation.

Finally, by the time you read this article, ASE 12.5 will be available with its new performance diagnostic tool. This has been designed to specifically address these contention issues while providing the user with a great deal of useful and relevant information. Check it out!! ■

Ian Smart is Senior Technical Evangelist for Sybase's Enterprise Systems Group. He has addressed U.S. and European ISUG audiences on many occasions, as well as speaking at numerous user group and masterclass events in the UK. If you have a question you would like answered or a comment about an article, Ian can be reached at ian.smart@sybase.com.

Brick Walls Do Not a Foundation Make

By Thomas Lamb

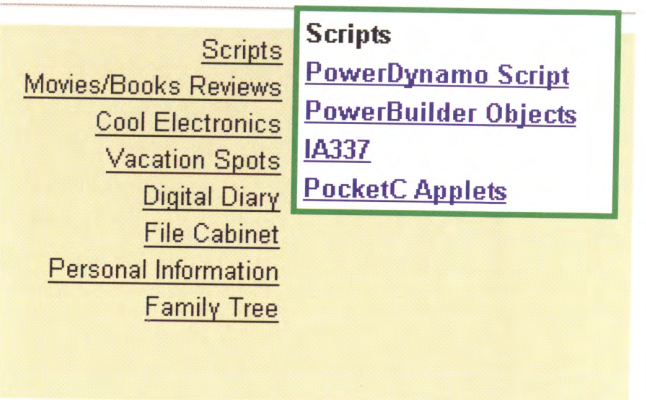
And so another year begins. Some exciting things and not-so-exciting things have happened since the writing of my last article. The company that I was working for failed miserably. Fortunately, I was in a position to see it coming, and have successfully landed a position with Accenture, formerly known as Andersen Consulting.

Last year I began a journey to mobilize the Movie Database application we've been looking at for quite some time now. I was planning to work with you on the full C application. Unfortunately, with the change in jobs I've been a little busier than expected and have yet to complete the application. However, it is well underway, although it's been a long time since I've had to write that much code. The development of Palm OS applications is not up to the same level as that of other environments. However, during the fourth quarter board meeting, I began talking with both Raj Nathan of Sybase's Internet Applications Division and Terry Stepien of iAnywhere (Sybase's new spin-off) about integrating Palm into the Sybase development tools.

First, I've got some code snippets to share with you, and then I would like to share my thoughts with you on how PowerBuilder could be enhanced to develop Palm OS applications. Rest assured that in the next issue I will be examining our expanding Palm application again. In the meantime, feel free to stop by my website (www.lamb.tj or www.lambtj.com) to see what is going on.

Layer upon Layer

Recently, I've come across some exciting new books, published by Peachpit Press, which are part of a Visual Quick Start Guide series. Two of them, covering JavaScript and XML, I've found to be very informative and are consistent with the approach I use for these columns. They present clear snippets of code with thorough explanations, and are written to help you find exactly what you are looking for when you need it. Armed with the JavaScript book and after exploring various sites, I've incorporated popup menus into my website.



As the mouse hovers over one of the main options in the menu, a popup menu is displayed with the available suboptions. However, not all browsers support JavaScript. For those browsers that do not, the page will display as it did before.

On my website, you can download the main JavaScript that provides the popup menu support, fom.js (short for "flyovermenu"). Let's take a look at how this code can be utilized on our page. By "including" the code we can make it more difficult for others to find out exactly how the popup menus are created. Additionally, we can encapsulate it to make it easier to reuse on other pages.

```
<SCRIPT LANGUAGE="JavaScript"
TYPE="text/javascript" src="fom.js">
</script>
<noscript>
</noscript>
```

Let's take a look at the HTML code that is needed to do this:

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
Layer[1] =        new LayerSpecs(160,345,160,
        '<B><font face=Arial,Helvetic,sans-serif
color="#000000" size=-1><a
name="#Scripts">Scripts</a></font></B>',
        '<A HREF="publish/index.html" alt="PowerDynamo
Scripts" onMouseOver="Show(1)"><B><font
face=Arial,Helvetic,sans-serif size=-1>PowerDynamo
Script</font></B></A>',
        '<A HREF="palm/index.html" alt="PowerBuilder Objects"
onMouseOver="Show(1)"><B><font face=Arial,Helvetic,sans-
serif size=-1>PowerBuilder Objects</font></B></A>',
        '<A HREF="ia337/index.htm" alt="IA337"
onMouseOver="Show(1)"><B><font face=Arial,Helvetic,sans-
serif size=-1>IA337</font></B></A>',
        '<A HREF="pocketc/index.html" alt="PocketC Applets"
onMouseOver="Show(1)"><B><font face=Arial,Helvetic,sans-
serif size=-1>PocketC Applets</font></B></A>'
);
1
//
LayerWrite();
//-->
</script>
<noscript>
</noscript>
```

The code between the <SCRIPT> and </script> tags defines the popup menus as layers on the HTML page. *LayerSpecs* is a function defined in fom.js that provides storage of Left, Top, Width, Title, and HREF tags for each of the popup menus. This function has been developed to provide ease of use, so I will avoid going into a great deal of detail on the function internals. Basically, the function builds a string containing information for a Table Row (if layers are not supported) or a Layer. At the end of scripts that define each of the popup menus, the function *LayerWrite* is called. This will write out each of the popup menus as either part of a Table, or as part of an inclusive Hidden Layer.

Now let's take a look at how this is created with PowerDynamo. I extended the code in common.ssc for the oTable object, oRows object, and the oBtn object by adding a new function, *writeflyover*, to each. Additionally, the oTable object has another new function, writeflyanchor. For the oTable object, *writeflyanchor* provides the anchor menu item that provides the code link when the mouse passes over it. The writeflyover function provides the call to the *LayerSpecs* JavaScript function.

