nocoug JOURNAL

Official Publication of the Northern California Oracle Users Group

VOL. 15, No. 4 · NOVEMBER, 2001 $15

Survey Says!

Results of Online NoCOUG Survey

After NoCOUG's Summer Conference on August 23, we sent out a special request by email for attendees to participate in our online survey. Thank you to the dozens of you who responded! Your participation in our online survey helps us understand your needs and your interests. At our NoCOUG board meetings, we refer to the feedback you give us when planning future NoCOUG Conferences. Here's what you had to tell us:

About You:

- ➤ About 60% of you work for larger commercial companies, about 20% work for smaller commercial companies, and the remainder are split between government, education, nonprofit, and independent consulting.
- ➤ About 60% of you are DBAs, about 20% are application developers, and the remainder hold various other titles including Director, Manager, and CTO.

(continued on page 4)

Don't Miss Our Fall Conference!

The NoCOUG Board has planned a great conference that you won't want to miss. It's taking place on Thursday, November 15 in South San Francisco. Sponsored by Genentech, the one-day conference will have three parallel tracks:

- ➤ DBA
- ➤ Development
- ➤ Data Warehouse

Registration begins at 8:00 a.m. For more details, see our website at www.nocoug.org. ▲

TABLE OF CONTENTS

Editor's Note

On behalf of the NoCOUG Board, I would like to extend a huge **'Thank You' to Colleen Childers** of Database Specialists for her assistance in the production of the NoCOUG Journal. Her outstanding work makes this volunteer effort possible each quarter!

In other news . . . it is my goal as Journal Editor to make sure all members receive the NoCOUG Journal prior to each quarterly conference. However, many NoCOUG members did not receive the August Journal before our August Summer Conference because of address label errors at our mailing house. The mailing house mailed the publication again to rectify their mistake. Therefore, some of you may have received the Journal after the conference, while others received an additional copy due to the second mailing. I apologize for any inconvenience.

—Lisa Loper, *NoCOUG Journal* Editor

Other User Groups

Local

NorCalOAUG – Northern California Oracle Applications Users Group

- **Contact:** Michael Capelle (650) 562-1167
- **Email:** capelle@tru-course.com
- **Website:** www.norcaloaug.org

Sacramento

SacOUG – The Sacramento Oracle User Group

- **Contact:** Ravi Verma (916) 705-3261
- **Email:** ravi.verma@telcommand.com
- **Website:** www.sacoug.org

International

IOUG-A – International Oracle Users Group of the Americas

- **Website:** www.ioug.org

U.S. Domestic

OAUG – Oracle Applications Users Group

- **Website:** www.oaug.org

ODTUG – Oracle Development Tools User Group

- **Website:** www.odtug.com

Canvassing calls by employment recruiters to local chapter contacts is strongly discouraged.

Publication and Submission Format

The NoCOUG Journal is published four times a year by the Northern California Oracle Users Group approximately two weeks prior to the quarterly regional meetings. Please send your questions, feedback and submissions to: Lisa Loper, NoCOUG Journal Editor, at journal@nocoug.org.

The submission deadline for the upcoming February issue is January 2, 2002. Article submissions should be made in electronic format via email if possible. Word documents are preferred.

NoCOUG does not warrant the NoCOUG Journal to be error-free.

ADVERTISING RATES

Contact: Nora Rosingana

325 Camaritas Way
Danville, CA 94526
Ph: 925/820-1589

The NoCOUG Journal is published quarterly.

The rates are:

Size	Per Issue	Per Year
Quarter Page	$100	$320
Half Page	$200	$640
Full Page	$400	$1,280

Personnel recruitment ads are not accepted.

PRESIDENT'S MESSAGE

by Vilin Roufchaie, President, NoCOUG

Vilin Roufchaie.

Being engaged in a volunteer organization, such as the Northern California Oracle Users Group (NoCOUG), in any capacity, is a very fun and rewarding experience.

The rewards are the by-product of what one accomplishes along the way. For me personally, it has allowed me to take charge of various responsibilities, create and manage programs, meet and work with some highly respected professionals, stay on top of the technological evolution and somehow affect its course.

Working alongside fellow board members, one will notice their quest for excellence and their drive to make NoCOUG a great success. Over the course of fifteen years, the volunteers of NoCOUG have been putting together conferences, bringing to the Bay Area some world-class Oracle experts as speakers and trainers, publishing a quarterly technical journal and maintaining a superb website with information you can use in your daily activities as an Oracle user.

Today NoCOUG stands tall and very well established. The core of NoCOUG's board of directors is comprised of experienced and dedicated volunteers whose aim has been delivering the very best, highest-quality programs for the Oracle user community. This year we have had five outstanding professionals join the board. Please join me in welcoming: Lisa Loper, Database Specialists, Inc., who as journal editor transformed our quarterly publication; Roger Schrag, Database Specialists, Inc., who as webmaster has greatly improved the quality and content of our website (www.nocoug.org); and both Darrin Swan, LECCOTECH, and Hamid Minoui, Fritz Companies, Inc., who as board members-at-large have made great contributions. Welcome to our newest member, Ganesh Sankar from Providian, who just joined us in October.

I also want to express my gratitude to our long-time board members: Merrilee Nohr for her many years of dedication to NoCOUG and for her work this year as membership director; and Judy Lyman for her contributions as Treasurer/Secretary and for her outstanding assistance during our conferences.

Joel Rosingana's contributions as vice president and acting IOUG-A Representative have been enormous. He has been solely responsible for searching, soliciting and procuring of all of the NoCOUG's conference venues for the past few years. Along with many other roles Joel has assumed above and beyond his formal undertakings, he has assisted me whenever needed. Without Joel, it would not have been the same. Thank you, Joel!

I cannot possibly thank all of my colleagues enough for their invaluable contributions throughout these years.

My special thanks also goes to Nora Rosingana who, as a paid consultant in the organization, has always gone beyond what has been asked of her in making matters straight and keeping our books void of flaws.

In the past four years of my involvement with the NoCOUG, I have witnessed several individual volunteers join the board, all getting involved to varying degrees in making things happen for this Oracle user community. I am certain that there are a lot more talented folks out there among the membership. Are you ready to step forward and volunteer some of your time to help enhance the qualities of this magnificent volunteer organization for the Oracle user community? Here are some ways you can get involved:

- ➤ Participate as a conference speaker
- ➤ Contribute an article to the NoCOUG's quarterly journal
- ➤ Offer your corporate site as a venue for a conference
- ➤ Help out at a conference as a volunteer
- ➤ Exhibit at a future conference to showcase your products
- ➤ Join the board

See you at the next conference! ▲

Share Your Knowledge!

Have you worked on a project lately that others would be interested in learning about? Do you have some great scripts that all your friends like to borrow? Do you know of some great resources on the web that other Oracle users should know about? Want to become a published author?

Write for the NoCOUG Journal!

NoCOUG was formed to provide a forum for sharing information within the Oracle User community. It all starts with you, our members. If you've got an idea for something you'd like to contribute to the NoCOUG Journal, write to journal@nocoug.org today.

Fall Conference Highlight

Oracle performance tuning expert Guy Harrison is coming to the Bay Area to give a two-hour presentation at the NoCOUG Fall Conference on November 15.

GUY HARRISON
Oracle SQL
High-Performance Tuning
SECOND EDITION
THE PRENTICE HALL PTR ORACLE SERIES
THE INDEPENDENT VOICE ON ORACLE

Guy's presentation is titled "Top 10 SQL Tuning Tips." Guy is Director of Monitoring and Oracle Tools Development at Quest Software, and he is well known for his very popular books including:

- *Oracle SQL High-Performance Tuning*
- *Oracle Desk Reference*

Mark November 15 on your calendar now, so you won't miss out on this great opportunity. ▲

Author Guy Harrison to speak at NoCOUG's Fall Conference.

TECH TIPS

NcFTP Is the Smart Way to FTP

Have you ever been downloading a huge file, just to have it stop halfway through and force you to start all over from scratch? Well, with NcFTP Client, you may be on your way to hassle-free downloading.

NcFTP Client (also known as just NcFTP or "nice FTP") is a set of FREE application programs implementing the File Transfer Protocol (FTP). NcFTP will remember where your download stopped and hold your place while it keeps trying to reconnect to the file server and resume the transfer.

The NcFTP homepage states "the program has been in service on UNIX systems since 1991 and is a popular alternative to the standard FTP program. NcFTP offers many ease-of-use and performance enhancements over the stock FTP client, and runs on a wide variety of UNIX platforms as well as operating systems such as Microsoft Windows and Apple Mac OS X."

See for yourself at **http://www.ncftpd.com**.

Thanks to Oracle DBA Jay Stanley of Database Specialists for this Tech Tip.

Survey Says!

(continued from page 1)

The Technology You Use:

- Over 90% of you are running Oracle 8i in production, but at the same time, two-thirds of you still have Oracle 7.3 or Oracle8 databases in production as well.
- About one quarter of you use PL/SQL as your primary programming language or development tool, with another quarter using Java, and about 15% each using C and Oracle Forms.

What You'd Like to See in Our Conference Sessions:

- Over 90% of you find technical presentations by Oracle Corporation to be of interest.
- Only half of you find non-technical presentations by Oracle Corporation to be of interest.
- About 80% of you believe technical presentations by third-party vendors discussing their products to be of interest.
- A resounding 98% of you find technical presentations by Oracle users to be of interest.

After our Fall Conference, we'll be sending out another request for you to participate in our online survey. It's as easy as visiting **http://www.nocoug.org/survey.htm**. It only takes one or two minutes of your time to register your thoughts and help us make NoCOUG an organization that repre-

SPACE EXPERT™ GIVES YOU A SMARTER WAY TO *visualize* YOUR DATABASE'S SPACE PROBLEMS.

It's not always easy to see where pesky space problems are lurking, ready to degrade the performance of the database that your business depends on. And it's not always easy to keep your database operating at peak levels without lots of highly trained DBAs.

Until now, that is.

Introducing Space Expert™ for Oracle from BMC Software. It intelligently and automatically visualizes, isolates, analyzes and corrects space-related problems. Enabling even a novice DBA to perform the work of many. Which means the IT department's SLAs are good as gold. IT managers can concentrate on managing. Your database's performance is enhanced. And your company is more competitive than ever.

See for yourself. **Register for a 30-day trial copy of Space Expert for Oracle today at www.bmc.com/distdata/spaceexpert.** Then you'll discover just how easy we make it for you to help your database, and your business, really perform. Because it takes intelligence, not hocus pocus.

bmc software

Assuring Business Availability®

Oracle Optimal Flexible Architecture in a Nutshell

By Sekhar Palli, LSI Logic, Inc.

