Out in Left Field
strategic tools for strategic teams
Volume One, Issue Three

DDS or SQL

...does it matter?

The most common questions to people new to SQL relate to creating database objects:

- Are physical files and SQL tables the same? If not, how are they different?
- What exactly is an SQL view and how does it compare to a logical file?
- Is there an advantage to SQL over DDS?

Tables: When a programmer creates an SQL table, it looks exactly like a physical file. However, there are several important differences between a table and a physical file. The first difference is that an SQL table can only have a single member. Because the SQL standard does not include the concept of physical partitioning, multi-member physical files are not allowed. If you try to add a second member to an SQL table, an error will be signaled.

A second difference is that, by default, deleted records are reused in SQL tables. This means that they do not have to be reorganized to recapture the space wasted by deleted records. It also means, however, that applications that depend on newly inserted records being put at the end of the file will not work.

Thirdly, SQL tables are created so that they do not have limitations on their size. In other words, the maximum size is set to *NOMAX which means that there is no "safety net" if an application has a bug.

Lastly, SQL tables do not allow data items to be inserted that would introduce corrupted data. A common problem with physical files created with DDS is that invalid decimal data gets inserted into packed or zoned fields. When data is inserted or updated in an SQL table, the data is immediately verified. If invalid decimal data exists, the operation is rejected and an error is signaled.

Views: An SQL view is similar to a logical file. Like a logical file, a view can include a subset of the fields from a physical file or generate new data using substring and concatenation operators. Views can also have filtering criteria and thus have exactly the same function as a select/omit logical file. Like logical files, views can also be defined to join several files together.

On the other hand, SQL views never have an associated database index. Almost all logical files are defined with key specifications. This means that programs that use the logical file read the data in the order defined by the key fields. If a view is used without an associated order by clause, the data may come back in any order. The advantage of views over DDS-based logical files is the power of SQL. Almost all syntax supported by the SQL SELECT statement is supported by views. This means that sophisticated calculations, CASE logic, grouping functions, and advanced forms of joining tables together are supported. This allows the application programmer to embed a significant amount of logic into the file being read and thus improve the functionality and time to create applications.

(Continued on page 2)

Complimentary Comprehensive CompTechCall©

Maximize the business value from your iSeries investment. There's still time to sign up for your CompTechCall© today.

What is CompTechCall©? To help us better understand the iSeries and increase our level of quality service to our client-partners, Centerfield offers a limited number of: complimentary, comprehensive, no-obligation, no-sales person involvement, one-hour sessions to help you devise a plan of attack.

Have one of our DB2 experts help you and your company in areas such as:

- *Database and SQL performance tuning*
- *System and work management configuration*
- *Control of end-users running queries*
- *Access path (index) management*
- *OneWorld® / EnterpriseOne® performance*
- *Application diagnostics*
- *Batch job performance*

We can help you understand your iSeries challenges. It is easy - just send an e-mail to: comptechcall@centerfieldtechnology.com

Centerfield Technology, Inc.
1312½ 7th Street NW
Rochester, Minnesota 55901
888.387.8119
www.centerfieldtechnology.com

IBM announces Tools Innovation Program

Centerfield Technology participated from day one of this exciting program for iSeries partners. Follow the developers roadmap and see CTI "Improve Your Productivity" at http://www-1.ibm.com/servers/enable/tools/innovation/improve.html

Centerfield helps write IBM performance redbook

Centerfield Technology staff recently participated in an IBM residency to write a redbook on SQL Performance. When published, the Redbook will guide the user in diagnosing SQL performance issues relating to the database of the iSeries. Once diagnosed, corrective action-step recommendations are provided.

We look forward to seeing the results of the residency Summer 2005.

CTI seeks Customer Advisory Board members

Interest? Contact us at 888.387.8119 ext 101

Interested in becoming a development-partner? We often look for development-partners in order for us to gain a better understanding of day-to-day real world administrative needs and concerns.

(DDS or SQL—Continued from page 1)

Indexes: In SQL, an index object is independent of both tables and views. In the world of SQL, indexes simply exist to improve performance – they are not necessary for any function. When an index is created with SQL, a "logical file" appears in the library that holds the index. This logical file, however, is marked as an index and is not directly usable by an SQL statement. If a SELECT statement references an index, an error is signaled.

Much like a logical file with a unique key, an SQL index can be created to enforce a constraint to eliminate duplicate key values.

