

nocoug

JOURNAL

Official Publication of the Northern California Oracle Users Group

VOL. 16, No. 3 · AUGUST, 2002

\$15

Installing and Configuring Oracle9i on the Solaris Platform

by Roger Schrag

Introduction

This paper will walk you through the steps of installing Oracle9i release 2 (Oracle version 9.2.0) in a Sun Solaris SPARC environment. About 90% of the material presented here applies to other platforms as well. Everything you read in this paper is hands on, roll-up-your-sleeves-and-get-busy material for Oracle users who want to get an Oracle database up and running quickly without reading hundreds of pages of documentation and “readme” files.

These steps are meant to get you up and running as fast as possible, while leveraging best practices in order to set up a scalable, robust database environment that offers high performance. In order to keep the steps reasonably simple this paper does not cover Real Application Clusters (RAC), nor does it cover Oracle Internet Directory or Oracle Management Server installation.

There are four phases to getting Oracle up and running on your server:

1. Prepare the server
2. Install the Oracle software and create a starter database
3. Fine tune the starter database (optional)
4. Complete the server configuration

(continued on page 4)

NoCoug Online to Get New Look

We are excited to announce that NoCOUG will be unveiling a new website in August. Our webmaster, Vadim Barilko, has been working behind the scenes to give us an improved look and feel, complete with credit card processing for memberships, discussion groups, and more. Keep an eye out for these great changes at www.nocoug.org!



TABLE OF CONTENTS

Installing and Configuring Oracle9i.....	1
NoCOUG Online to Get New Look.....	1
From the Editor.....	2
NoCOUG Board.....	2
User Groups.....	2
Publication and Submission Format.....	2
Advertising Rates.....	2
President's Message.....	3
Tech Tips.....	4, 5
Board Member Spotlight.....	5
Query Tuning.....	7
Oracle Urban Legends.....	12
Treasurer's Report.....	17
Sponsorship Appreciation.....	17
Resource Corner.....	20
NoCOUG Summer Conference Session Descriptions.....	27
NoCOUG Summer Conference Schedule.....	28

—ADVERTISERS—

XIOtech.....	3
Quest Software.....	6
Promatis.....	8
Informatica.....	8
BMC Software.....	10
dbDoctor.....	22
LECCOTECH.....	22
Promeria.....	25
Database Specialists.....	26
Embarcadero Technologies.....	26

Looking for Volunteers

Do you want to get involved with a great group? The NoCOUG team of volunteers is always in need of an additional pair of helping hands. You'll get to help plan and implement the direction of NoCOUG. And you'll get to know some very fun and smart people along the way. Come join us! Start by sitting in on a board meeting or two, and then decide when and where you might be able to contribute to the success of our users group. Just talk to a board member at the upcoming conference or email us at board@nocoug.org. We look forward to hearing from you!

—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.

NOCOUG BOARD

President

Joel Rosingana, Independent Consultant
joelros@pacbell.net

Vice President

Roger Schrag, Database Specialists, Inc.
rschrag@dbspecialists.com

Treasurer/Secretary

Judy Lyman, Contra Cost County
Public Works
gooma@california.com

Membership

Vacant Position

Webmaster

Vadim Barilko, Independent Consultant
vabarus@onebox.com

Journal Editor

Lisa Loper, Database Specialists, Inc.
lloper@dbspecialists.com

Vendor Relations

Ganesh Sankar, Providian Financial
bgs2k2@yahoo.com

IOUG-A Representative and Past President

Vilin Roufchaie, Cingular Wireless
vilin.roufchaie@cingular.com

Members, At Large

Mike DeVito, Independent Consultant
mike@devito.com

Hamid Minoui, Fritz Companies
hamid.minoui@fritz.com

Darrin Swan, LECCOTECH
darrin@leccotech.com

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 November issue is October 1, 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.

Copyright © 2002 by the Northern California Oracle Users Group. Permission to reproduce articles from this publication, in whole or in part, is given to other computer user groups for nonprofit use, with appropriate credit to the original author and the Northern California Oracle Users Group Journal. All other reproduction is strictly prohibited without written permission of the editor. Two copies of each reprint should be sent to the editor.

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.



Conference Updates

by Joel Rosingana, President, NoCOUG

I OUG Live was fantastic! Maybe it was due to better session management on my part. But, I believe it just had a number of excellent presentations. I concentrated on developer and application sessions. Even though I am a developer, I sat in on a few DBA sessions. I picked Rich Niemiec's (IOUG President) presentations—a beginner and an intermediate. Rich wouldn't accept that I was a beginner and had me run his slides. Actually, this helped me focus more. The moral of the story: conferences like IOUG Live, NoCOUG and Oracle World allow us to attend sessions we wouldn't take the time for in the regular work-a-day world. I'll attend a DBA presentation at a conference. But, I wouldn't schedule a full DBA class—it doesn't normally pertain to my job function.