What is OFA?

We keep hearing, "Did you follow OFA in implementing your database?" Let's try to define it first. In a nutshell, OFA is a set of guidelines that will allow administrators and architects to arrange the database for better performance, more reliability and less time in maintenance.

Why should you implement OFA?

- Easier maintenance by standardizing file organization
- Increased reliability by spanning data over multiple physical devices
- Increases performance by decreasing I/O contention
- Helps to eliminate fragmentation of free space in the data dictionary, isolates other fragmentation, and minimizes resource contention
- Organizes large amounts of complicated software and data disk to avoid device bottlenecks and poor performance
- Easier switching among multiple Oracle databases
- Adequately manages and administers database growth
- Facilitates routine administrative tasks such as software and data backup functions, which are often vulnerable to data corruption

What are OFA Guidelines?

- Name all mount points (device and directory trees) that hold database files to match a predetermined pattern.
- Name home directories matching a predetermined pattern.
- Refer to explicit path names (device and directories) in as few files as possible.
- Store different versions of Oracle Server software in directories that match a predetermined pattern.
- Store administrative files in a predetermined directory pattern.
- Name database files using a predetermined pattern.
- Separate database segments by segment characteristics in different tablespaces.
- Name tablespaces descriptively with a minimum of characters.
- Choose a small set of standard sizes for character-special files.
- Select one and only one node to be the administrative home for a cluster.

How do I go about it, if I already have databases implemented?

Here are some steps, which you can follow in order to convert your environment if it is not OFA compliant. Before attempting any kind of operation on your production server, absolutely have a complete backup of the system. Do not attempt this if you are not comfortable with the database administrative procedures.

- Run a trace on your environment to find out the current layout and sizes.
- Design and create an OFA complaint directory structure, which you want to implement.
- Modify the database initialization file.
- Move the initialization file to the new location.
- Copy/Move/Rename the data files.
- Re-create the control file script to reflect the new OFA structure.
- Rename the database.
- Run create control file script.
- If ORACLE_HOME is changed then make sure your environment variable reflects the new structure.
- Change all your scripts (Ex: backup, scheduled jobs) to reflect the new directory structure.
- Do a clean shutdown of the database and take a complete backup of the new system.

Conclusion

Figure A gives an example of an OFA directory structure. You must have at least two physical devices to reap the benefits. There is no better time to implement OFA in your environment. If your environment is growing faster with multiple applications and multiple versions of a database, it is better to bite the bullet and implement it. The job seems very scary, but it is not painful. Obviously be careful and only attempt to implement OFA when you are fully confident and qualified.

Disclaimer

Please note that this article in no way negates the need to obtain a copy and to fully understand the OFA guidelines and how they apply to your environment. Any representation or statement expressed or otherwise implied is made in good faith but on the basis that the author does not give any warrant or reliability or accepts any responsibility for any errors or omissions.

About The Author

Sekhar Palli is currently working as Sr. Applications Manager at LSI Logic, Inc. He can be reached at **spalli@lsil.com** for any comments.

The following table represents one possible implementation of OFA-compatible file system on Unix.

File System	Directory Structure	Note
/u01/	/u01/u01/oracle/product/816 |-- /dbs |-- /network/admin |--/SID/ARCHIVE |--/admin/SID |-- /adhoc |-- /adump |-- /bdump |-- /cdump |-- /create |-- /exp |-- /logbook |-- /pfile |-- /udump |- /oradata/SID |-- /control_01.ctl /redo_01a.log /redo_02a.log /redo_03a.log	Oracle Home Archive Files Oracle Base Oracle Control File Oracle Redo Log Files Member A
/u02/	/u02/oradata/SID |-- /control_02.ctl /redo_01b.log /redo_02b.log /redo_03b.log /system01.dbf /SID_index01.dbf /temp01.dbf /rbs01.dbf	Oracle Control File Oracle Redo Log Files Member B Oracle Data Files Ex: System01.dbf Ab_Index01.dbf Temp01.dbf Rbs01.dbf
/u03/	/u03/oradata/SID |-- /control_03.ctl /SID_data01.dbf /SID_data02.dbf	Oracle Control File Oracle Data Files Ex: Ab_data01.dbf Ab_data02.dbf
/u04/	/u04/oradata/SID/backup |-- /hotbackup/ /coldbackup/ /dba/	Backup and DBA scripts are placed in this directory

Figure A.

Perl and FastCGI: A Viable Alternative to Java Server Side Applications

By Timothy Niesen, Mike Musta, and Milton Hankins

In today's fast-paced economy, customers and users are demanding thin-client applications that can be deployed across the enterprise. Their user interface of choice is fast becoming the web browser. By enterprise, we mean applications that can be deployed and utilized by clients across entire organizations. An organization could be a project, a group of projects, a laboratory, or an entire division or corporation.

This paper discusses lessons learned in using Perl and FastCGI in developing and supporting applications that utilize web-based technology running databases on the backend. It is not our intention, however, to indicate that the use of Perl and FastCGI is better than a Java approach, only that it provides a viable alternative to deploying small to large enterprise web-based systems. Perl provides a robust language and many on-the-shelf libraries that can quickly put together and deploy an application. FastCGI provides a means to greatly increase web server performance. Together, they can provide performance similar to many existing traditional windows-based client-server architectures.

Why Perl?

Much has been written about Java technology, and how it is revolutionizing the software world. The majority of the actual "dirty" work being done on the web and net in general is still done by Perl. It is by far the most popular language/tool for system administrators, web developers and anyone who needs to glue together varying systems or develop highly customizable/flexible web applications. One of our favorite descriptions of Perl comes from Sun's own first webmaster, Hassan Schroeder, who said: "Perl is the duct tape of the Internet."

Technologies such as Java and ActiveX have huge financial backing in spreading their virtues to the world. Perl simply has a group of dedicated programmers, webmasters, system administrators and the like continuing with its development and growth. Perl, as a language, tries to be a combination of the best of C, UNIX shell and other scripting languages such as sed and awk. Because of this it quickly won the approval of many UNIX system administrators, because it was so well suited to performing system administration tasks with a minimal amount of coding and development time. With the emergence of the web, Perl found another niche, which has caused its use to explode. For a while, many people thought those newer Internet technologies such as Java or ActiveX would slowly push Perl aside. This has not been the case as of yet. Some of the things that set Perl apart from the rest are [O'Reilly97]:

- Unequalled ability to process text with an enormous amount of built-in functions as well as power features such as regular expressions
- Availability and platform independence. Perl is available on virtually every UNIX platform, as well as NT, VMS, OS/2 and many others
- It is embeddable and extensible
- Enormous, active, dedicated user community
- Huge online archive of freely available pre-packaged libraries and modules
- Last, but certainly not least, it's FREE

Some general benefits of Perl are that it is fairly easy to code, platform/architecture independent, extensive number of external modules/libraries exist on the Internet, and supported by an enormous and extremely active user community. Perl 5 supports enhanced regular expressions, complex data structures, embedded C/C++ functions, and object oriented programming capabilities. Best of all, its FREE! In some cases, you get what you pay for, but so far this hasn't been the case with Perl. The major drawback is that it is an interpreted language. However, one should note that only the first time through is each line interpreted, after which Perl runs as fast as any compiled code, much like Java. Overall, we have not seen much lost in performance, if any, over complied C/C++ applications.

DBI, the database interface for Perl 5, gives the developer a powerful set of libraries that provide a platform and database independent interface for applications on the Web. DBI provides a simple and standard connection to the database via CGI. It allows one to connect, fetch, or modify database objects, and to disconnect to a variety of databases distributed over one or more servers. You can even connect to different databases (Ingres, Oracle, Sybase, etc.) at the same time. If your application needs to be ported from Oracle to Ingres, you need not change a thing at the application level. The only changes necessary are to develop similar schema and database objects. [DBI97].

Why use the Web?

Over the past few years, the web has changed just about every way we think about developing and deploying applications. The web provides a solution to two major problems of client server computing. The first is the transition from the fat-client architecture to a thin-client architecture where a web browser can run simple to complex applications on a low-end computer or even a network computer. The second is that administration and maintenance need only to be performed on the server, dramatically reducing costs. FastCGI greatly increases the performance of the web and database applications, while Perl allows you to rapidly develop applications. Bluestone Inc. cites three advantages for web-based applications over traditional client-server applications [Blue96]:

➤ Platform Independent: Provides seamless integration of a heterogeneous platform environment.

➤ Flexible: Application resident on the server implies no change for the client. Allows rapid response to changing competitive and market environments, both internal and external.

➤ Cost Effective: The single front-end graphical user interface capability provided by the web browser significantly reduces the cost of implementation, installation, and training.

These three reasons are very appealing when you consider that many companies today are composed of geographically separated organizations. The work still needs to get done despite the distances involved, but it is not always feasible to physically co-locate projects. In addition, many projects and heterogeneous legacy relational databases must be accommodated.

Web Database Architectures

We have found the use of web technology and HTML to be a convenient way to distribute information internally. The benefits of this approach include ease of development, ease of use, standard presentation over multiple platforms, and access to real-time information. JavaScript can be used to provide client-side error and style checking before a call is made to the Web server and the database. (Examples include validating data, performing date and numeric value checks, and determining if values have been entered into mandatory fields.) It is important from a performance standpoint to check this information at the client side rather than finding a simple error at the server side after connecting to a database. Typically, access to databases is accomplished in one of four ways: Common Gateway Interface (CGI), web server API, Plug-Ins, or Java. However, we have found that running a product called FastCGI (from Open Market) with Perl provides for very sound Web/DB applications.

The FastCGI Alternative

FastCGI is a product from Open Market that provides a fast, secure, and open web server interface for solving the performance problems resulting from using CGI or proprietary APIs. The major difference between CGI and FastCGI is that processes are persistent. After a process request is completed, it simply waits for a new request instead of exiting. In the strictest sense, FastCGI is a protocol, not an application or library. This allows CGI applications to run on dedicated machines, away from the web server. Rob Saccoccio has written an excellent implementation of FastCGI for the Apache web server.

FastCGI provides the best of both worlds for CGI and API developers. Some of the major advantages include [Brown1]:

➤ Processes are persistent and are reused as needed (thus solving the major performance problem inherent with CGI).

➤ Persistent processes mean that the database connection is maintained, eliminating the need to connect and disconnect to a database during each transaction. This is a significant performance benefit as we have seen our applications run up to 200% faster than traditional CGI.

➤ Language independence (unlike APIs, you can use Perl!)