Summary: So is there an advantage of SQL over DDS? At the end of the day the answer to that question depends on your requirements. If you have a need to maintain strict compatibility with existing RPG applications you may not want to experiment with SQL. On the other hand, if you would like to take advantage of increased data integrity, the power of SQL, and the advantage of application portability, then give SQL a try!

iSeries Prestarted Jobs

...like a bullpen that's already warmed up

Let's discuss a real life iSeries scenario:

You have users using Excel, Microsoft Access, Crystal Reports, QBEs (Query By Example), etc., that are hitting your iSeries system hard. Most of the time this is manifested as QZDASOINIT jobs hogging system CPU.

Your knee-jerk reaction is to either hold or end the jobs in question and that will work in the short term by temporarily reclaiming system resources. This solution is woefully inadequate, though, in the long term, particularly from the end user perspective. If there are no process or system changes enacted to address these jobs, chances are problems will occur again.

What are prestarted jobs?

All modern client/server architecture designs use connection pooling for purposes of performance enhancement and isolation of similar work in the same subsystem. Some platforms achieve this through the use of threads, while others achieve it through the use of processes. The iSeries term for 'process' is 'job', so iSeries connection pooling for the purpose of enhanced performance is achieved through the use of ***prestarted jobs*** (PJs). This is a set of jobs that are pre-started by the subsystem and are waiting for work to come over the network connection. When the work requests arrive, their status is changed from PSRW (prestarted wait) to some other, active state (i.e. RUN, DEQW, etc.). When the client terminates the connection, this job is sent back to the pool to be reused for another connection. This way the expensive overhead of starting a new job is avoided and end user throughput is significantly increased.

Am I taking advantage of connection pooling?

Most likely you are. An easy check is to see if you have the new (as of V5R1) QUSRWRK subsystem active and if there are QZDASOINIT or other Q* type jobs running under it. But what if they're not? An experience I had with a customer will serve to illustrate:

I happened to be on a customer system and noticed that there were no PJs on their machine. As I was using ODBC, I was curious as to how my requests were being satisfied. After some investigation, I found the answer:

This particular customer is a long-time 400 shop. They have been upgrading their machine from V4* versions of the OS through V5*, all the way to V5R3. They also happened to implement their own startup program (QSTRUP) long ago and have pointed the system value QSTRUPPGM to their modified version. Over the course of these upgrades the customer failed to notice that IBM changed the system-shipped startup program and those changes were not migrated to their modified version.

It turns out that at V5R1 IBM added starting of the new QUSRWRK subsystem to the QSTRUP, implementing ODBC connection pooling by having QZDASOINIT database server jobs run as PJs under that subsystem. Connection pooling for a number of other network interfaces was also implemented with PJs running under QUSRWRK.

In true and solid IBM tradition, a fallback plan is automatically implemented when this subsystem is not active, and that is to revert to traditional processing as batch immediate (BCI) jobs. Since this customer's startup program was never modified to include the V5R1 QSTRUP change, they were not starting the new QUSRWRK subsystem and hence the jobs were still running as BCI jobs under QSERVER. I performed a RTVCLSRC PGM(QSYS/QSTRUP) SRCFILE(&YourLib/&YourSrcFile) SRCMBR(*PGM) and pointed out the differences between the customer's program and IBM's.

Why the long story?

The above illustration is important since the following performance recommendations assume that you are using IBM-recommended connection pooling via PJs. I have recently been involved with some customers complaining of PJs using too many

(Continued on page 3)

Major League Performance

The salary cap in major league baseball forces owners to think about getting the best players for the least amount of money. Your CIO or VP thinks the same way – only their decisions are harder to make because their choices don't come with a batting average or on-base percentage.

The phrase "Return on Investment" (ROI) means different things to different people but the basic meaning is clear to everyone – get a higher payback from a budget item than the money spent on the item or service. Today, the focus on obtaining a good return is as high as it has ever been due to increased pressure to control costs and hold to budgets.

It can be rather difficult to determine a return on investment, especially if the savings or improvements must be estimated in advance and are hard to verify even after the fact. The purpose of this article is to provide a starting point to justify the cost of performance tuning software.

Performance problems appear in several different forms. Probably the most obvious is when users complain about the time it takes to complete a transaction or look up information. Another common performance problem is when batch work does not complete in its designated window. This can result in "downstream" ripple effects when those jobs are part of a critical business process. Both of these issues become more critical when they impact customers or partners that expect and depend on responsive applications that interact with your business.