I also represented NoCOUG at IOUG Live. In that capacity, I attended the GAP (Grassroots Alliance Partner) meeting. This meeting is attended, mostly, by User Group representatives. This is a venue that allows an exchange of ideas between the groups. We are able to obtain help through the discussion of common problems and gain insight from an individual group's successes. This is also the time to present awards. Our two-term Past President, Vilin Roufchaie, was presented with the Outstanding Volunteer Award. Congratulations Vilin! Jeannie Midkiff, of Oracle Corporation, was introduced to the group. Jeannie is in charge of user group relations for Oracle. Jeannie has spent considerable time and effort supporting NoCOUG meetings. I was happy to finally get to meet her. Jeannie has again performed a miracle for our August meeting. Details later in this article.

I mentioned Oracle World (previously Oracle Open World). Vilin

Roufchaie and I were invited to be part of a focus group to determine the variables that make up a successful Oracle World. Vilin and I spent a half a day at the Oracle Conference Center with other like-minded Oracle users. We were encouraged to give our opinions and thoughts about all aspects of producing a successful conference. It is very commendable that Oracle spent the time and money to find ways to improve their conference. I'll be interested to see the results that may come from our round table. A real plus from this experience is that



Joel Rosingana

we are bringing back some great ideas for NoCOUG.

Now that I have given plugs to the weeklong conferences, it's time to toot our own horn. Check out the agenda in this journal, or the website. You will find a top-notch lineup planned for our summer conference. If you take the number of sessions offered over a year, and do the math, NoCOUG is the best education money you can spend. I don't know how anybody can pass up this deal!!

Now for Jeannie's coup I mentioned above. I am pleased to announce that our Keynote Speaker will be Senior Vice President, Oracle 9iAS, Thomas Kurian. With the very strong session lineup, I would RSVP early. This could be a full house!! ▲

Thank you to Chevron for hosting the NoCOUG Summer Conference on Thursday, August 15, 2002. Note that we will be meeting in a new building. Please see the back page for details.

tech[®]
A Seagate Company

(continued from page 1)

We will walk through these phases one at a time, detailing all the steps involved. The end result will be a very usable database that can be scaled up quite large. Of course, every implementation is unique, and you will need to evaluate each step carefully against your particular requirements. However, this paper will get you off to a very solid start.

Prepare the Server

These steps configure your database server so that it will be ready to accept the Oracle software and database. In this

section, we will make sure your server meets Oracle's minimum requirements, create a Unix user and group to "own" the software, and create some directories that will be used by the Oracle software and database. All of the steps in this section are run as the root user.

1. Make sure that your Solaris system has all of the required operating system patches installed. You must have Solaris 2.6 or Solaris 7 (32 bit) or Solaris 8 (32 or 64 bit) to run Oracle9i release 2. If you are running Solaris 2.6 or Solaris 7, the patch requirements are as follows:

Patches for Solaris 2.6 (32 bit)	Status
106040-11 X Input and Output Method patch	Required
105181-15 Kernel patch	Required
105284-25 Motif Runtime Library Patch	Recommended
105490-07 Dynamic linker patch	Recommended
105633-21 OpenWindows 3.6: Xsun patch	Recommended
105568-13 Libthread patch	Recommended
105210-19 LibC patch	Recommended
105669-07 CDE 1.2: libDTSvc patch (dtmail)	Recommended
106409-01 Chinese TrueType fonts patch	Recommended
Patches for Solaris 7 (32 bit)	Status
107636-01 X Input and Output Method patch	Required
106980-05 Libthread patch	Recommended
107607-01 Motif fontlist, fontset, libxm	Recommended
107078-10 Open Windows 3.6.1 Xsun patch	Recommended

(continued on page 18)



TECH TIPS

Proactive Oracle Storage Management: A Method For Predictable System Performance

Gaja Krishna Vaidyanatha of Quest Software Inc. is consistently a top-rated speaker at Oracle-related conferences. At the International Oracle Users Group conference in San Diego earlier this year, he presented a great session on Oracle storage management. "The complexity of enterprise storage management, and its interaction and dependencies on Oracle systems, presents a non-trivial problem to define, scope and manage," writes Vaidyanatha.