➤ Non-Proprietary - works with most free (Apache, NCSA) and commercial servers (Netscape, Microsoft)

➤ Support for distributed computing (we have not tried this)

➤ Process isolation: FastCGI does not crash the web server or other applications. Errors are caught and dealt with by the application.

One of the greatest problems with traditional CGI scripting in a high-demand environment is the time needed for the OS to load the CGI binary into memory and execute it. This happens for each hit on a CGI-produced page. If the CGI program is written in a scripting language, then additional overhead is required for the script to be parsed. CGI persistence mechanisms such as FastCGI eliminate this per-page overhead by moving it to the web server's start-up routines. A CGI program then exists as some form of daemon; it processes individual requests in a queue [Brown1].

This infrastructure permits one to separate the web servers from the FastCGI servers for better load balancing. This frees the web server from being bogged down by CGI load. This leads to cost savings because one can spend money on machines with faster I/O and bandwidth for the web servers, for instance, and then concentrate on good database connectivity and processor power for the FastCGI servers [Brown[1]].

It is relatively easy to move current CGI applications to FastCGI. However, it is often a good technique to merge all of the scripts under the con-

Convergence
Two come together as one.
VERITAS enhances the power of Oracle.
ORACLE
SOFTWARE POWERS THE INTERNET

When it comes to accessing e-business information, two tracks converge for the surest route to success. Oracle and VERITAS.

VERITAS provides the essential data availability infrastructure for Oracle and is now available for HP environments. You get complete solutions – Oracle aware and integrated end to end. With VERITAS supporting your Oracle database, you'll have the leading data availability solutions integrated with the leading database.

If your database is Oracle, your data availability solution should be VERITAS.

VERITAS™

BUSINESS WITHOUT INTERRUPTION™

www.veritas.com/oracle

trol of one FastCGI process. This makes things easier to manage.

Present alternatives to FastCGI include mod_perl, SpeedyCGI, Zope, and Java servlets. Each of these is a language-specific solution. SpeedyCGI and mod_perl can only be used with Perl (and the latter only with the Apache web server). Zope is primarily designed for Python (but Perl support is moving right along). Java servlets are designed to run Java bytecode.

Support for Netscape/IPlanet and Microsoft IIS web servers is now in question because Adero, Inc. bought the previous vendor (Fast Engines). At this point in time, Adero is not making the FastCGI extensions available for these web servers.

FastCGI can be somewhat difficult to configure, particularly for the Apache web server. There are no GUI tools that come bundled with the main distribution. One of the authors of this paper has had a very hard time getting it to dynamically start new FastCGI daemons on an as-needed basis, but the feature is there and presumably works for other people.

Database Strategy

The general strategy is to imbue the database with as much semantics as possible, i.e., to approach as closely as possible the concept of an intelligent server. This is even more important in the web environment than in the client/server environment. The additional gain is garnered from the encapsulation of each transaction into a database procedure; this is essential in the CGI environment that does not support state and session maintenance.

We emphasized a consistent, well-named, ordered approach to the five aspects of database design: structure, integrity, authorization, manipulation, and performance. Also considered as part of the overall database design is a datasuite designed to test our design.

The structural design was based on functional dependency principals and the requirement to merge the bug-reporting tool of two separate Raytheon divisions, migrating decades of data to our web-based tool.

The integrity design utilized declarative constraints where possible; installing database rules if that was not possible. Only static validation criteria such as format, mandatory field and non-volatile data value checking were done in the front end, the rest of the integrity requirements were enforced in the database.

The manipulation schema, i.e., the collection of all legal manipulations of base table data, is defined by and encapsulated in database procedures (DBPs), which are called by any front end or a rule. There are many advantages to this approach where direct table modification by any front end is not permitted; all modifications are made only via database procedure calls. There is a strong parallel with data normalization, where one fact is stored in one place. Here with "procedure normalization," each procedure is stored in one place only — the database. Advantages arising from this "oneness" and from the location of the procedures are:

- ➤ **Development** - a single, extractable from the system catalogs library of DB access routines are created. Division of labor and expertise: DB experts write the stored procedures, and GUI experts create the front end by utilizing the database procedure library.
- ➤ **Maintenance** - changes, enhancements made in one place, recompilations reduced, new versions stay in the database, no redistribution required.
- ➤ **Security** - access to data only via DBPs.
- ➤ **Consistency** - only valid transactions via DBPs are allowed.
- ➤ **Intregity** - rules enforce integrity constraints, calling DBPs based on DB State.
- ➤ **Performance** - improvement occurs in network traffic, disk I/O, memory utilization, and hence CPU utilization. Instead of multiple, larger transmissions, now only the DBP name and parameter values are sent to the server. The DBPs are precompiled once (at creation time) executed many times, rather than compilation occurring with each DB request passed to the server. Commonly invoked DBPs will remain in memory, reducing disk I/O; further, although the data portion of the DBP is not sharable; the text portion is sharable, reducing memory requirements. When transaction/query optimization is required, it is extremely advantageous that all DB access routines have one copy in one place.
- ➤ **Portability** - Since all DB access routines reside on the DB server, this obviates any need to have more than one version of any DB access routine. Although the DBPs are written in Ingres-specific 4GL, and they would have to be rewritten if a decision were made to change to another DBMS vendor, the application code has been written to an API defined by the DBP calls, and as such needn't change. Only the library of DBPs and rules would need to be rewritten.

FastCGI Web Database Applications at Raytheon

We have a software trouble reporting system in production supporting many geographically separated projects and users. The application has been very reliable and users have found it very easy to start using. It works fairly quickly, but we found that performance, like anything, is dependent on network, bandwidth, and machine processing/memory. Overall, it makes running an application seems to be as fast as a traditional client/server application. We improved one application by 600% simply by running it on a server that contained the application, database, and a local copy of Perl. Another benefit of this approach is that it only uses the database as needed (thus reducing the total number of needed concurrent run-time licenses and potential locks) and runs on just about any platform with a very thin client (all you need is a browser). Another application is an Inspection System to handle issues found at reviews (code, design, requirements, documentation, etc.). Many other small to medium scale systems have been implemented as well.

Developing with FastCGI

Information on FastCGI can be found at http://www.fastcgi.com. Libraries and other items need to be set up on your machine. The instructions differ depending on your installation platform and web server. Open Market provides support if you purchase the software from them, but a mailing list and FAQ's exist to handle most questions. Every FastCGI application runs the following skeleton: 1) initialize variables and invoke one-time startup calls, 2) perform the main request processing loop, and 3) clean-up when terminating the process. Let's look at each of these in more detail.

The first step is to figure out what things must be done once during initialization and what ones must be accomplished for each request. Performance can depend on what you can do once and cache during initialization (such as opening a database).

The second step is establishing a processing loop to handle each request. At this point, unlike a CGI, one should not exit the process unless something is seriously wrong. We have utilized a wrapper around the typical FastCGI application to handle new requests and catch any errors that occur. We have routines that deal with specific errors (database, CGI, and server) and then return back to handle new requests. For most errors, we display the error message on a black screen that the user can read. Some errors are automatically mailed to the support team. This wrapper comes with the standard FastCGI library package. Additionally, another wrapper to set up a pool of database connections allows us to determine if a database was already opened so that one can utilize an already open connection to a database. The wrapper is called DBIPOOL and also comes with FastCGI. A max timeout option automatically disconnects a connection after it has been inactive for some time. From then on, a main-request processing loop that handles the different commands (such as Add Trouble Report, Query Trouble Report, etc.) provides the user access to desired screens [Brown2].

Java vs. Perl Metrics

Some of us had taken a Java class though Oracle training and were disappointed by the performance problems we saw with very simple applications. In light of the poor performance observed we decided to perform some simple test cases when we returned. We did a simple comparison test between Perl w/DBI-OCI and Java w/JDBC-OCI and the results were very, very surprising.

We prepared a simple program with the following query of sample data from our common bug tracking system: 'select * from wstr'. There are 1028 entries in this wstr table. So the query returned 1028 rows of 45 columns. We looped on the query 100 times. Results are as follows:

- ➤ Java - 522 seconds to run 100 queries, or 0.192 queries per second. —poor.
- ➤ Perl - 30 seconds to run 100 queries, or 3.33 queries per second. —good.

In this simple case Perl was 17 times faster than Java!

Each program looped as follows 100 times:

- ➤ create/prepare a statement
- ➤ execute the statement
- ➤ print only the wstr number from the result set (even without printing, Perl was 6 times faster).
- ➤ close result set
- ➤ close statement.

Some environment information:

- ➤ Test Machine: CAESS5 Solaris 2.7 Ultra-4
- ➤ Database: Oracle 8i version 8.1.6.0.0
- ➤ Perl - Version 5.005_03 with DBI 1.13, DBD-Oracle 1.03.
- ➤ Java/javac - Solaris VM (build Solaris_JDK_1.2.1_03a, native threads, sunwjit), and JDBC OCI driver.

Lessons Learned

Many lessons were learned in developing and deploying Web applications (regardless of using FastCGI). Many are just basic common sense approaches for any application. Web development does not magically make sound engineering practices obsolete.

Know Your Customer

You are designing the system to be used by your customer. Here are some questions you should ask yourself: Can your customer handle Java? What can they afford? What do they already have in place (databases, Web servers, browsers . . .)? What types of people will be using the system? What networks are in place? Is it an Extranet, Intranet, or Internet application? Who are the major players? Most importantly, have you involved the users in your development process, especially requirements and screen designs?

Is Your Data Available?

Worried about your data? Network Appliance®, an industry leader in enterprise data storage, has the solution. NetApp® storage appliances (filers) provide dedicated database storage, delivering unparalleled reliability, availability and scalability. Our patented Snap Technology replaces any data availability doubt with quiet confidence.

- Snapshot™ — Safely back up your database in seconds
- SnapRestore™ — Full data recovery in minutes, not hours
- Scale your database seamlessly — with zero downtime

Network Appliance. The reliable, simple, and cost-effective data management solution.

Call 1-800-443-4537, or visit our website at www.netapp.com.

Gather Requirements Up-Front

You've built your application, but the customer complains it was not what they wanted. A requirements document between the customer and developer agreed to up-front gives you the necessary documentation to back up your application. Prototyping is a good way to get started, but at some point you must have a production system that can map back to real requirements.

Design Carefully

The web provides only an interface to your application. Proper database design is the key to making your applications perform reasonably. Take a look at the global picture and envision how other future applications may fit within your design.

Keep as much of the processing at the database level through database objects such as rules, triggers, procedures, and functions. This increases re-use, security, and performance; this is true for both client-server and web applications. If possible, make it database-independent to provide options for your customer. For transactions, try to have procedures/functions call other procedures because CGI processes will autocommit each SQL transaction. If the procedure/function fails, you can rollback the transaction.