Performance problems cost money in many ways:

- Wasted time and productivity for in-house personnel trying to do their jobs.
- Delays to business processes that result in unnecessary interruptions to invoicing and other product shipments.
- The cost of hardware upgrades to solve the performance problem.
- The time and money for system administrators, programmers, or outside consultants to diagnose the problem, propose solutions, and implement those solutions. Since these solutions also introduce risk to system operation, they can also cost more than originally planned if they introduce unexpected problems.
- Customer dissatisfaction and reduced loyalty if systems they use are slow and frustrating to use.

At a high level, each of these has a unique cost for each business. On the other hand, these costs can easily be $100,000 - $200,000 per year when the problems impact time-critical processes or hundreds of end-users. Lets look at a couple of different scenarios to fix performance problems:

Buy hardware: The purchase of new hardware is often one of only two solutions that IT shops explore due to lack of time or knowledge about alternative solutions. While this may be the correct approach in some instances, it is not necessarily the least expensive (and thus not the one with the best return on investment). If you look at the costs to do a hardware upgrade it might look something like this:

(Continued on page 4)

(iSeries Prestarted Jobs—Continued from page 2)

resources. There is no single right answer to this issue, but a good starting point is ***fine-tuning the prestarted jobs***:

Quick fix

If you are a traditional green-screen RPG type of shop and usage of network jobs for ODBC, file serving etc. is incidental and is really not your core business, changing the priority of the class these jobs run under may be a quick fix to your headaches.

1) CRTDUPOBJ OBJ(QPWFSERVER) FROMLIB(QSYS) OBJTYPE(*CLS) TOLIB(QSYS)NEWOBJ (ODBCCLS)
2) CHGCLS CLS(QSYS/ODBCCLS) RUNPTY(21)
3) CHGPJE SBSD(QSYS/QUSRWRK) PGM(QSYS/QZDASOINIT) CLS (QSYS/ODBCCLS)
4) CHGPJE SBSD(QSYS/QUSRWRK) PGM(QSYS/QZDAINIT) CLS(QSYS/ODBCCLS)

NOTE: the above example uses priority 21 to illustrate a simple change so these jobs run with priority lower than your green-screen interactive jobs. If you want your batch work superseding these jobs' priority as well, you may want to use priority 51. Change is immediate for any new PJs.

Another quick fix may be to isolate this type of work in its own memory pool. I usually don't recommend over-analyzing the OS's allocation of system resources, but it does make sense to separate this type of network induced interactive work in its own sandbox.

Getting serious

If, on the other hand, you are using your iSeries as a true enterprise level database server, you're in for more work. If you have not done it yet, you MUST tune your prestarted job allocations and thresholds.

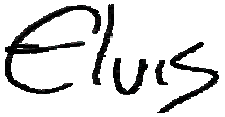
Since iSeries tune up settings are unique to iSeries work loads, this type of tuning can only be accomplished through careful observation of the impact of changed settings via the CHGPJE command.

The rule of thumb for this type of tuning is to always have enough prestarted jobs waiting for work so that the arriving requests don't have to spend any time waiting for the jobs to start. Having a high number of prestarted jobs autostarted will mean that it'll take longer for the QUSRWRK subsystem to fully start. However, I strongly believe the benefits of the added end-user throughput far outweigh the cost of longer subsystem startup time.

In baseball, superior development, management, and strategic deployment of relief pitchers is key to winning, kind of like tuning iSeries prestarted jobs.

For exact details on how to tune prestarted jobs I refer you to time proven IBM experience report: **Tuning prestart job entries**

Fun Reading!



Elvis Budimlic
Development Director
ebudimlic@centerfieldtechnology.com

(Major League Performance—Continued from page 3)

1. Go from 2 to 4 processors – $20,000
2. Increase memory from 2GB to 4GB – $5,000
3. Downtime to do the upgrade – 5 hours (at $1000/hour x 5 hours = $5,000)
4. Problems introduced by new version of the operating system (3 problems x 20 hours of diagnosis and follow-up @ $90/hour = $5400)

Total cost = $35,400

HIRE CONSULTANTS: Hiring consultants to solve performance issues can also be very expensive. Because the consultant has to learn a complex environment from scratch, diagnose issues on a live production system, and mesh with existing personnel, the costs can be quite high. The other issue with bringing in outside consultants is that there is often no knowledge transfer to existing IT staff, creating a one-time-only return on the investment. A two-week engagement can easily cost $16,000 or more when the time of in-house personnel to assist and implement solutions proposed during the engagement is factored in, as recommendations must be phased in over time to minimize risk and to accommodate other projects being implemented at the same time.