```
function writeflyanchor( slayer ) {
    document.writeln( "  document.write(\"<TR><TD
Align='Right'>   <A HREF='index.html'
onMouseOver='Show(" + slayer + ")' onMouseOut='Hide(" + slayer
+ ")'>\");" );
    document.writeln( "  document.write(\"<font
face=Arial,Helvetic,sans-serif size=-1 color='#000000'>" +
this.slabel + "</font></A></TD></TR>\");" );
  }
  function writeflyover( slayer ) {
    document.WriteLn("Layer["+slayer+"] =      new
LayerSpecs(160," + (325 + (20 * slayer)) + ",160,");
    document.WriteLn("                          '<B><font
face=Arial,Helvetic,sans-serif color=\"#000000\" size=-1><a
name=\"#" + this.slabel + "\">" + this.slabel +
"</a></font></B>'," );
    var j = this.aRows.length;
    for (i in this.aRows) {
      this.aRows[i].writeflyover( slayer );
      if (j > 1) document.writeln( "," );
      j-- ;
    }
    document.writeln( "" );
    document.writeln("];");
  }
```

writeflyanchor provides the necessary function call to display the popup menu in onMouseOver event call to the function *Show*. Within the code for the writeflyover function, we see that there is an interaction for loop through the aRows array. This iteration generates each of the popup menus suboptions by calling the writeflyover function for the oRows object.

```
function writeflyover( slayer ) {
  var j = this.btn.length;
  for ( i in this.btn ) {
    this.btn[i].writeflyover( slayer, this.slabel );
    j-- ;
    if (j > 1) { document.WriteLn(","); }
  }
}
```

Since the "button" is what actually provides the hyperlink to the next HTML page, the button is what provides the hyperlink in the popup menu. Also, since the button is an array, the code iterates through calling the writeflyover for each of the oBtn objects within the oRow object.