In the white paper that accompanied his technical session, Vaidyanatha discusses Proactive Oracle Storage Management (POSM), which addresses the relevant configuration and management issues that confront all DBAs in an Oracle environment. The paper goes on to provide a set of storage best practices for dealing with various storage-centric issues within an Oracle environment. Vaidyanatha writes that, "the ultimate goal that POSM strives to achieve is to provide proactive guidelines to manage storage-related performance issues before they become tuning problems in a production environment." To read this excellent paper, check out http://www.quest.com/whitepapers/Proactive_Oracle_Stor_Mgmt_Gaja.pdf.



Making a Difference

by Lisa Loper

“**I**aking a difference is rewarding,” says Hamid Minoui, who has been a very active board member of NoCOUG since last summer. Hamid is the lead DBA for a large company in San Francisco, but he still makes time to participate in NoCOUG events.

When asked why he participates, Hamid says, “I feel that I am contributing to the community, and that provides a sense of satisfaction for me.” Hamid stays busy on the board providing input, planning and organizing speaker tracks for the meetings, helping on the day of each conference, and more.

He has been attending NoCOUG conferences since 1995, yet he was still surprised at what he learned after joining the board. “I thought there had to be a lot more people behind the scenes making the quarterly meetings as good as they are,” says Hamid. Some things, however, were not a surprise. “I knew there must be some very professional people involved, and my involvement only confirmed this,” says Hamid. “This organization would not be running so well without the people behind it.”

Hamid brings a varied background to the board, along with a lot of experience both in technology and in participating on boards. Hamid, who has been an Oracle DBA since 1995, has always had an interest in computers. Both his bachelor’s and master’s degrees are in computer science. His master’s thesis paper compared relational databases with hierarchical databases, and thus began his interest in database technology. Hamid was a Multics system programmer. After that, he was a Unix system administrator for about ten years prior to making the transition to Oracle DBA. That Sys Admin experience made the transition an easy one for Hamid.



Hamid Minoui

He has always enjoyed being part of a team, and it’s evident when you hear about his involvement in a variety of groups. Hamid has been on boards for organizations ranging from his homeowner’s association to a Persian community group, to Toastmasters International, where he was an area governor in charge of running five Toastmaster clubs.

To do everything he does, Hamid must be a great manager of time. In

addition to his work and community contributions, he has a full family life in the East Bay. He is married and has two children—a son in college who is interested in mechanical engineering and a teenage daughter who is an accomplished composer who plays instruments including piano and guitar.

When there is spare time, Hamid likes to go skiing, ice skating, rollerblading, and windsurfing. He is also quite a music fan whose tastes range from classical to rock of the 1960s-80s including Led Zeppelin and Pink Floyd. He’s an avid movie fan and has an amazing amount of knowledge of films and the Oscars. So, I warn you not to get into a movie trivia contest with Hamid! You can, however, chat with him the next time you see him at a NoCOUG conference. Don’t be shy. You can say a friendly hello in English, French, or Farsi. Hamid speaks all three languages thanks to being born in Iran and spending much of his childhood in France. But that’s another story that you’ll just have to ask him about! ▲