Watch Out for JavaScript

JavaScript provides excellent client-side processing to relieve the burden placed on the web server. While there are many neat tricks that can be accomplished, try to keep things simple since we have learned the hard way that JavaScript may not work the same (or work at all) on different platforms. With UNIX, things work generally as planned, but watch out for all the different PC platforms. What's worse, error messages are not always displayed, so one has to test extensively on various platforms to make sure things work for all users.

Development and Production Machines

A development team needs to continue to refine existing applications and develop new ones without affecting the current users. We have dedicated production web servers in place to make sure development does not affect the production application. These production machines can be distributed to handle projects that are located at specific sites to reduce network traffic and increase performance. While one could probably handle many projects with one web Server machine, performance will degrade if many users are accessing the same machine at the same time. The nice thing about the web is that most users are only using the server machine for about 10 seconds at a time. Most of the processing is done at the client side: typing in information (HTML forms) and validating it (with JavaScript).

PC servers are cheap and provide excellent performance when the application is small, but may lack scalability for more transaction throughput and better performance as your database and user base grows. We have utilized UNIX workstations with sufficient memory and processing to achieve peak performance. Also, PC NT machines are notorious for many security problems.

Standardize

We have standardized on Netscape as our browser of choice. This eliminates the need for testing on browsers such as Microsoft Explorer and Mosaic. It also decreases the chances that users will encounter problems that were outside the development environment. Design and coding standards should also be employed for maintenance and reuse. The Perl community has already recognized this with many freely available Perl routines complete with documentation on how to use them (see http://www.perl.com).

Be Secure

Security is a critical factor in a web application design. Databases have been providing various levels of security for many years. An infrastructure should also be developed to deploy various web components within the Intranet, Extranet, and Internet. RES has accomplished this with its internal on-going Web Contractor Integrated Technical Information System (WebCITIS) that is currently being utilized on many government projects. Obviously, because of the nature of the data being transmitted, security is of extreme importance. One needs to ensure that only authorized users can access the databases/documents via the Web and that the data cannot be intercepted as it travels across the Internet.

We suggest that you get away from database accounts and replace them with web server accounts so you need not generate a user ID and password for every user who wishes to connect to the database. We capture the user information from the user's web login and password that they must provide to make sure they have access to a

Quest Software

offers comprehensive Oracle solutions that ***accelerate*** development, accurately **deploy** changes, and **ensure** the reliability and availability of your production databases.

QUEST SOFTWARE

For more information on other Quest Software solutions, call 949.754.8000 or visit us at www.quest.com today.

given project's data. We then log into the database with a single account and store roles and other permission information in database tables by making the web login one of our primary keys.

Stand-alone or Network?

The answer is that your application should be designed for both modes. Ideally, your system is available to everyone on a series of networks. However, some applications will need to be run on secure areas in a stand-alone mode. The databases, browsers, web servers, and applications should be designed so that they can easily run within a small Intranet without the cost and performance penalties a large infrastructure can impose on them; even if this happens to be in one room with one or more computers! Conversely, a secure infrastructure facilitates developing and deploying applications to the Intranet, Extranet, and Internet.

Don't Forget Training

While a web-based tool provides users with an interface they are accustomed to, training is still essential so that they will get the most out of the application. A class also introduces the application to the users so that they are not forced to use it blindly from day one, and the class provides a forum for users to express their concerns. Training classes can be conducted and revised throughout the life of the application.

We are currently putting together a multimedia training class on using the Software Trouble Report system. Training resources are not always easy to find, and one can tire of teaching the same things over and over. This issue can be resolved by providing multimedia training that is available either on the Web or can be sent to run on a CDROM. An organization will need to make sure that user's machines and networks can handle full multimedia (voice, video, etc.). Initial results of the prototype at the time this paper was written provide some very promising progress.

Documentation Is Still Important

Software engineers are in great demand. Chances are that developers may not be with your company in the long run, especially if they have experience with web application development and integration of databases. Sure, the web makes applications easy to use; yet someone will have to maintain it for years to come. A requirements specification provides baseline requirements for your system; a user manual is the paper interface to the user community; and a maintenance manual provides details on how to maintain, build, and deploy your application. These documents can also be integrated into your system on-line to provide context-sensitive help. We store user documentation that is accessible in Adobe PDF format so that users can view, search, and print help information.

Freely Available Products Are Sometimes The Best Choice

Many companies prefer to go with COTS products because they are supported and widely available. However, as in the case of Perl, we have found "Perl Zealots" out there who are constantly providing free support (via newsgroups, email, and phone). Updates and new libraries are coming out daily with very innovative approaches.

Build For and With Reuse

Is this the only application your organization will be developing? Take advantage of what is already out there by incorporating various libraries available on the Intranet (many are free). We found many excellent JavaScript, Perl, and Java libraries that we incorporated directly into our application (and sometimes improved upon). We have also begun seeding our software reuse repository (also an Intranet application) with these routines and libraries. This approach makes rapid development and deployment of future applications feasible.

Test with a Broad Range of Users

Your system will probably work well for a majority of the high-tech users with elaborate engineering degrees. However, in many instances non-technical people, like some of your customers, managers, and others such as secretaries and engineering aides, will also be using your application on a day-to-day basis. It is essential that you get their feedback to make sure the system is understandable and useable by the entire user community. This will increase the chances that your system will actually be utilized.

Allow for User Feedback

We have used software trouble reporting systems to allow users to submit problems and suggested improvements on both internal and COTS applications. The reports are prioritized through software review boards that are attended by users and developers, and the results are incorporated into future releases. A web-based version of a software trouble reporting system is important because it provides enterprise-wide access to all users. Email can be sent to authors and key personnel whenever trouble reports are submitted, updated, and completed. Most applications also have a comments button on the home page to provide email feedback.

Ease Into Production with a Transition Plan

A transition plan provides a high-level plan to integrate the application into the enterprise. A good tool without a viable transition plan is doomed to fail, while a fair tool with a good transition plan has a chance to succeed. If you have many legacy systems, plan to spend lots of time converting to your new system. Be aware that all data may not map nicely so you will need to negotiate with your projects on how it is mapped. This may have to be done on a project-by-project basis.

Support Your Application

The application will need to be supported after it's released. This can be done through email, phone, or hands-on support. FAQ's provide answers to common questions posed by users. Make sure that they are up to date. Also make sure that support groups are in place (for applications, networks, and systems) and requests are responded to in a timely manner. The last point we cannot emphasize enough, users will support your system and continue to use it if you can solve their problems like it was your problem.

Ideally, a centralized help desk allows problems to be handled and logged quickly and reported to key personnel to resolve. Users like being able to talk to real people to feel that their problems are being taken seriously. It's difficult for developers who are under pressure to maintain and develop new applications to be tied up with support. Engineering Aides can be trained to handle many issues, which is important given today's scarce software resources.

One should also verify throughout the day that the application is up and running. Processes can be written to verify if the database or web server is down and alert the proper authorities. web server logs should also be checked to analyze usage and potential problems. Production machines should also be checked on a daily basis to determine the load and to check for run-away processes.

Finally, your application can do more than simply flash an error message to the user. It should also log all errors to a file and/or send email to supporters. FastCGI applications provide a catch utility to find all kinds of errors: web, database, server, etc. It's up to the application, not the user, to make sure they get reported and dealt with a reusable database error checking system.

Summary

The web has changed the way we think about developing and deploying applications. Some of the basic lessons learned in the client-server world apply directly to the web. Whether we like it or not, the preferred user interface today is the web browser within our heterogeneous platform environments. Our experiences have shown that users are familiar with and like the navigational features of the Web. Easy to use applications offer the potential to reach a large group of users over the enterprise, whether it be an Intranet, Extranet, or Internet. The use of Perl and FastCGI makes your transition to these technologies much easier and we feel it is a viable alternative to the use of Java.

Bibliography

[Blue96] "Sapphire Web: Frequently Asked Questions," **http://www.bluestone.com/products/sapphire/saweb_faq.html**, Bluestone, Inc., 1996.

[DBI97] "DBI - The Database Interface for Perl 5," Descartes and Bunce, The Perl Journal, Issue 5, Vol. 2., No. 1), Spring 1997.

[Brown1] "Understanding FastCGI Application Performance," Web White Paper, **http://www.fastcgi.com**, OpenMarket, Inc., 10-Jun-1996.

[Brown2] "FastCGI: A High-Performance Web Server Interface," Web White Paper, **http://www.fastcgi.com**, OpenMarket, Inc., April 1996.

[Net96] "NetDynamics: Delivering Applications in Web Days instead of Man Years," Web White Paper, **http://www.w3spider.com/ndwhite.html**, Spider Technologies, 1996.

[O'Reilly97] "The Importance of Perl," **http://www.perl.com**, Web White Paper, Tim O'Reilly and Ben Smith, 1997. ▲

This article is reprinted with permission from the IOUG-A Live! Conference, April 2001

Business today is performance, but …

Performance doesn't just happen.
Make it happen.

And make it *Precise*.

For more information please contact
Precise Corporate – 800-310-4777 x374

Many Thanks to Our Sponsors

NoCOUG would like to acknowledge and thank our generous sponsors for their contributions. Without this sponsorship, it would not be possible to present regular events while offering low cost membership dues. If your company is able to offer sponsorship at any level, please contact NoCOUG President Vilin Roufchaie at 925-487-9588 or write to board@nocoug.org.

Long-term full event sponsorship:

LOCKHEED MARTIN

CHEVRON

Long-term supplemental event sponsorship:

ORACLE

Thank you! Year 2001 Gold Level Support Vendors:

- BMC Software
- Cast Software, Inc.
- Database Specialists, Inc.
- DB Doctor
- Embarcadero Technologies
- Informatica
- Lecco Technology, Inc.
- Network Appliance
- Precise Software Solutions
- Quest Software, Inc.
- Seek Systems
- Veritas Software Corporation
- XIOtech Corporation

For information about our Gold Level Vendor Program, Contact Nora Rosingana at 925/820-1589 or via email at noraros@pacbell.net

TREASURER'S REPORT

Judy Lyman, *Treasurer*

Beginning Balance		
July 1, 2001		**$ 52,479.54**
Revenue		
Advertising	-	
Sponsorships	-	
Membership Dues	2,370.00	
Meeting Fees	1,720.00	
Vendor Receipts	13,500.00	
Interest	113.25	
Tax Refund	3,200.00	
Total Revenue		**$ 20,903.25**
Expenses		
Regional Meeting	5,286.95	
Journal	5,207.24	
Membership	542.15	
Administration	830.00	
Income Taxes	838.67	
Board Meeting	500.00	
Web Site	75.00	
Miscellaneous	-	
Total Expenses		**$ 13,280.01**
Ending Balance		
September 30, 2001		**$ 60,102.78**

Things Somebody Should Have Explained Better

By Tim Quinlan, TLQ Consulting Inc.