```
function writeflyover( slayer, slabel ) {
  document.Write( "                    '<A HREF=\"" +
this.slink + "\" alt=\"" + slabel + "\" onMouseOver=\"Show(" +
slayer + ")\"><B><font face=Arial,Helvetic,sans-serif size=-1>"
+ slabel + "</font></B></A>" );
}
```

Finally, we can see that here at the button level we write the hyperlink. Notice the reference to onMouseOver: This event forces the menu to stay active when the mouse is dragged over the menu item by using the Show function as well.

PowerBuilding a Palm Application

During the course of this series I have identified three tiers of integration possible between PowerBuilder and a Palm application:

Tier	*Integration*
1	Query only
2	Distributed-style application using the GetFullState and SetFullState functionality
3	Full .prc and conduit application

Tier 1

In this environment we use a Palm OS Application similar to that found in our earlier exploration of jFile. However, this program automatically handles an extracted datawindow or datastore view with a full set of specialized display capabilities. Specifically, we would see at a minimum the following levels of operation:

Level	*Operation*
1	Present a list of available datawindows
2	Display a list of available rows within the datawindow: support filtering, sorting, etc.
3	Display a selected row: support all standard Palm OS control types (radiobuttons, checkboxes, popup lists (dropdowndatawindows), labels, editable fields, etc.

Tier 2

Extend the functionality defined in Tier 1 by supporting updating (Insert, Update, and Delete) functionality via a standardized conduit protocol that uses the datawindows *GetFullState* and *SetFullState* functions. The same Palm OS application could be used in both tiers with a specialized indicator being downloaded with the datawindow indicating whether the information is query-only. In this tier, we also extend the functionality of Level 3 to a fourth level of operation:

Level	*Operation*
2	Extend the row management functionality to include options for inserting and deleting rows. Normal select functionality would be enhanced to support updates.
3	Extend the editable fields to provide edit masks support date, time, etc. manipulation through standardized Palm OS control functionality.
4	Include validation functionality to insure accuracy of information entered. Additionally, support cancel capabilities to void an insert, update, or delete process.

Tier 3

In general, a Palm application consists of a database and a set of windows. Within these windows, we typically find that one window presents a list of the available rows, another displays information about a selected record, and a third provides editing and validation of a selected record. Additionally, there are several specialized functions available. These include *beaming*, setting preferences for fonts, support for the Palm's *find* operation, text clipping functions (cut, copy, and paste), application "About" information, etc. There is also the extra operation of moving the information to and from the PC, usually through a conduit.

Let's first examine the issue of a database. We've already looked at the concept of using a datastore to generate a Palm database. Within a PowerBuilder application, a datastore can be defined to utilize a set of stored procedures for insert, update, and delete processing.

Of course, we are already familiar with the capabilities of a datawindow. Within recent versions of PowerBuilder, we've seen the datawindow extended to include functionality for HTML pages. By extending its capabilities a little further, it should also be able to work within the environment of a Palm application. The application would need a specialized Window container object that would interact with menu options. Again, the menu could be created within the PowerBuilder environment using the standard menu painter.

There should be some additional validation when a Palm application is built to ensure that the right object ancestry is utilized, ensuring that the object will fit within the confines of a Palm interface. Of course this does not mean that a datawindow won't extend beyond the boundaries of a window supporting horizontal scrolling. Utilizing the functionality of buttons within the datawindow, we can provide full event support within our application.

Beaming presents a unique opportunity to extend the functionality of a pipeline. If the application allows beaming, it

could invoke a pipeline to send data from one handheld device to another with a defined mechanism, providing the necessary events to validate the information received.

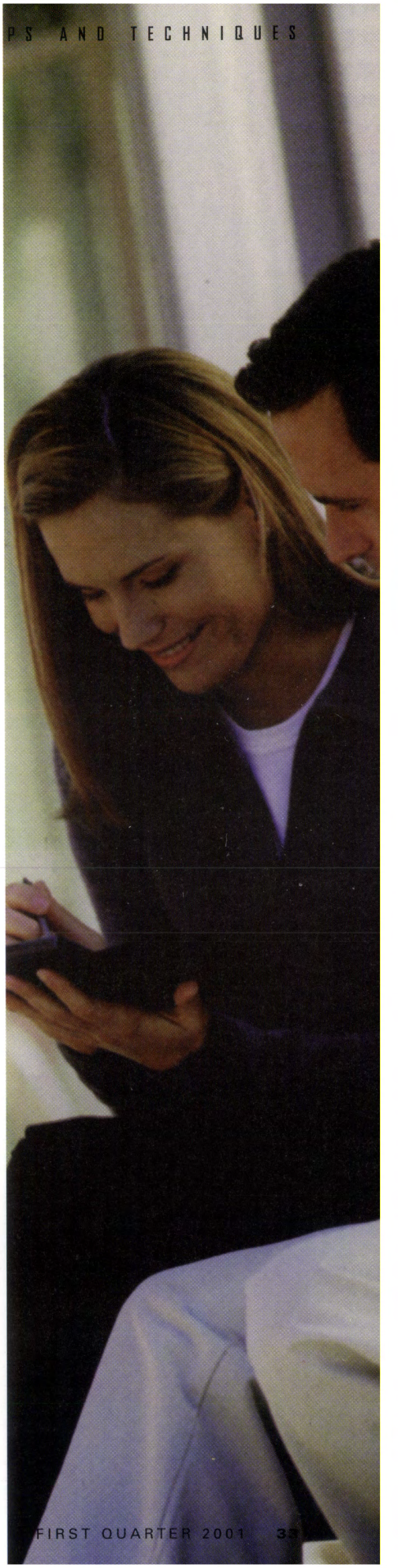
Supporting the Palm's *find* operation is simply a matter of providing the correct set of events to create an interface to process a call to search the applications database for a string and display the results. Additionally, the application needs to be able to respond to the *finds result* window, allowing the selection of a specific record.

Below, I've summarized the information in a table:

The one thing that I've not covered yet is conduits. iAnywhere has already begun making headway in this area by expanding the capabilities of Adaptive Server Anywhere to include Ultralite database technology. By combining this technology with IAD's enterprise application server technology, it is not a far stretch to envision the ability to have conduits that work in conjunction with an application server, allowing not only for Palm to PC integration but a full range of integration opportunities leveraging TCP/IP. This would open up the Palm application to being able to provide wireless synchronization through Palm query applications, as well as over a local area network, even a wireless modem connection.

You might say that this sounds impossible or too good to be true. Up until recently I might have agreed with you, given the state of application development available for the Palm OS. However, a company has recently done the very thing I'm proposing for a community of developers that utilize a tool that has for some time been considered competition (by some) for PowerBuilder. If that environment can be enhanced to provide this functionality, then certainly PowerBuilder and PowerJ should be able to as well. ■