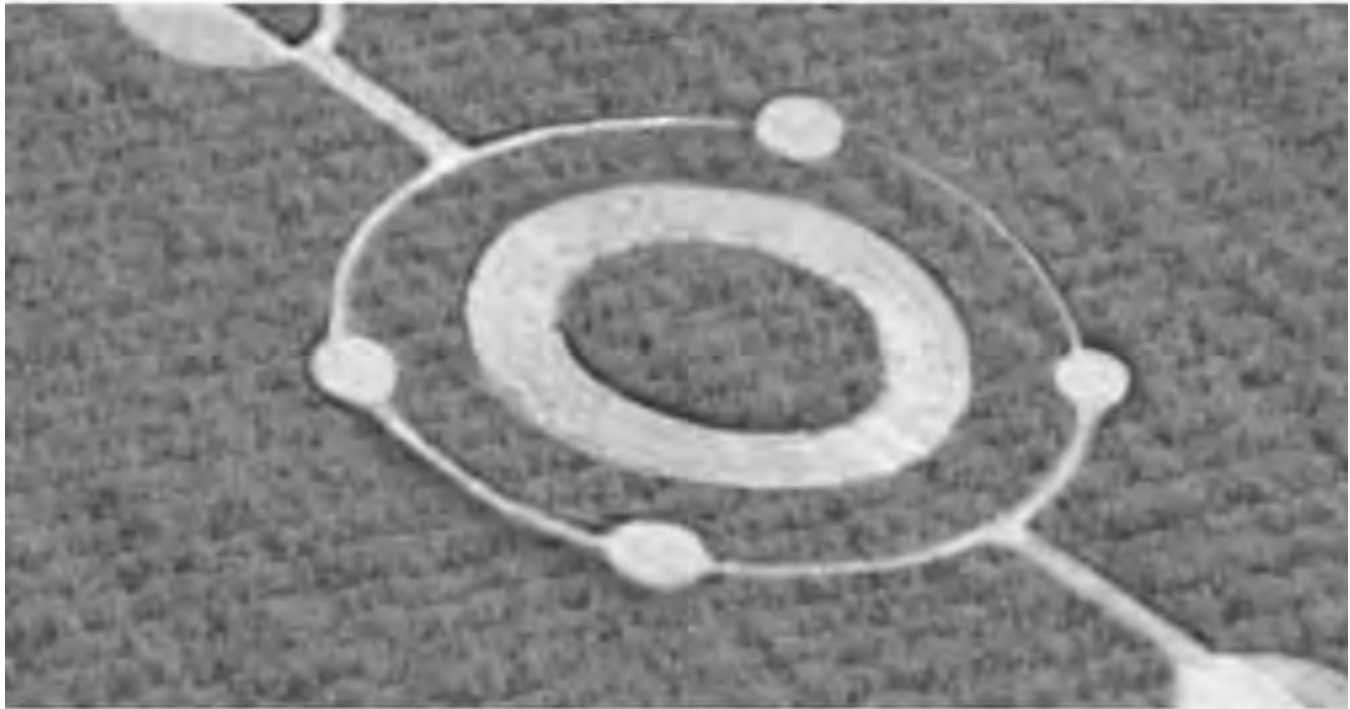
TECH TIPS

Database Forums Online

No matter how long you’ve been working with Oracle technology, there is always something to learn. And a great way to learn is with the help of fellow IT professionals. You’ll find some of them online at dbforums.com. On this site, you’ll find a variety of database forums, and Oracle is at the top of the list. There are also forums on topics such as perl, DBI, PL/SQL, and Unix shell scripting. Anyone can read the advice supplied by members in the forums, however, you’ll have to register to post your own comments.

See what’s available at <http://www.dbforums.com>.

With the Right Tools, Anything is Possible



Quest Software Brings Higher Intelligence to Database Development

Smart people with good tools can mystify the masses and achieve the impossible. With Quest Software's market-leading tools for Oracle development, so can you. Quest Software offers the complete solution to design, develop, test and deploy code quickly and accurately, leaving you plenty of time to consider the possibilities.

The evidence is clear: more than 200,000 users worldwide believe Quest Software's development tools are the intelligent choice. Download your free trial versions from www.quest.com and see for yourself.



www.quest.com • 1.800.306.9329 (U.S.) • 1.949.754.8000 (outside U.S.)

Query Tuning Using dbms_stats

By Dave Ensor, BMC Software, Inc.

Introduction

Increasingly, enterprises are purchasing their mission-critical applications, whether these use Oracle or other data management software. Typically the licensing and support agreements for such applications seek to prevent the customer from reverse engineering or modifying the application in any way. Although such restrictions may be extremely sensible from a supplier's point of view they can prevent an individual site from making changes that would result in valuable performance benefits.

This paper describes the work performed to overcome a specific performance problem in a purchased application without having to resort to the obvious (but impossible) solution of modifying the application code.

Test Environment

The experiments conducted during the preparation of this paper were performed on the author's laptop, a Compaq Armada M700 with 448 Mb of memory and a single 20 Gb hard disk partitioned into a FAT device for the operating system, and an NTFS device for both the Oracle installation and the database files. The single processor is a Pentium III with a reputed clock speed of 833MHz; it certainly executed queries from the Oracle9i cache at impressive speed. The machine was running Microsoft Windows/2000 Professional with Service Pack 2, and Oracle9i Enterprise Edition release 9.0.1.0.1 with Oracle's pre-packaged "general purpose" database, although this was customized in a number of ways that did not affect the issues discussed in this paper.

Background

BMC Software Inc. uses Oracle as the datasever for the majority of its administrative applications, and in most cases the applications themselves are proprietary and run under license. These applications are supported by their vendors but their administration is performed by BMC's internal IS department, which includes a specialist team of Oracle DBAs. As part of their role, this team monitors the resource consumption of the applications and the SQL Statements being run on the data servers. In the summer of 2001 one of the team identified that a very frequently executed statement form in one application was using excessive amounts of CPU time.