Introduction

This paper covers many diverse areas of Oracle DBA activities. It covers some situations that I wish I'd known more about before I had to deal with them under the line of fire. This could have been called "things I wish I hadn't learned in production." Some of the issues discussed in this paper are:

- Rollback Segment Issues and Tuning
- Cursors and Commits
- Encountering Problems and Shutting Down the Database
- Dealing With Very Large Exports
- The Advantage of Having Multiple Backup Techniques
- The Value of RowID's
- Hashing: How exactly does this work?
- User Passwords to Change After Install
- The Importance of Getting Disk Space
- Tablespace Creation Tips
- Coalescing at 1%
- Shrinking Datafiles (alter resize)
- Minimizing Databases
- Problems With Triggers
- Do it in the Database

Let's get right into it.

Rollback Segment Issues and Tuning

Here I will cover some important, quick topics that will help you understand some key concepts.

What Rollback Segments are used for:
For Rollback, Recovery and Savepoints

Changes to tables through DML (insert, update or delete) statements generate redo logs in the form of "after" images that are written out to the Oracle log. The "before" images of these records are stored as "undo" data in rollback segments (RBS). These RBSs are required for backup and restore operations as well as for transaction rollback, recovery and savepoints. The rollback segment tablespaces are themselves managed like any other tablespace by Oracle and must be logged as redo records in the Oracle log. So, in Oracle recovery, the Oracle logs are read and all complete (committed) transactions are applied and the RBSs are recreated. The RBSs are then read and the "undo" records are used to rollback uncommitted transactions.

For Consistent Querying

When one transaction (transaction "A") is updating a table and then a second query transaction (transaction "B") starts accessing the same table, the rollback segment is used to provide a consistent "view" of the data. This allows transaction "B" to access data as it looked at the start of the query, since it is using "undo" (or before) records that match the transaction's start time.

How Many and What Size?

Create at least one non-system rollback segment. If your system's disk space is limited and you have very few concurrent transactions (for instance less than four), then one non-system RBS will be optimal. This is due to the fact that two will require you to cut your entire RBS space in half. In those cases where you have ample space, then multiple RBSs will protect the more efficient transactions from being impacted by one poorly designed (the large update with no commits) transaction.

A complete transaction entry must be completely contained in one rollback segment. An RBS extent can be used by many transactions. Also, the database assigns a transaction to a rollback segment in a round-robin fashion, unless the transaction names a rollback segment to use with the "set transaction use rollback segment segment_name" statement. Due to this, it makes sense to create rollback segments of the same size with many equally-sized extents.

Most rollback segments should conform to a size that matches the typical transaction with only a couple of special RBSs sized to deal with atypical transactions. Long-running transactions perform better with larger RBSs because the rollback entries can fit in pre-allocated extents.

Rollback segments should be placed in a separate tablespace.

Sizing Rollback Segments

Some things to consider when sizing rollback segments are:

- Make sure that transactions do not wrap around the rollback segment too quickly.
- Test your rollback segment sizes by starting with small rollback segments and forcing the application to extend them.
- When sizing your rollback segments, a rule-of-thumb is to try to size them with an extent size that will be somewhere between 10 and 30 extents. This number may seem to be quite arbitrary but the reasons behind it are:

- A rollback segment that is sized to have very few extents (say, 1-3) will likely be sized very large and waste a lot of space. When it extends, the next extents will also be very large and likely be larger than what is really needed.
- A rollback segment that is sized with an extent size that is too small will extend and contract too frequently and will incur a great deal of overhead.
- A rollback segment that increases between 10 and 30 times, will allocate a new extent that will not be grossly overused and it will allocate new extents at a reasonable rate.

This information came from Metalink note 62005.1.

Three Scenarios of Different System Types

1. Steady Transaction Rate and Size
   For systems with a steady transaction rate, do the following:
   - Make all extents the same size.
   - Allocate extra space in the tablespace to allow segments to grow.
   - Make OPTIMAL large enough to cover the minimal coverage size.
2. Frequent Large Transactions
   Systems that regularly have large transactions are difficult to tune. If you are faced with this situation, consider the following:
   - OPTIMAL is not a great option since you cannot count on a segment shrinking in time for other large transactions.
   - You have two options:
     - Reduce the segments so they are all large enough to hold all of the transactions.
     - Build some (one or more) large rollback and have the large transactions use the 'set transaction use rollback segment' syntax.
3. Infrequent Large Transactions
   When your system has a mixture of small transactions and some infrequent large transactions, then:
   - Leave freespace in the RBS tablespace to fit the largest transaction rollback data into it.
   - Take advantage of the OPTIMAL parameter. Make it large enough to handle at least the minimal coverage and allow it to grow to fit the largest transaction.

Snapshot Too Old

The infamous ORA-1555 error is caused by one simple thing: the use of inactive data by an active transaction. When an updating transaction is complete (commits), the rollback segment entries for that transaction are marked as inactive. Once this occurs, another transaction is free to reuse the blocks that were used by the completing transaction. A query can continue to read the blocks as long as they remain in the RBS. Once the segments either go "away" or are overwritten, an ORA-1555 may be issued on

There's no substitute for experience.

At Database Specialists, we know the internals of Oracle and we can help you harness its power. Count on our experience for:

▲ Database architecture, design and review

▲ Supporting mission-critical applications

▲ Performance tuning

▲ Upgrades and migrations

▲ Backup and recovery strategies

▲ Capacity planning

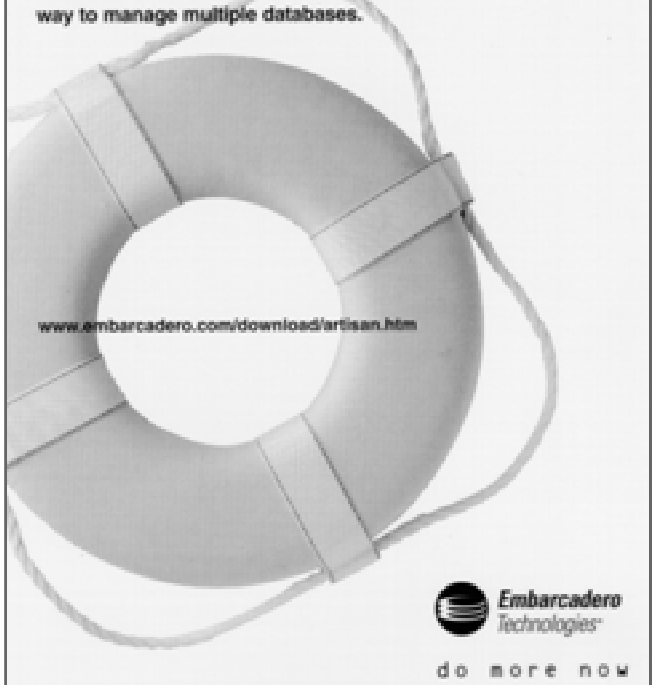
Experience.

Database Specialists, Inc.

ORACLE CERTIFIED SOLUTION PARTNER

Dive into any database administration challenge. Without the risk.

DBArtisan. The fastest, easiest—and safest—way to manage multiple databases.



www.embarcadero.com/download/artisan.htm

Embarcadero Technologies

do more now

©2000 Embarcadero Technologies, Inc. All rights reserved. Embarcadero Technologies and DBArtisan are registered trademarks of Embarcadero Technologies, Inc. All other products are the trademarks of their respective companies.

the querying transaction.

The use of the "optimal" parameter may cause RBS blocks to "go away" by decreasing the RBS to the optimal size by removing those RBS blocks that had been extended by a running transaction that is not complete. Also, many small RBS extents are more likely to be re-claimed than a few large extents. Therefore, it is important to minimize long-running queries at times when updating transactions are being performed.

ORA-01555's can be caused by the following situations (from Metalink 1005107.6 and 45895.1 and 40689.1):

- ➤ Not enough rollback segments and those that do exist have small sizes in an active transaction database. This is relatively easy to fix. For OLTP, have a relatively large number of rollback segments and a decent extent size and with batch use a larger rollback segment.
- ➤ A long query takes place at the same time as an update. Potential solutions include:
  - Add more and/or larger RBSs. When adding more RBSs consider the Tablespace datafile sizes to ensure that RBS extensions will fit in their datafiles. Consider spreading RBSs across tablespaces to better control space.
  - Do not run long-running queries with OLTP transactions.
  - Consider removing the optimal size from the RBSs in question. This will hold committed values in the RBS as long as possible.
  - Delay commits so that RBSs will remain active longer and cannot be reused.
  - Write shorter queries.
  - Use read-only snapshots if they can be created efficiently (quickly without taking up too much space).
- ➤ Read consistent view during "insert." When performing a full-table scan, Oracle will read to the high water mark (HWM) as specified at the start of the query. If rows are inserted above the HWM by another transaction, then an ORA-01555 will not occur. If rows are inserted below the high water mark then they may. Some potential solutions are:
  - Make loads parallel to minimize the time a table is unavailable to query.
  - Do not pre-allocate extents or pre-allocate only as much as needed when they are needed.
  - Use truncate rather than "delete * from ..." to flatten HWM and remove gaps.
  - Use SQL*Loader direct loads to write data above the HWM.
- ➤ A corrupted rollback segment prevents a consistent read. Try to drop and recreate the rollback segment with the database up and running. These types of errors are rare and will likely result in a call to Oracle Support.
- ➤ An open cursor is used across commits and fetches are performed across commits. This may be remedied by committing less often (yes, this is not a typo).
- ➤ A fetch was performed across commits with delayed block cleanout. This occurs because the data block was not updated with the last committed image until the next reader accesses the data block, which requires verification from the transaction table of the rollback segment. This is a very complex issue caused when all of the following occur:
  - An update is performed and committed and the blocks are not touched again.
  - A long running query begins on previously updated blocks (it's using RBS).
  - A lot of DML takes place (not on the previously updated block) and the RBS wraps around. The system commit number (SCN) of the first update is cycled out of the RBS.
  - The lowest SCN in the RBS is higher than the read consistent SCN in the query.
  - The query learns that the updated block was committed but not cleaned out, so it cleans out the block. But since the SCN was written over in the RBS, it cannot determine if the block was updated and committed before the long running query began. The ORA-01555 is the result.
- ➤ This can be resolved by the following:
  - Consider using OPS and partition DML across instances to avoid clobbering RBSs.
  - Using "set transaction use rollback segment."
  - More and larger RBSs so information is not overwritten.
  - A solution to this provided by Metalink was to perform a full table scan on the involved tables to perform the delayed block cleanout before handling the cursor.
- ➤ Oracle allows cursors to remain open across commits. This is not an ANSI standard (ANSI required the "with hold" option to do this properly), so programmers using this should do so infrequently and should be aware of the risks. Some ways to alleviate ORA-01555's are to: reduce commits, run the program standalone (with no other users on the system) and remove the optimal clause from the RBS being used.