PowerBuilder Object	*Palm Object*
Project	Defines a collection of related Palm objects used to create a PRC file. The project could also be extended to provide generalized version and about information for a standard Palm application "About" box.
Application	Provides initialization for a Palm application. Since Palm applications never really shut down, some of the behavior found in Windows applications is not necessary. Additional functionality should be provided for support of the Palm find operation.
Window	Used to provide basic encapsulation of a set of controls. Typically a datawindow control will be referenced to provide the bulk of the functionality. The window will provide primary response mechanisms to menu events.
DataStore	A representation of a Palm database.
DataWindow	The standard UI for interacting with a Palm database. Initially, this should support the functionality of lists, grids, and freeform.
Pipeline	Used to provide support for beaming data from one Palm device to another.
Menus	Provide standardized Palm menu functionality.



ISUG Board Launches New Initiatives for 2001

By Kathy Ridley, ISUG Vice President

In its final board meeting of the year 2000, the ISUG Board of Directors met in Emeryville, CA, at the Sybase headquarters on November 13–15 to determine the directions our organization would take in 2001. In an era of tightening user budgets at Sybase, it was decided that 2001 would be "The Year of the Local User Group"—in which the Board would provide measures to support local user group activities to encourage networking and information sharing on a local level. The most important of these measures include:

Integration with PowerBuilder Groups

Over the past two years, ISUG has pursued a policy of uniting local user groups (LUGs) focused on database and large systems with the PowerBuilder User Groups (PBUGs). We have found enthusiastic support among the users of every persuasion, and have been able to provide a number of new opportunities for developers, such as enhancements voting on the latest tool releases. In 2001, we will continue to work toward uniting our organizations, building a powerful user front to strengthen our position with Sybase. For more information on user groups and the benefits ISUG provides to developers, visit our website at www.isug.com.

ISUG European Road Shows

User conferences are big-ticket items, and one way in which Sybase is trying to cut costs is to rethink its approach to these large-scale events. In Europe, for example, Sybase and ISUG are cooperating to produce the ISUG Innovation Tour 2001, a series of road shows that will travel around much of the continent providing one-day seminars covering enterprise portal, database and Internet development and deployment products, ASE 12.5, and the latest on PowerBuilder. Countries to be visited include Sweden, Norway, the United Kingdom, Belgium, Germany, Spain, France, Ireland, and Switzerland. Again, for the latest schedule and to register for a road show, go to our website at www.isug.com.

Events in the Asia-Pacific Region

In addition to Europe, ISUG is considering taking the Innovation Tour to India and Taiwan. Although no specifics are currently worked out, the Hong Kong office may also once again be sponsoring a conference in China or the Asia-Pacific area.

Meanwhile, the North America TechWave conference will take place on August 12–16, 2001 at the San Diego Convention Center. We hope that positioning the conference on the West Coast will make it convenient for more of our Australasian members to participate there as well. You can pre-register for the North America conference by April 2, 2001, to receive a $100 conference discount. Complete the form at http://techwave.sybase.com/TechWave/preReg.stm for the discount and to have a chance to win a TechWave VIP package.

Support Materials for LUG Meetings

Sometimes users know they want to get together, but aren't very experienced at creating an exciting series of events. ISUG has developed an initiative called the "Meeting in a Box"—several kits of materials that can be sent out for use in user group meetings. These kits will include presentations, general and frequently asked questions, marketing materials, and handouts. Having such materials available will make it easier to plan engaging meetings and encourage greater member participation.

Supporting Wireless Communications

In 2001, ISUG is also formalizing its relationships with Internet Applications Division (IAD) and iAnywhere, Sybase's mobile computing spin-off, including providing increased enhancements request and voting processes. Web applications and wireless communications are increasingly prevalent in today's business world, and Sybase provides a number of technologies in support of IT personnel working in these areas.

For more information, contact Kathy Ridley at kathy.ridley@cgiusa.com or check out the ISUG website at www.isug.com. ■

ISUG Welcomes Board of Directors for 2001

Thomas Lamb, *President*
Thomas Lamb begins his second year as the ISUG President, having previously served as Vice President and SIG Director–Tools, and treasurer and webmaster for the ChicagoLand PowerBuilder User Group for two years. He has presented at several Sybase and PowerBuilder conferences on the local, regional, and international level, and writes a regular column for PowerBuilder users in the *ISUG Technical Journal*. He is currently a manager with Accenture (formerly Andersen Consulting).

Kathy Ridley, *Vice President*
Kathy Ridley begins her second year as vice president, after serving on the International Sybase User Group board for the past two years as Treasurer and SIG Director, and has been a member of the local user group board of Houston as treasurer and president. She is a director of consulting services for CGI.