Cost of the total solution? Over $50,000. A cost that must be incurred (and justified if possible) again when the performance problems re-emerge.

A BETTER WAY could well be application of a set of performance tuning tools that is relatively inexpensive, easy-to-learn, and sophisticated enough to quickly uncover the root causes of the performance problems while *helping to justify* a properly sized hardware upgrade and continuing to add value as in-house staff becomes more experienced in the application of the tools.

After all, smaller market baseball teams have consistently shown that paying attention to the fundamentals of management and sound skills development while creating a cohesive team environment enables them to WIN BIG while SPENDING LESS.

If you would like more information or help in justifying the cost of a performance project let us know. We can provide a complimentary customizable spreadsheet to help you calculate the costs of the performance problems and the cost of alternative ways of fixing them.

STEROIDS for your Database… …GOOD or BAD?

While steroids are very controversial topic in sports, wouldn't it be nice to be able to put a little more muscle on your database so it could do extraordinary things? A few years ago IBM introduced a legal (but not free) technique to boost the performance of DB2 called parallelism.

This technique, one of the most innovative technologies IBM has introduced for IBM® DB2® Universal Database™ (UDB) for iSeries™ in recent years, has been the ability to use more than one CPU for a single request. With this technology, the database can take advantage of processor capacity that may otherwise be left idle. If you have a system with multiple processors, the capability to run database requests with multiple CPUs is enabled with a licensed feature called DB2 Symmetric MultiProcessing (DB2 SMP).

Centerfield recommends the use of DB2 SMP in situations where the elapsed time of the database operation needs to be reduced. Typically, this includes batch jobs that use SQL, when SQL-based reports are run against large files (including data warehouse environments), and when database maintenance needs to be done quickly (reorganizations, etc.). Lets take a look at each of these situations.

Batch jobs that use SQL tend to either generate reports or do a large number of file modifications. The DB2 SMP feature can be of benefit for both the reading and writing of data. If an SQL statement is run in parallel on four processors for example, it would not be unusual for the total elapsed time of the query to be slightly over one-fourth of the previous time. When rows are added, changed, or removed from a file, the access paths (indices) over that physical file must be done. With SMP installed, more than one index can be updated at a time.

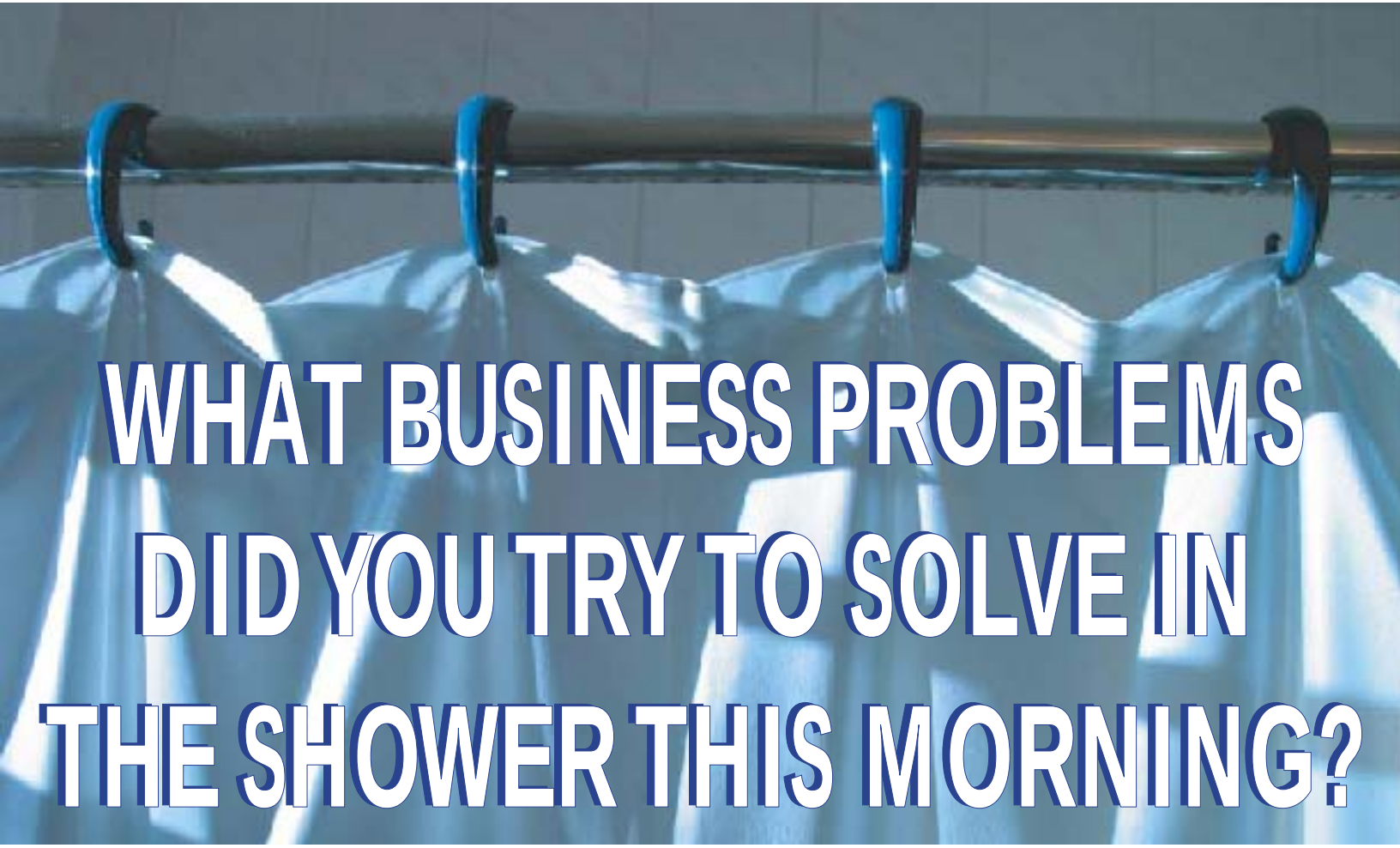
In a reporting environment, especially one with large files, SMP can result in significantly faster completion times. The SMP support breaks the work for a single SQL statement into parts and assigns each of the subtasks to different processors. If the query is predominantly CPU bound, the time to complete the request will scale down with the number of processors used to service the query. Data warehouse environments, which can have very large files combined with complex questions, SMP can provide a performance boost that easily justifies its cost.