Temporary Tables and Rollback Segments

Temporary tables generate no redo logs in the form of 'after' images that can be re-applied. But, there is transaction control so the transaction can rollback in the form of undo records. As mentioned above, these undo records for rollback are stored in rollback segments. RBSs are needed for transaction recovery, rollback and savepoints. Undo data is stored in RBSs and the

RBS tablespace is treated like any other tablespace and logged as redo records in Oracle logs. So, even though there are not standard redo logs written for DML operations on temporary tablespaces, there are redo records written for the RBS. This will amount to about 50% of the size of regular (non-temporary tablespace). If inserts are done rather than updates or deletes, this percentage savings becomes larger.

Rollback Segments not in init.ora but Online

It is possible to have more RBSs online than exist in the init.ora parms. This is confusing but occurs because the ceil (Transactions/ Transactions_per_rollback_segment) is higher than the entry for rollback_segments, the system will set other "public" owned RBS's online until there are none left or the calculation is met. If none are available, Oracle uses what it has. In cases where Transaction is derived from Sessions that is derived from Processes, then Transactions can be set quite high.

No Logging and Rollback Segments

Although no logging may be specified for a table, RBS entries are still created. Where is that NoRBS option?

Inserts and Rollback Segments

You may think that Insert Statements would generate no RBS entries since there is no undo information generated. Unfortunately, this is not true. Insert statements to generate some undo information in RBSs but use less space than an Update and Delete would use. For example, an Insert of 165000 rows used 7MB of RBS space generating 14 500K extents. The corresponding Delete used 30Mbytes across 60 500K extents. This was for a table with 5 columns and a rowlength that is approximately 100 bytes in length.

Cursors and Commits

When opening a cursor for update, performing a fetch, a commit and then looping to the top of the loop to fetch the next row, you will receive an "ORA-01002 fetch out of sequence" error. The reason for this is that a commit releases all locks, therefore the cursor that was open with commit option is now invalidated and closed. Oracle does not properly support open cursors across commit points, though they can work and are often used. The ANSI SQL approach to this is to use an "open cursor . . . with hold" or to close and re-open the cursor after a commit has occurred.

The solution to this particular ORA-01002 is to remove the "for update" clause and to check the columns at Update time to ensure that they have not been changed by another transaction since they were initially read.

Encountering Problems and Shutting Down the Database

If you are noticing a problem in your alert logs, do not immediately shutdown your database, unless you are sure that it will restart. Before shutting down, make sure that you understand the problem that is occurring. If you do not, then contact Oracle support services and/or Metalink. There may be information that you need to get while the database is up. You may even be able to correct the problem while the database is up and running – or do things that will make it easier to recover (like perform a database export). Shutting the database down too early may result in your not being able to restart the database easily (some rollback segment corruptions may not allow Oracle to

LECCO
SQLExpert
SQL with a Higher IQ

LECCO SQL Expert automates the task of performance tuning and optimization of database applications by providing developers and DBAs with expert knowledge of SQL. Through its proprietary Feedback Searching engine, it provides optimum SQL statements–guaranteed to provide the correct results. No guesswork or hopeful suggestions, statements are actually rewritten.

- Oracle
- Sybase
- MS SQL Server

Join the ranks of the experts and download a FREE evaluation copy today!

Where the experts go for performance.

LECCOTECH
www.leccotech.com

WHEN YOUR STORAGE SYSTEM PERFORMS WELL, SO DO YOU

FasFile RAID from Seek Systems accelerates the performance of database applications without modifications to your existing infrastructure.

Call us...
We can help you!

You can download our technical paper on how to improve your Oracle database application performance at:

http://seeksystems.com/products/adaptive.html#downloads

OR . . . Call Seek Systems Region Manager, Chris Fleharty, at 866-733-5797. We'd be happy to mail you a copy, or answer any questions you may have.

Seek Systems

www.seeksystems.com
1-800-790-7335

restart without intervention) and thus digging a larger hole for yourself.

Dealing With Very Large Exports

A long, long time ago, (version 8.0) Oracle did not properly support exports larger than 2 Gbytes. To be more correct, the exports worked fine, but the imports did not always work cleanly. When importing under 8.0.5 an export that was > 4Gbytes, I would be able to create tables and populate data (the most important part), but SQL such as Grants at the end of the export file would not be properly executed. A solution recommended from Oracle was to do the following 3 steps:

1. Do a full import with SHOW=Y and redirect the output to a log file with the following command: impsystem/manager file=a.dmp full=y log=a.log SHOW=Y
2. Perform a full import with GRANTS=NO using syntax: imp system/manager file=a.dmp full=y GRANTS=NO
3. Edit the log file a.log to extract all the GRANT statements and run these GRANTS through SQL.

This worked but was not that easy since I had to keep changing the Oracle Connection to perform the Grants. This was performed through an awk and shell script.

Fortunately, Oracle 8i there is a solution to this. Exports can be split across datafiles using the filesize parameter as shown below:

```
exp system/manager full=Y compress=Y
consistent=Y file=P068exp1.out, P068exp2.out
filesize=2000M log=%ora_lly_backup%\P068exp.log
```

In the above example, each file will grow to 2Gbytes (2000 Mbytes) and at that point, the export will continue to the next file that is then created. These can be used to import the data – a simple solution to some of those complex exports we've had to build in the past.

The Advantage of Having Multiple Backup Techniques

Be paranoid about backing up data – and being able to recover it. This will help you live those slogans that you hear . . . "being able to sleep at night" and "looking like a hero". To do this, consider the following:

- ➤ Find out the business requirements to recover data. Sorry to insult you with this tip, but it is the starting point to this. Is the business requirement truly 7*24*365? Is it 5*9 or do you have 4 hours to recover data? There are many systems that can still go back to a copy of the data from the night before.
- ➤ Test your backup and restore.
- ➤ Implement as many backup techniques as you can. These should not impact the system's performance requirements and should provide different purposes. Examples of this are:
  - Cold or hot backup as well as a full export
  - RMAN with user exports
- ➤ Along with the above, test restoring your database as follows:
  - Point-in-time recovery
  - Recovery from a cold backup
  - Recover a single table using an exported table.
  - Recover a set of tables using a cold/hot or RMAN backup and create a clone database.
- ➤ Test these often to stay sharp
- ➤ Use RMAN: it's simple, powerful and manages files efficiently.

I've been in many situations where I was happy to have full (hot/cold) database backups as well as exports.

The Value of RowIDs

ROWIDs can be used in many ways to solve problems and improve system performance. These are the quickest ways to access a row. Even direct index access incurs the overhead of having to traverse an index tree before accessing data. Two classic ways that ROWIDs can be used are shown below.

1. Using ROWIDs to remove duplicates
   The fastest way to access a row in Oracle is to use a feature called a ROWID. This is an internal identifier that can be thought of as a pseudo-column. It never actually appears as a column if you perform a describe on a table, or if you query the "column" catalog table all_tab_columns. Despite this, you can retrieve the ROWID when selecting a row from a table. An example of this is:

```
Select column1, column2, rowid
from table_name
where table_key = id1;
```

The ROWID that is returned above can later be reused to select, update or delete the same row. It is a direct pointer to a given row in a block and a file for an object.

Why ROWIDs Are So Fast

To understand why a ROWID can be used to access a row so quickly, all we need to do is decipher a ROWID. An Oracle ROWID needs to be deciphered using Oracle built-in functions. Without these functions, the ROWID is unintelligible (unless the ROWID is in the older, restricted format). To get the row number, block number and file number, use the DBMS_ROWID package as shown below:

```
Select DBMS_ROWID.ROWID_BLOCK_NUMBER(rowid),
DBMS_ROWID.ROWID_TO_ABSOLUTE_FNO(rowid),
DBMS_ROWID.ROWID_ROW_NUMBER(rowid),
from table_name
where table_key = id1;
```

This will return information similar to the following:

```
BLOCK FILE ROW
321 52 4
```

The ROWID has built into it the exact location of a row.

2. ROWIDs to Improve Updates and Deletes
   In some cases, you may have a cursor or a select statement that later results in a delete or an update of the retrieved row. These statements can use a few approaches to update or delete. They can use "where current of cursor", they can use the Primary Key (or Unique ID) or can use a ROWID. In a case where "current of cursor" cannot be used, use a ROWID that was previously retrieved from as part of the Select statement.

Hashing: How Exactly Does This Work?

The basic premise to a hash-join is that an in-memory hash-table can be built from the smaller of two tables being joined. This smaller table becomes the "build-input." The larger table is used to probe the hash table. This is the "probe-input" and it does not need to fit in memory to perform well.

A simple explanation of hash-joins using the simplest case of hash-join:

- When the build-input fits into memory specified by hash_area_size then an in-memory hash table is built and the probe-input is read and used to probe the input. The probe-input is read and joined to the build-input in memory and if the join is successful, the row is immediately output. A bitmap is also created which sets a flag stating whether a particular hash-bucket contains values.

The bitmap can best be explained with a simple example. Consider two input tables that are being joined that have the following join keys:

```
Table X: {0,0,2,2,2,5,9,12}
Table Y: {0,1,5,22}
```

Table X becomes the probe-input and Table Y becomes the build-input. Using a Mod 10 hashing algorithm, the buckets look as in Figure 1.

In Figure 1, join values are hashed to buckets using mod-10. Therefore, value 22 from Table Y will hash to bucket Y2. The bit-vector for Table Y (the smaller build-input) will have "on" values for buckets Y0, Y2, and Y5. When value 12 is read from Table X, the bitmap value for the corresponding bucket for the build-input is checked. Since this is set to on, the other row is read. The join in this case is unsuccessful since 12 is not equal to 22. Now let's look at the case of value 9 in bucket X9 from Table X. The bitmap is checked and returns a false value. Therefore bucket Y9 is not even read. This shows the value of the bitmap to process hash-joins.

Here's a more complex explanation of the hash-join. What if the build-input (the smaller table) does not fit into the hash-area in memory? In this case, the following steps are performed.