Luc Van der Veurst, *Treasurer*
Luc Van der Veurst enters his second year as Treasurer, after having been ISUG President since 1997 and held the position of Enhancements Co-Director since 1994. He has been a key player in automating the enhancement process and maintaining the ISUG website, and has served as a member of several committees. He is a founding member of the Benelux user group and has been its president for the past four years.

Karen Pursch, *Secretary*
ISUG secretary Karen Pursch is in her second year on the board, and works as director of market analysis in corporate marketing at Sybase. She has been a Sybase employee for 11 years, holding numerous positions in marketing as well as North American sales. Prior to Sybase, Karen was a DBA at Hughes Aircraft, Rockwell International, and Jet Propulsion Laboratory.

Cindy Bean, *Conference Director*
Cindy Bean takes a new position as Conference Director, after two years as the ISUG North American RUG. She was secretary of the Sybase User Group of Texas, where she also served as president in 1997 and vice president 1995–96. She has led ISUG's Database and Administration SIG and participated in paper reviews for ISUG conferences. Cindy is a software consultant at BMC Software, Inc., specializing in the application availability products for Sybase.

Cynthia Gill, *ISUG Technical Journal Director*
Cynthia Gill takes over the *ISUG Technical Journal* after several years as ISUG's Conference Director. She previously served as SIG Director 1995–96. Cynthia was president for the Texas Sybase User Group, chairing Houston's first Sybase vendor fair. From 1993–1996 she managed and implemented the first Sybase training facility in the Southwest U.S. She is vice president of consulting services for CGI.

Jeffrey L. Roberts, *Electronic Media Director*
Jeffrey L. Roberts moves to the position of Electronic Media Director, having filled the position on an interim basis since mid-2000 and previously served as a Member-at-Large. He was the founder of the Atlanta PowerBuilder User Group and serves its current president, and was the founding editor-in-chief of the *Sybase Developer's Journal*. He is president and CEO of CCS Consulting.

David Johnstone, *SIG Co-Director (Database)*
David Johnstone is in his second year on the ISUG board as a Special Interest Group (SIG) co-director. He also serves as vice-chair of the UK Sybase User Group and has been a Sybase DBA for five years. He is employed at IBM Systems Operations Ltd.

Bryan Enochs, *SIG Co-Director (Tools)*
Bryan Enochs joins the board after serving as the Orlando PowerBuilder User Group president as well as the webmaster for the user group website for the last two years. Bryan is a senior technical staff manager at AT&T Solutions and has been worked with Sybase technology since 1994.

Jay Hunt, *Enhancements Co-Director (Tools)*
Jay Hunt becomes co-director of enhancements after serving as SIG Co-Director and as vice president of the St. Louis Sybase Internet Tools User Group. As a PowerBuilder and Internet solutions developer, he has published several articles in the *ISUG Technical Journal* on the use of Sybase tools in N-Tier architectures and presented at TechWave. He is a solutions architect for Parametrics Technology Corp. (PTC).

Philip Brantley, *Enhancements Co-Director (Database)*
Philip Brantley joins the board after having been involved with database development for over 15 years, and is currently Operation and Support Manager for Getronics Government Solutions. He has served as president and vice president of a local database users group and has been developing with Sybase for the past three years.

Joe Burger, *Membership Director*
Joe Burger enters his second year on the board, having worked with Sybase products since 1995 and been active in various local user groups for many years. He participated in the last five ISUG North America conferences and has been a member of the Paper Selection Committee for the past three years. As a DBA with The Longaberger Company, he leads the Sybase development and support effort.

Frank Monteverdi, *Partner Membership Director*
Frank Monteverdi returns to manage ISUG partnerships while continuing to serve as president of the Sybase User Group of Texas. He is a DBA for several databases, including ASE, and has presented at ISUG user conferences. Frank manages DBAs and technology experts for Deloitte & Touche, DRT Systems.

Richard Brooks, *North American RUG Director*
Richard Brooks joins the board for the first time, but has been active in the PowerBuilder community for several years. He was a presenter at TechWave 2000 and presents for Sybase user groups. He is webmaster of the first and longest-running PowerBuilder website, "Rick's PowerBuilder Dojo," is involved with the *PowerBuilder Developer Journal*, and has authored five books on PowerBuilder. He works as an independent consultant.

Brian Pollard, *North American RUG Director*
Brian Pollard is also new to the ISUG board, and has been a member of the Rocky Mountain region user group for five years. Since 1993, his role at Micron Technology, Inc. has involved working closely with Sybase products. He presented on middleware/data replication topics at TechWave 1999 and is currently sponsoring a pilot installation of the ASE 12.0 DTM premium feature at Micron.

Anthony Mandic, *Asia-Pacific RUG Director*
Anthony enters his second year on the ISUG board, and is an independent consultant specializing in database and system architecture and administration. He has worked with Sybase products for many years, both as a developer and as a DBA. He received the 1999 ISUG Outstanding Achievement Award for his efforts in the Sybase-related Usenet forums where, for many years, he has been found offering advice based on his knowledge and experience.

Ramesh Babu Ve, *Asia-Pacific RUG Director*