The third area SMP can benefit system administrators is in the area of file maintenance. If it becomes necessary to rebuild access paths, the SMP feature allows the database to exploit more than one processor to build the index. Because index builds are very CPU intensive, this allows the system to apply resources to reduce the bottleneck of a single processor.

The Danger: So is there a downside to the SMP feature? The short answer is yes. Once you give the database permission to use multiple processors, any request can potentially use more than one CPU. If an unimportant report or batch job executes a query that runs in parallel, it can take more system resources than you want it to take. Unfortunately, if you turn on the parallel support with the system value, all jobs are now capable of using more CPU resources than you may want them to take. The alternative is to use the CHGQRYA command in each job to get the right level based on the jobs importance to your business.

The good news is that Centerfield has a fully configurable product called insure/RESOURCES that includes a way to police the use of SMP based on your unique requirements (among many other policy options). This allows you to define the rules and then let the database do the rest. For further information on this product go to the following link: http://www.centerfieldtechnology.com/tools/insureresources.asp.

insure/RESOURCES can help you put DB2 on steroids without causing all of the bad side-effects you can get by "juicing up" your requests.



WHAT BUSINESS PROBLEMS DID YOU TRY TO SOLVE IN THE SHOWER THIS MORNING?

That special iSeries database project needs to be done *NOW*
You don't have the time or the expertise in-house
Budget for outside consultants - already stretched thin

Call Centerfield

Centerfield takes a unique approach to iSeries database tuning and manipulation. Because the Centerfield toolset is composed of modules of code that can be combined and recombined in a myriad of ways - chances are that the code needed to complete your project is already 90% written. This portability, mixed with the skill sets possessed by the Centerfield development team, can provide you with very fast project turn-around - at an extremely low cost.

In less than two weeks, the experts at Centerfield Technology were able to put together a tool that reports input and output files within an AS/400 query...a job that would have taken many programmer hours to complete if we had done it on our own. All for a price that would have barely covered an outside consultant's travel and living expenses." Steve Burton, Senior P/A, LMI Aerospace

We make your business applications run better on the iSeries

Centerfield TECHNOLOGY

If you need custom iSeries security, SQL tuning or database manipulation done quickly at a *packaged tool price*, call us today for a CompTechCall® or write comptechcall@centerfieldtechnology.com

Centerfield Technology, Inc.
1312 1/2 Second Street NW
Rochester, Minnesota
888-387-8119
www.centerfieldtechnology.com

IBM Business Partner
IBM Server Proven
IBM server TOOLS NETWORK