- Process the Build-Input
  - Determine the number of hash partitions to use based on hash_area_size, db_block_size and hash_multiblock_io_count.
  - The smallest row is read (called the build-input) and each row goes through a hashing algorithm. The hashed value is then placed in the appropriate partition. A bitmap of the hashed values is built using the hash value. A separate hash value is calculated using another algorithm and stored with the row in case a rehash is later needed. A bitmap of this second hash value is also created.
  - If the hash area becomes full, the largest partition is written to disk. Any new rows that hash to this partition must be written to disk. The above steps continue until the build-input is exhausted.
- It's now time to process the probe-input as well as the above created hash table.
  - Read the larger row source, the probe-input, hash each row and see if it hits a partition that is in memory. If so, then return a joined row. If not, write the row to a new partition using the same hashing algorithm as used to build the previous hash-table. This partition is stored on disk.

Figure 1.

PARTITION		X0	X1	X2	X3	X4	X5	X6	X7	X8	X9
	VALUES	0,0		2,2,2,12			5				9
Y0	0	TRUE									
Y1	1										
Y2	22			TRUE							
Y3											
Y4											
Y5	5						TRUE				
Y6											
Y7											
Y8											
Y9											

Continue this until the entire probe-input has been processed.

- If partitions and rows from both inputs still exist on disk, load the smallest partition from either input set and build an in-memory hash table using the second hashing algorithm values (to distribute rows more evenly).
- Choose the corresponding partition from the other table and begin the join (as above). Continue this until the rows are exhausted.

When using hash-joins, make sure your hash_area_size is set correctly. When estimating this, allow for about 20% overhead for areas such as bitmap vectors.

Sources for the above are: Metalink note 41954.1

User Passwords to Change After Install

There are many userid's and passwords to change after Oracle products have been installed. Of course the classic ones are:

```
SYS/change_on_install (userid=SYS, password=change_on_install
SYSTEM/manager
OUTLN/outln
TRACESVR/trace
AURORA$ORB$UNAUTHENTICATED
ORDSYS/ordsys
ORDPLUGINS/ordplugins
MDSYS/mdsys this is used for spatial objects (metadata)
CTXSYS/ctxsys for intermedia
IFSSYS/ifssys for the IFS
DBSNMP/dbsnmp
Owner1/owner1 for Oracle designer
```

These are only some userid/password combinations where the password should be changed. The point is that every time you install Oracle software, investigate the default passwords and change them.

In ORACLE_HOME/rdbms/admin grep all files for keywords such as "identified".

```
        Grep identified *
        Grep IDENTIFIED *
```

Also, check table dba_users for all usernames to see if any appear to be Oracle userid's. Then check the install information for the product involved to find the corresponding default password. Another important point to note is that many of these have considerable authority in Oracle.

The Importance of Getting Disk Space

There is nothing worse than not having enough disk space in your environment. There is nothing worse than wasting valuable time because you are trying to find some space to perform an operation. When you fall back on the axiom that disk space is cheap and that you should be able to get some more quickly you find out a few thinks very quickly:

- ➤ Disks become obsolete and incompatible with new technology very quickly.
- ➤ You require new racks, or flow space.
- ➤ New drives are not so cheap after all.

When sizing the data needed by your system, determine the following:

- ➤ The system's Oracle data size
- ➤ Index size
- ➤ Requirements of load files
- ➤ Requirements of sort space and system space in Oracle
- ➤ The size of sort space outside of Oracle
- ➤ Work space required outside of Oracle
- ➤ Work space in Oracle
- ➤ Margin of error
- ➤ Online backup space

In other words, once you have estimated your Oracle disk space requirements for an environment (say for dev or test or prod), do not stop there. It is reasonable to multiply these estimates by a factor of eight (sometimes even more). You will be happy to have this space up front, and you will use it.

Tablespace Creation Tips

When determining which tablespaces to create for a system, there are many approaches that you can take. A couple of obvious things to consider are:

- ➤ Separate the tablespaces used for data and indexes.
- ➤ Consider the backup and recovery requirements of your system. Placing tables that must be restored together in the same tablespace can simplify backup.
- ➤ Estimate the best extent size for tables and consider grouping these together.

Some people like to put all data in one tablespace and all indexes in another; others take the opposite approach and place one table/index per tablespace. The approach we have been using involves placing tables in one of four tablespaces called:

```
[sys]_Small_data_01, [sys]_Medium_data_01,
[sys]_Large_data_01, [sys]_Huge_data_01
[sys]_Small_index_01, [sys]_Medium_index_01,
[sys]_Large_index_01, [system]_Huge_index_01
```

In the above, [sys] can be an application name, or any system name. The numbers can be used to add multiple tablespaces of the same size across devices for a given system.

Sample parameters for these tablespaces are:

```
CREATE TABLESPACE SYS_SMALL_DATA_01
DATAFILE '/r01/oradata/SID/SYS_SMALL_DATA_01_01.DBS' size 1000M
DEFAULT STORAGE ( INITIAL 96K NEXT 96K PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_MEDIUM_DATA_01
DATAFILE '/r01/oradata/SID/SYS_MEDIUM_DATA_01_01.DBS' size 1000M
DEFAULT STORAGE ( INITIAL 2M NEXT 2M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_LARGE_DATA_01
DATAFILE '/r01/oradata/SID/SYS_LARGE_DATA_01_01.DBS' size 2000M
DEFAULT STORAGE ( INITIAL 16M NEXT 16M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;
```

```
CREATE TABLESPACE SYS_LARGE_DATA_02
DATAFILE ‘/r02/oradata/SID/SYS_LARGE_DATA_01_01.DBS’ size 2000M
DEFAULT STORAGE ( INITIAL 16M NEXT 16M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_HUGE_DATA_01
DATAFILE ‘/r02/oradata/SID/SYS_HUGE_DATA_01_01.DBS’ size 2000M
DEFAULT STORAGE ( INITIAL 64M NEXT 64M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_SMALL_INDEX_01
DATAFILE ‘/r03/oradata/PRD_82/SYS_SMALL_INDEX_01_01.DBS’ size
1000M
DEFAULT STORAGE ( INITIAL 96K NEXT 96K PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_MEDIUM_INDEX_01
DATAFILE ‘/r03/oradata/PRD_82/SYS_MEDIUM_INDEX_01_01.DBS’ size
1000M
DEFAULT STORAGE ( INITIAL 2M NEXT 2M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_LARGE_INDEX_01
DATAFILE ‘/r04/oradata/PRD_82/SYS_LARGE_INDEX_01_01.DBS’ size
2000M
DEFAULT STORAGE ( INITIAL 16M NEXT 16M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;

CREATE TABLESPACE SYS_HUGE_INDEX_01.Paper #
DATAFILE ‘/r04/oradata/PRD_82/SYS_HUGE_INDEX_01_01.DBS’ size 2000M
DEFAULT STORAGE ( INITIAL 64M NEXT 64M PCTINCREASE 0
MINEXTENTS 1 MAXEXTENTS 1000
) ONLINE;
```

The maxextents and file sizes listed above are arbitrary values that you can set to suit your environment. The initial and next sizes can also be changed in your environment to suit your applications. The point here is to create a reasonable number of tablespaces that can be easily managed and understood and can provide you with the flexibility to implement simple backup/recovery and high-end performance.

Once these tablespaces have been created, use their default parameters for all segments created in them. In other words, do not create different sized extents for each table and index in the tablespace. This will simplify your data management and will take advantage of evenly sized extents in the tablespace. The segments do not need to fit into a single extent. The allocation of multiple extents for a segment is not a large performance issue (when compared to row-splits).

Coalescing at 1%

The Oracle SMON process will coalesce tablespace extents to combine adjacent chunks of freespace for any tablespace. This, however, only occurs if the Pctincrease setting of a tablespace is > 0%. Because of this, many DBAs set the pctincrease parm to 1% to force the coalesce.

This may seem like a bug, but it is not. If the Pctincrease is 0%, then every extent will have the same size. Therefore, there is no use in wasting time and resources in combining extents that will eventually all be the same size anyway. In other words, setting this to 1% from 0% is not necessary.

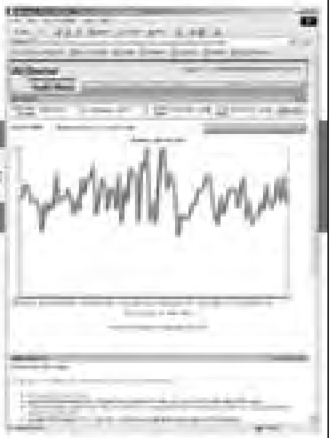
dbDoctor

"Health Watch" for Oracle

Has "Hatched" version 1.4

New Features, More Reports

Lower Price !



See for yourself, request an evaluation and ask for special "NoCOUG" prices!

db Doctor
Keeping Databases Healthy

http://www.dbDoctor.net (866) 323-6286

ORACLE TRAINING

DOESN'T HAVE TO COST AN ARM AND A LEG.

On-site, Customized Oracle Training at Not-So-New Economy Prices.

If your company has 2 or more people that require Oracle training, why come to us, when we'll come to you.

We've been providing affordable Oracle education globally for over 10 years. Isn't it time you cashed in?

eHORIZON

650-652-5977 www.ehorizonusa.com

Figure 2.

Select * from dba_extents where tablespace_name = 'MEDIUM_DATA_T01';

```
OWNER          SEGMENT_NAME          PARTITION_NAME        SEGMENT_TYPE        TABLESPACE_NAME
--------    ----------------      ----------------      -------------       ----------------
EXTENT_ID        FILE_ID         BLOCK_ID            BYTES         BLOCKS        RELATIVE_FNO
----------      ---------        ----------          -------------------         ---------------
TESTUSER       CUST_ADDR_TEST                        TABLE        MEDIUM_DATA_T01
       18              8                  462        245760         30                8
TESTUSER          PLAN_TABLE                                       TABLE            MEDIUM_DATA_T01
        0              8                 2902         81920         10                8
```

Shrinking Datafiles (alter resize)

When managing tablespaces, we have many ways to add space to a tablespace by adding datafiles or by extending datafiles. We now have a way to shrink a datafile with the "alter database datafile [file_name] resize [new size];" command. Consider the case of a datafile that is 50Mbytes in size that you want to shrink. You look at the dba_free_space table and feel that you can decrease this file to 20Mbytes. You issue this command:

```
Alter database datafile
'C:\ORACLE\ORADATA\TLQ000\MEDIUM_DATA_T01_01.DBS' resize 20M;
```

This fails with the following error message:

```
ORA-03297: file contains used data beyond requested RESIZE value
```

The resize failed, because there are extents in the range that you want to shrink. The next step is to interrogate the dba_extents table for this datafile. This provides us with the information provided as shown above in Figure 2.