Ramesh Babu Ve joins the board as India's only ISUG member, and plans to raise ISUG awareness in his native country. He is currently involved in developing a component technology for retail trading houses using EAServer. Each component is developed in PowerBuilder and EJB using open database technologies for rich and thin-client environments.

Dorus Kruse, *European RUG Director*
Dorus returns to the ISUG board to represent Europe users for his third year. He is active as a member of the Dutch Sybase User Group (DSUG), which recently merged with the local PowerBuilder and the Sybase User Groups. Dorus works as product manager in client server technology at the Dutch Tax Office.

Linda Morison, *Member-at-Large*
Linda Morison returns as Member-at-Large after filling roles on the ISUG board including North American RUG director (1994), Vice President (1995), President (1996 and 1997), and Past President (1998). She also served on the Steering Committee for the Washington, DC Area Sybase User Group (DCA SUG) from 1992 to 1995. She is a senior database architect at Lockheed Martin Space Mission Systems and Services.

Jibu Abraham, *Member-at-Large*
Jibu Abraham changes positions to become a Member-at-Large after serving on the board for three years as an Enhancements Co-Director (Tools). He has presented on enhancements at ISUG conferences and via the ISUG website. He is employed at Jones Cyber Solutions. ■

INTERNATIONAL

Sybase User Group

ISUG Membership Application

ISUG Benefits

- NEW Free Copy of SQL Anywhere Studio for NT or Linux
- EA Server with PowerJ evaluation CD
- ASE 12.0 Technical Documentation CD
- A subscription to the quarterly ISUG Technical Journal
- NEW Free copy of the ASE Tables Diagram
- Access to the online ISUG membership directory
- Access to the online ISUG Enhancement Request process
- Technical Training at local user groups
- TechWave Conference proceedings CD
- NEW Free iAnywhere t-shirt, limited edition ISUG pen

Education Discounts

- Save 10% on Sybase education classes
- Save 20% on Certification Exams from Sylvan Prometric
- $75 discount on registration for TechWave and other ISUG-affiliated conferences

Note: Benefits are subject to change. Please refer to the ISUG website (www.isug.com) for current list.

please check one:

❑ individual membership*	US	$ 75.00
❑ corporate membership* (10 people)	US	$500.00
❑ government subscription	US	$ 75.00

* Members joining from Germany, the United Kingdom, Belgium, Luxembourg, the Netherlands, or France must join through their local user group. See LUG Directory for contact information.

Sybase Site ID or CBSS number (includes application & middleware product Site ID or CBSS number)

number _____

Your Site ID (CBSS number), found on your invoice or packing slip, is a 5-digit number followed by "- # -#".

please fill in the following contact information completely:**

name _____

title _____

phone _____

fax _____

email _____

company/organization _____

department _____

address _____

mailstop _____

city _____ state _____

postal code _____ country _____

If this is a corporate membership, please attach a separate sheet with the above information for nine additional persons.

Sybase products•

version platform

1. _____
2. _____
3. _____

•**Note:** *This information will be kept confidential.*

Are you a member of a Local User Group (LUG)? ❑ Yes ❑ No

Name of group _____

❑ Please send me information on the LUG in my area.

Are you a member of a Special Interest Group (SIG)? ❑ Yes ❑ No

name of group _____

I am most interested in the following SIGs:

- ❑ Adaptive Server
- ❑ Application Tools
- ❑ Architecture & Design
- ❑ Query & Reporting Tools
- ❑ Data Warehousing
- ❑ NT Server
- ❑ Systems Management
- ❑ VLDB
- ❑ WWW/Internet
- ❑ Middleware
- ❑ PowerBuilder

payment instructions

Please send a check made payable to "International Sybase User Group." Outside North America, please send a check for the currency equivalent.

Note: *All checks must be drawn from a US bank.*

To pay by credit card, please fill in the following information:

❑ VISA ❑ Mastercard ❑ American Express

card number _____ exp. date _____

cardholder signature _____

Return this form with check or credit card information by enclosing in an envelope and applying stamp.

membership directory release form

Contact information will be distributed in the membership directory to ISUG members only. Product information will not be released. Please sign this form if you want your contact information published in the ISUG directory.

signature _____ date _____

non-disclosure agreement

ISUG members agree not to use any Sybase confidential information for any purpose except their business relationship with Sybase. Members also agree that they will not disclose confidential information to any person other than their company's employees who are directly involved in the use of Sybase products.

signature _____ date _____

Mail form to...

International Sybase User Group
6475 Christie Avenue
Emeryville, CA 94608
Fax: 510-922-0882
E-mail: isug@sybase.com

Note: Members joining from Germany, the United Kingdom, Belgium, Luxembourg, the Netherlands, or France must join through their local user group. See LUG Directory for contact information.

Source Code J

Board of Directors

President

Thomas J. Lamb
Accenture
Chicago, IL
Phone: 312-693-6473
t.lamb@bigfoot.com

Vice President

Kathy Ridley
CGI
Houston, TX
Phone: 713-868-5537, x160
kathy.ridley@cgiusa.com

Secretary

Karen Pursch
Sybase
Boulder, CO
Phone: 720-890-0188
karenk@sybase.com

Treasurer

Luc Van der Veurst
Academic Hospital, VUB
Brussels, Belgium
Phone: 32-2-477-6980
lucv@az.vub.ac.be

Conference Director

Cindy Bean
BMC Software, Inc.
Houston, TX
Phone: 713-918-1841
Fax: 713-918-1173
cynthia_bean@bmc.com

ISUG Technical Journal Director

Cynthia Gill
CGI
Houston, TX
Phone: 713-868-5537 x1409
cynthia_gill@drthou.com

Enhancements Co-Directors

Jay Hunt
Parametric Technology Corp.
St. Louis, MO
Phone: 314-579-9644
Fax: 314-579-9495
djayhunt@charter.net

Philip Brantley
Getronics Government Solutions
Montgomery, AL
Phone: 334-416-2825
philip.brantley@gunter.af.mil

Membership Director

Joe Burger
The Longaberger Company
Newark, OH
Phone: 740-322-5091
jburger@longaberger.com

Australasia RUG Co-Directors

Anthony Mandic
Mandic Consulting Pty, Ltd.
Sydney, Australia
amandic@start.com.au

Ramesh Babu Ve
ewak Creative Compusoft Ltd.
Chennai, India
Phone: +91 44-6217523
rameshve@vsnl.net

Electronic Media Director

Jeff Roberts
CCS Consulting
Atlanta, GA
Phone: 770-582-0360
Fax: 770-582-0590
jroberts@consultCCS.com

Members-at-Large

Linda Morison
Lockheed Martin
Columbia, MD
Phone: 410-715-1127
Lmorison@erols.com

Jibu Abraham
Jones Cyber Solutions
Denver, CO
Phone: 303-784-3612
jibu_abraham@hotmail.com

Partner Membership Director

Frank Monteverdi
CGI
Houston, TX
Phone: 713-868-5537x149
Fax: 713-868-4014
Frank_monteverdi@drthou.com

European RUG Director

Dorus Kruse
Belastingdienst
Automatiseringscentrum
The Netherlands
Phone: 31 55 528 7945
doruskruse@hotmail.com

N. American RUG Co-Directors

Richard Brooks
Brooks & Young
Austin, TX
Phone: 512-680-4826
rik@rikbrooks.com

Brian Pollard
Micron Technology, Inc.
Boise, ID
Phone: 208-368-5437
bpollard@micron.com

SIG Co-Directors

David Johnstone
IBM Systems Operating Ltd.
Edinburgh, Scotland
Phone: 44-0131-442-7957
david_johnstone@uk.ibm.uk

Bryan Enochs
AT&T
Orlando, FL
Phone: 407-858-8320
bryan@HotTurf.com

Sybase Contact:

Aline Martin
User Group Marketing
Sybase
Emeryville, CA
Phone: 510-922-7525
Fax: 510-922-0882
amartin@sybase.com

Sybase User Group Directory

European Region

Austria
Dr. Wolfgang Lipa
ZAMG
Phone: 43-0222-36-44-53 x 2603
Fax: 43-0222-369-1233

Belgium "Blues"
Luc Van der Veurst
Academic Hospital VUB
Phone: 32-2-477-6980
lucv@az.vub.ac.be

Denmark
Lars Henrichsen
COWI Consult A/S
Phone: 45-45-972068
or 45-45-972211
Fax: 45-45-972212
lh@cowi.dk

Estonia
Gustav Seier
Tallinn, Estonia
Phone: 372 6655467
gustav.seier@eyp.ec

Finland
Kristiina Salminen
Sanoma Corporation
Phone: 358 9 122 87 3184
kristiina.salminen@sanoma.fi

France "Fibonacci"
M. Gerard Lledo
C.E.A.
CENTRE ETUDES SACLAY
Phone: 33-1-6908-9616
Fax: 33-1-6908-9608

Germany

Sybase User Group
Jens Timmermann
TOPOLOGIX GmbH
Phone: 49-40-352-253
Fax: 49-40-340-340
dsug@topologix.de

PowerBuilder User Group
Ludwin Feiten
Power People
Vorstand@pbugg.de
www.pbugg.de

Ireland
David Quirke
Phone: 00 353 1 677 6777
Fax: 00 353 1 6776614
dquirke@sybase.ie

The Netherlands
Jolanda Zwaan
Phone: 036 5226249
Fax: 036 5226249
dsug@tref.nl

Norway
Irene Negarden
Norsk Hydro ASA
Tel: +47-22738987
Irene.Negarden@hydro.com

Spain
Manuel Blanco Ulled
Unipapel, S.A.
Phone: 91-806-96-10
Fax: 91-803-52-22
mblanco@unipapel.com

Switzerland
Dr. Roberto Buzzi
Zurich Versicherungs
Phone: 41-1-205-2121
Fax: 41-1-205-3375
chzurkrb@ibmmail.com

Turkey
Levent Sensezgin
Sybase Turkey
Phone: 90-212-284-8339
Fax: 90-212-284-8342
levent.sensezgin@sybase.com.tr

United Kingdom
Khosro Parnian
Sensitek, Ltd.
Phone: +44-0171-325-0374
parnian_khosro@jpmorgan.com

UK PowerBuilder User Group
Anne Bocock
PB Associates
Phone: 020-84213533
Fax: 020-84201420
anne@pbug.co.uk

International

Argentina
Sergio Di Cuffa
Sybase Argentina
Phone: 5411-4313-4488
Fax: 5411-4315-8834

Australia
Igor Geninson
IBS Technology
Phone: 61-02-388-7675
Fax: 61-02-388-8067

Brazil
Luiz Guilherme Mendonca
LOCUS Ltd.
Phone: 55-21-553-3086
Fax: 55-21-551-8461
lguilherme@locus.com.br

Chile
Jean Pierre Lefranc
Sybase Chile
Phone: 56-2-3306700
Fax: 56-2-3306800

Israel
Yoval Malchi
ITG, Inc.
Phone: +972-9-970-2739
Fax: +972-9-970-2727
ymalchi@itginc.com

New Zealand
Shayne Duncan
Sybase NZ, Ltd.
Phone: 64-4-473-3661
Fax: 64-4-499-9068
shayne@sybase.com

Saudi Arabia
A. Omran
Al-Omran
Phone: 96-61-4662373
Fax: 96-61-4662502

South Africa
Rudie Bronkhorst
Phone: +27-11-8043740
rudie@sybase.co.za

Thailand
Chakkrapong Tongsak
Bangkok University
Fax: 65-273-9159
ct@lily.bu.ac.th

United Arab Emirates
George Khouri
Sybase Products - Middle East
Abu Dhabi
Phone: 97-1-2-325-911
Fax: 97-1-2-340-850

Sybase Local User Groups of North America

Northeast Region

Regional Contact
Jan Barcelou
Barings Asset Management
Phone: 617-946-5325
Fax: 617-946-5410
Jan.barcelou@baring-asset.com

Albany, New York PowerBuilder User Group
Greg Fisher
gfisher@dttus.com
www.anypbug.org

Boston PowerBuilder User Group
Olga Demidova
SOFTRAX Corp.
ODemidova@softrax.com
www.middlewarecg.com/pbug

Boston (SNEKUS–Sybase North East Kingdom Users Society)
Judy Ohnemus
Sybase Inc.
judy.ohnemus@sybase.com

Harrisburg, PA, PowerBuilder User Group
Dennis Rehm
dennisrehm@appliedcomputing.net

New Jersey PowerBuilder User Group
Boris Gasin
Phone: 973-402-5600
bgasin@dynamictechgroup.com
www.njpbug.org

Ottawa
Tony Antonallo
Phone: 613-798-7507

Ottawa PowerBuilder User Group
Carole Hargrave
Saphire Consulting Services, Inc.
Phone: 613-292-2018
opbug@opbug.com
www.opbug.com

Philadelphia
Sybase, Inc.
Phone: 610-260-4300
Fax: 610-260-4399

Philadelphia Sybase IAD User Group
Chuck Miller
cmiller@appltech.net
pbug.fcg.com/PBUG/index.html

Toronto PowerBuilder/ Sybase User Group
Paul Bis
Tpbug@interlog.com
www.interlog.com/~tpbug

Southeast Region

Atlanta PowerBuilder User Group
Jeff Roberts
CCS Consulting
Phone: 770-582-0360
Fax: 770-582-0590
jroberts@consultccs.com
www.apbug.org

Baltimore Area PowerBuilder User Group
Fred Grau
fred-g@vips.com
www.geocities.com/bapbug

Boca Raton
John Glover
Data Breeze
Phone: 954-427-4416 x309
Fax: 954-427-0280
jdg@phamis.com

Charlotte PowerBuilder User Group
Alan Augustine
aaugustine@ttpartners.com
www.charpbug.org

Charlotte Sybase User Group
Bob Kunkle
Phone: 704-375-5788

Greensboro/Piedmont Triad PowerBuilder User Group
James Stoertz
jstoertz@fitech-gso.com
www.triadpbug.com

Louisiana PowerBuilder User Group
Joe Sherrill
joesherrill@msn.com
www.notrs.com/lapbug

Miami
Ray Liera
University of Miami
Phone: 305-284-4207
Fax: 305-284-4753
Rleira@miami.edu

Music City Powersoft User Group
Jeff Gibson
jgibson@frontrunner.com
www.frontrunner.com/mcpug

Nashville
Jimmy Hogan
Phone: 615-641-5550
Fax: 615-841-5556

North Carolina
Clay Bullard
SMCI
Phone: 704-375-5788
Fax: 704-375-5699
smcic@ix.netcom.com

Northwest Arkansas Sybase User Group