With a db_block_size of 8192, the plan_table begins at block 2902 or at approximately byte 23773184 of the datafile. The following statement will fail.

The following statement however succeeds:

```
Alter database datafile
'C:\ORACLE\ORADATA\TLQ000\MEDIUM_DATA_T01_01.DBS' resize 25M;
Database altered.
```

You have trimmed this file size in half. In order to claim more space, you can move, drop or export/import the table to move the used extents to the front of the datafile and can resize this datafile even further.

Minimizing Databases

It may appear as though there is less involved in managing a few small databases than there is in managing a single large database. This may be true if you only have a few applications and databases. What if you have hundreds of databases? What is the impact of implementing many systems in one Oracle service rather than implementing each in its own Oracle database?

Consider the following points:

- ➤ The CPU, memory and disk overhead of each running Oracle Database.
- ➤ Startup and shutdown costs when dealing with tens or hundreds of instances.
- ➤ Management costs of upgrading, tuning, debugging and backing up many Oracle Databases.

Given the power of today's Oracle software, your goal should be to segment your systems by schema, tablespace and partition and to implement as many systems as possible in a single Database. The extra efforts required in implementing a large Database will generally be less that are required to implement one instance per application system.

Your goal should be to have one production Oracle Database for OLTP systems and one for Data Warehousing. You may not be able to achieve this due to peculiarities of systems that you purchase, but you should keep this target in mind. You will of course need Development and Test environments and of course a separate Lab environment (or sandbox) for the DBAs and technical specialists to try new features in.

Problems With Triggers

Triggers are very powerful and useful features in Oracle but need to be managed very carefully. It is absolutely critical to ensure that triggers remain active and valid. A change to a trigger object can invalidate a trigger without anyone being aware of the fact. This can quickly lead to data integrity errors that may not be detected until it is too late. Make sure that your triggers are all valid and active. Check them after performing any DDL changes.

Do it in the Database

What else is a DBA supposed to say? When given the choice of performing an operation outside of a database (in a file) or in a database, I almost always prefer doing it in the database. The ease of sorting and grouping data and operating on it with PL/SQL, breaking up complex operations into multiple steps with work tables that are temporary in nature and moving data around, makes operating in Oracle a preferred choice. Once the data is in Oracle, I like to keep it there and perform all of my processing in Oracle. This simplifies the toolset, the steps involved in data transformation and potential trouble points.

A lot of ETL approaches extract data from the database, perform many operations on the extracted files and then load the data back into the database. In cases where you are operating in the same Oracle database, keep the data in Oracle and perform your data manipulations there. You will have many more features to your fingertips in Oracle.

Conclusion

Hopefully some of this will help you avoid doing too much learning in a production environment under the line-of-fire. Thanks to Bonnie Baker whose series of DB2 presentations on "Things I Wish I'd Known 5 Years Ago" inspired this. ▲

This article is excerpted and reprinted with permission from the IOUG-A Live! Conference, April 2001.

Summer Conference Recap

The Board Members Get Around

Our NoCOUG Summer Conference, hosted by Chevron in San Ramon, was a great success. The August 23 meeting had over 220 attendees, and the program was outstanding. A big pat on the back goes to **Carol Hipp at Chevron** who handles all the facility arrangements for us and does an outstanding job. **Thank you, Carol!**

Following are some photo highlights from the meeting.

NoCOUG Membership Director Merrilee Nohr greets attendees at the registration table.

NoCOUG board members Darrin Swan (left) and Hamid Minoui (center) chat with another attendee.

NoCOUG Webmaster Roger Schrag (right) visits with featured DBA speaker Gaja Krishna Vaidyantha.

NoCOUG vice president Joel Rosingana (left) chatted with featured data warehousing speaker Michael Scofield.

RESOURCE CORNER

JL Computer Consultancy in the UK has provided "Folk Lore and Fairy Tales" about Oracle databases. There you'll see clever one-liners as well as extensive explanations helping you to debunk those nagging legends. See for yourself at: **http://www.jlcomp.demon.co.uk/myths.htm**l.

Want to try out some new tools? **OraPub** founder, Craig Shallahamer has published tools such as OSM Interactive, which provides "interactive information about an Oracle database system" and WebLoad, a synthetic load generator—just to name a few. Download these at: **http://www.orapub.com/cgi/genesis.cgi?p1=sub &p2=tools_main**.

Send your favorite resources to journal@nocoug.org for publication.▲

And the day ends with the reception at O'Kane's!

NoCOUG Fall Conference

Thursday, November 15, 2001

Session Descriptions

KEYNOTE

9:45-10:30
Jim Leonard, *Director, Genentech*

DBA TRACK

11:00-12:00
Protecting an Oracle Database
Aaron Newman,
Founder and CTO, Application Security, Inc.

Oracle databases are a core component of enterprise systems, and house some of an organization's most valuable assets. Yet the security of Oracle is often overlooked. There is little knowledge about database security leading to a large amount of security risk. Oracle databases are also becoming more and more accessible through the Internet. This leads to a situation where security becomes a requirement rather than a luxury. While Oracle's security features are rich, they are also complex and hard to implement. In this presentation, Aaron will address several of the most overlooked and unclear topics on Oracle security. He will also discuss how hackers are attacking databases and how to prevent this from happening to your database. *(Intermediate Level)*

1:15-2:15
Optimizing Your Oracle Database Through Storage Virtualization
Matt Miller,
Product Marketing Manager, Veritas

In the age of eBusiness and massive growth of data, databases have become mission critical applications. Storage virtualization, managing data from a logical rather than physical model, is essential to enable IT departments with tight budgets to meet the demands of growing data and tougher SLAs. Consistent management across multiple OS and storage platforms eliminates the need to learn multiple OS or array-based tools and can more efficiently use disk capacity. Performance and availability are also key considerations for Oracle environments in order to ensure application response times with databases growing exponentially. The ability to administer and manage Oracle online, through a file system, with the speed of raw partitions is no longer a nice to have—it is a necessity. Planned downtime is no longer tolerable and tuning of data access must be real time. Unplanned downtime must also be minimized or eliminated through quick recovery and policy-based capacity planning. This presentation will discuss these trends and how storage virtualization can enable IT to meet the demands of today's economy. *(Intermediate Level)*

3:00-4:00
TBA

DEVELOPER TRACK

11:00-12:00 & Continues 1:15-2:15
Top 10 SQL Tuning Tips
Guy Harrison, *Director of Monitoring and Oracle Tools Development, Quest Software*

SQL tuning is usually the most effective way to improve application performance and avoid unnecessary hardware upgrades. In this presentation Guy Harrison – author of *Oracle SQL High Performance Tuning* (Prentice Hall, 2000) and a senior product architect at Quest Software – will provide an overview of the fundamental principles of optimizing SQL performance. *(Beginner to Intermediate level)*

3:00-4:00
Java for PL/SQL Developers
Joe Greenwald,
President and CTO, ODK Inc.

This presentation is for experienced PL/SQL-Forms developers. Joe shows the similarities and differences between PL/SQL and Java, and stresses how easy it is for PL/SQL developers to learn Java. Joe also shows how to integrate PL/SQL and Java: using Java in the database server, calling Java from PL/SQL, calling PL/SQL from Java using JDBC, and calling Java classes from Forms. *(Advanced level)*

DATA WAREHOUSE TRACK

11:00-12:00
TBA

1:15-2:15
New Analytical SQL Functionality in Oracle8i
Willie Albino,
Senior Database Analyst, eBay

In this presentation Willie will provide attendees with an introduction to Oracle8i's enhanced analytical processing capabilities. These capabilities include the CUBE and ROLLUP extensions to the GROUP BY clause of the SELECT statement, a family of analytic SQL functions which enable rankings, moving window calculations, and lead/lag analysis, a family of regression functions, and the CASE expression which provides if-then logic useful in many situations. *(Beginner level)*

3:00-4:00
How We Avoided the Data Warehouse 'Death March'
Chris Lawson, *Oracle DBA consultant*

Data warehouse projects are notable for their poor track record. Often, frustrated developers "jump ship" as the project falls behind, customer dissatisfaction increases, and disaster looms. In this case study, Chris Lawson reviews a successful strategy used for implementing an Oracle 8i "Data Warehouse on the Web" for an e-learning corporation in San Francisco. Besides discussing overall strategy, Chris will also discuss various performance tuning tactics – some that work well in the database, and others that work well in marketing literature. The emphasis in this presentation will be on proven steps that will help the DBA and designer build a successful data warehouse. *(Intermediate level)*

Thank you to our meeting sponsor, Genentech.

NoCOUG Fall Conference

Thursday, November 15, 2001

Location: Embassy Suites, South San Francisco *(located at 250 Gateway Boulevard)*. Sponsored by Genentech.

8:00-9:00	**Registration and welcome. Refreshments served.**
9:00-9:45	**Opening Remarks and Announcements/Vendor Introduction**
9:45 -10:30	***Keynote:* Oracle@Genentech**
10:30-11:00	**Morning Break**
11:00-12:00	**Parallel Sessions**
	DBA Track: *Protecting an Oracle Database*
	Developer Track: *Top 10 SQL Tuning Tips*
	Data Warehouse Track: *TBA*
12:00-1:15	**Lunch Break**
1:15-2:15	**Parallel Sessions**
	DBA Track: *Optimizing Your Oracle Database Through Storage Virtualization*
	Developer Track: *Top 10 SQL Tuning Tips – continued from morning session*
	Data Warehouse Track: *New Analytical SQL Functionality in Oracle8i*
2:15-3:00	**Afternoon Break**
3:00-4:00	**Parallel Sessions**
	DBA Track: *TBA*
	Developer Track: *Java for PL/SQL Developers*
	Data Warehouse Track: *How We Avoided the Data Warehouse 'Death March'*
4:15-???	**Reception/No Host Bar onsite at Embassy Suites**
Cost:	**$40 admission fee for non-members. Includes lunch voucher.**

Gaja Vaidyanatha packed the house at our Summer Conference. Check out what we have in store for our Fall Conference!

For session descriptions, check out page 27.

Keep checking the NoCOUG website for the most updated Fall Conference schedule!

RSVP online at http://www.nocoug.org/rsvp.htm

NoCOUG
P.O. Box 3282
Danville, CA 94526

FIRST-CLASS MAIL
U.S. POSTAGE
PAID
SAN FRANCISCO, CA
PERMIT NO. 11882