Dan Luttrell
vpbo@vicsinc.com
www.cyberspace.org/~nwapbug

Orlando PowerBuilder User Group
Bryan Enochs
AT&T
Phone: 407-858-8320
Fax: 407-858-8352
benochs@ao.net
www.wedowebs.com/opbug

Tampa Bay Area
Jeffrey Garbus
Phone: 813-949-7016
Jeffg@soaringeagleltd.com
www.soaringeagleltd.com/sybase_local_user_group.htm

Washington Area Sybase Enterprise Application Tools User Group (WA/SEATUG)
Theo Rushin, Jr.
National Institutes of Health
Phone: 301-402-4609
Fax: 301-480-6105
rushint@mail.nih.gov

North Central Region

Regional Contact
Gaynel Walden
AT&T
Phone: 816-995-3723
gaynel@att.com

Central Ohio PowerBuilder User Group
Barry McDonald
bmcdonald@CBSInc.com
www.cmnpbug.com

ChicagoLand Sybase Tools User Group
Michael Baraz
Phone: 630-235-4529
mbaraz@tradesolutions.com
www.cpbug.org

Cincinnati/Dayton
Scott Stegman
Phone: 513-241-5949
Fax: 513-241-6731
sstegman@clientserver.com

Cleveland PowerBuilder User Group of Northeast Ohio
Todd Senauskas
tsenauskas@datavantage.com

Columbus/Central Ohio PowerBuilder User Group
Barry McDonald
bmcdonal@CBSInc.com
www.cmhpbug.com

Des Moines PowerBuilder User Group
Mary Eagan
meagan@ifmc.org

East Michigan PowerBuilder User Group
Benny Cheung
benny@mpbug.org
www.mpbug.org

Indianapolis
Julie Clark
Phone: 317-501-7559
jclark@moser-inc.com
www.cs.bsu.edu/homepages/sam/isug

Kansas City Area Sybase User Group
George Meiers
Hallmark Cards, Inc.
Phone: 816-545-6395
Gmeier1@hallmark.com

Michigan PowerBuilder User Group
Jagdish Karira
Phone: 248-524-3280
jagdish@techie.com

Minneapolis, MN
Maendra Saraswate
Hunter Software
Phone: 612-943-3986
Fax: 612-941-0933
msarasw@primenet.com

Minneapolis/Twin Cities Sybase Tools User Group
Bob Sharp
Phone: 612-483-1506
www.tcpbug.com

PowerBuilder User Group of Northeast Ohio
Todd Senauskas
tsenauskas@datavantage.com
www.internet-club.com/usa/neopbug

St. Louis Sybase Internet Tools User Group
Dave Blankenship
president@stlpbug.org
www.stlpbug.org

Winnipeg
Jim Novisat
Great–West Life Assurance Co.
Phone: 204-946-7748
Fax: 204-946-4567
jzn@gwl.ca

Wisconsin Sybase User Group
Michelle Murphy
Wisconsin Electric Power Co.
Phone: 414-221-2068
Fax: 414-221-4744
michelle.murphy@wepco.com

Bill Mitchell
Northwestern Mutual Life
Phone: 414-299-4022
Fax: 414-299-1686
Billmitchell@northwesternmutual.com
www.reveregroup.com/wisug

SW-Mountain Region

Regional Contact
Cindy Bean
BMC Software, Inc.
Phone: 713-918-1841
Fax: 713-918-1173
cynthia_bean@bmc.com

Austin
Rob Gustavson
BCS Systems, Inc.
Phone: 512-921-0074
Rgustavson@rfdinc.com

Colorado PowerBuilder User Group
Mark Juodawlkis
mjuodawlkis@gr.com
www.co-pbug.org

Colorado Springs PowerBuilder User Group
Judith McEntee
jmcentee@gr.com
www.camelotconsulting.com/cspbug

Dallas Sybase User Group
Phil Adams
FireSteed Software, Inc.
Phone: 817-296-6238
Fax: 817-282-8980
pcadams@firesteedsoftware.com
www.firesteedsoftware.com/tsug.html

Dallas/Ft. Worth PowerBuilder User Group
Marcie Jones
marciejones@yahoo.com
www.dfwpbug.com

Houston (Sybase Users' Group of Texas)
Tony Broussard
TradeCapture.com
Phone: 713-752-4063
admin@sugtx.org
www.sugtx.org

Inter-Mountain/Utah
Dianne Garcia
Discover Brokerage Direct
Phone: 801-902-4222
Garcia@discoverbrokerage.com

Oklahoma City PowerBuilder User Group
Dianna Demotto
okcpbug@usa.net
okcpbug.iwarp.com

Rocky Mountain/Denver/ Salt Lake City
Doug Thomas
Time Warner Telecom
Phone: 303-566-1432
Fax: 303-566-1400
dougtho@geocities.com
rmsug.intranets.com

San Antonio Sybase User Group
Bill Purnell
purnell@texas.net

San Antonio PowerBuilder User Group
Richard Carrier
rcarrier@tesoropetroleum.com
www.pfccheatsheet.com/pbugsa.htl

Tulsa PowerBuilder User Group
Mike Deasy
mdeasy@williams.com
www.geocities.com/tulsa_pbug

Wichita Kansas PowerBuilder User Group
Rodney Hughes
Phone: 316-517-1847
rjhughes@cessna.textron.com

Far West Region

Regional Contact
Cynthia Gill, CGI
Phone: 713-868-5537
cynthia_gill@drthou.com

Arizona Sybase/Powersoft User Group
John Lewis
oberon@home.com
azspug.webjump.com

Hawaii
Sterling Yee
Hawaiian Electronic Industries
Phone: 808-532-5870
Fax: 808-532-5828
syee@hei.com

Los Angeles
Cory Isaacson
Compuflex
Phone: 818-772-7990
Fax: 818-772-7999

Los Angeles PowerBuilder User Group
Brian J. Smith
Phone: 310-221-0798
75530.332@compuserve.com

Orange County PowerBuilder User Group
Victor A. Reinhart
Phone: 714-226-8253
victora.reinhart@phs.com
www.ocpbug.com

San Diego PowerBuilder User Group
Joe Hermiz
Phone: 619-334-7638
joe@workcompsoftware.com

Sybase Internet Application Tools User Group
Scott McGarrigle
Applied Knowledge, Inc.
Phone: 714-751-3480
siatug@aki.com
www.aki.com/siatug

Northwest Region

Regional Contact
Matt Townsend
Ayupp Inc.
Phone: 707-829-1173
Matt_townsend@earthlink.net

Boise
Michelle Featherston
Micron Semiconductor
Phone: 208-368-4498
Fax: 208-368-1043
mfeatherston@micron.com

Calgary/Edmonton
David Owen
dowen@midsomer.org

Edmonton PowerBuilder User Group
Theodore J. Allen
73614.3174@compuserve.com
www.freenet.edmonton.ab.ca/
~vschmid/pbug.htm

Pacific Northwest/Seattle (SNUG)
Chris Young
Cascade Software, Inc.
Phone: 206-224-3725
Fax: 206-224-6205
cyoung@cascadia-sw.com
www.cascadia-sw.com/snug

San Francisco PowerBuilder User Group
Amine Khechfe
Phone: 650-321-2640 x104
a@psi-systems.com
www.sfpbug.org/

Seattle/Northwest PowerBuilder User Group
Curtis Hanner
curtis@hanner.com
www.cascadia-sw.com/nwpbug

Vancouver PowerBuilder User Group
Dennis Lee
Phone: 604-893-7040
dlee@vusual.com

SYBASE®

WHO Developers, DBAs, Systems Analysts, Architects, Consultants, Engineers, and Technical Managers

WHAT Sybase TechWave 2001
User Training and Solutions Conference

WHERE San Diego Convention Center, San Diego, California USA

WHY Get answers to e-Business questions through training and education, networking with thousands, advance product news.

WHEN August 12-16, 2001

NOW Pre-Register
www.sybase.com/techwave2001
or call 781-278-2607 for Sybase TechWave 2001 Customer Service

SYBASE
TechWave 2001
USER TRAINING & SOLUTIONS CONFERENCE

e-Business. anytime. anywhere.