The general form of this statement is

```
select all ...
from dm_qual_comp_sp dm_qual_comp,
dm_qual_comp_rp dm_repeating
where (dm_qual_comp.r_object_id in
('080f449c80009d10', '080f449c80009d13', ...))
and (dm_qual_comp.i_has_folder = 1
and dm_qual_comp.i_is_deleted = 0)
and
dm_repeating.r_object_id=dm_qual_comp.r_object_
id;
```

It is worth noting that the select list is not exceptional and is always the same. It has been omitted simply because of its length. The two objects in the from list are both join views, and any number of hexadecimal strings can appear in the in list, though typical forms of the statement contain between 6 and 20 items in the list. The use of the all qualifier and the placement of the parentheses in the where clause may tell us something about the level of Oracle experience of the person writing the statement, but neither is relevant to its performance. Because this statement was deemed to be using excessive CPU (around 250 msec per execution with several thousand executions per hour under peak load) the DBA used BMC's SQL Explorer for Oracle to capture an example of the statement from the shared pool, and started to experiment with various optimizer hints in an effort to improve performance. He quickly discovered that the query executed much more efficiently if it could be persuaded to use hash joins. The tkprof output from his experiments is summarized below.

ORIGINAL STATEMENT							
call	count	cpu	elapsed	disk	query	current	rows
----	-----	---	-----	----	-----	-----	----
Execute	1	0.00	0.01	0	0	0	0
Fetch	8	0.25	0.24	9	17675	0	92

WITH HASH JOIN HINTS							
call	count	cpu	elapsed	disk	query	current	rows
----	-----	---	-----	----	-----	-----	----
Execute	1	0.01	0.01	0	0	0	0
Fetch	8	0.03	0.09	9	331	8	92

It should be noted that the tkprof output has been slightly reformatted to fit on both the page in this paper and on the PowerPoint slides within the associated conference presentation. It is also worth noting that the production application runs on an 8-processor server, and therefore is entirely capable of handling well in excess of the 4,000 executions per hour that might at first sight appear to be the maximum possible service rate.

The statements are clearly being built dynamically for each execution in order to include the literal values in the list. BMC Software does not have access to the source code and was therefore unable to include the required hints directly. At this point the author was contacted and asked whether or not he could think of a strategy that could allow IS to force this particular query form to perform hash joins without needing to change either the application code or the application schema. As discussed earlier, BMC Software does not have the ability to change the code. Schema changes, although technically feasible, would run the risk of being in violation of the relevant support agreement.

Oracle's Cost-Based Optimizer

This paper assumes the use of Oracle's Cost-Based Optimizer (CBO). This is used by the application in ques-

tion rather than the older Rule Based Optimizer (RBO). The latter is still available under Oracle but is best viewed as being present solely for older applications that have been tuned to use it. All new development should assume the use of CBO. However until Oracle9i CBO does not use CPU resource as part of its cost equation, estimating instead what the Oracle documentation tends to refer to as I/O cost but which in reality equates more to “block visits for read.” The difference between (nominal) disk reads and block visits is accounted for by Oracle’s block caching mechanism, and is sometimes magnified by operating system or device controller caching.

CPU Cost

Given that well-tuned Oracle applications are more likely to bottleneck on disk activity than on CPU, it might seem that basing CBO solely on I/O was an inspired decision. Unfortunately, by factoring in the I/O savings of Oracle’s multiblock reads, CBO has a marked tendency to prefer full table scans over indexed access for retrieval of more than about 5% of the rows in a table and this can lead to the execution of query predicate clauses for every row in the table. Depending on the complexity of these predicates, and the SQL features and functions used, the result can cause a surprising CPU load.

As a simple example, using an unindexed 1 million row table with 9,370 blocks and a buffer pool of only 7,864 buffers (to force I/O in each execution) the query

```
select count(memname) records from million;
```

ran in just over 2 seconds and the query

```
select count(memname) records from million
where memb# > 0;
```

took exactly the same elapsed time because the additional CPU required was overlapped with disk reading. However the unlikely variant

```
select count(*) records
from million
where upper(substr(memname,1,1)) between 'A'
and 'Z'
or substr(memname,1,1) in ('_', '/');
```

took over 4 seconds, and the equivalent form using an in list

```
select count(*) records
from million
where upper(substr(memname,1,1)) in
('A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P',
'Q','R','S','T','U','V','W','X','Y','Z');
```

took 12 seconds. As an aside, it may be worth noting that about 50% of the names in this test started with ‘_’ and the query execution time can be varied between about 4 and 16 seconds by moving the position of the ‘_’ within the list. The earlier it appears, the faster the query runs.

Key Statistics

Once statistics have been gathered on a table and indexes, CBO has available to it a number of values that it can use in estimating the number of database block visits that will be required. For tables these include the number of rows in the table, the number of blocks in the table and the average row length; and for indexes the number of

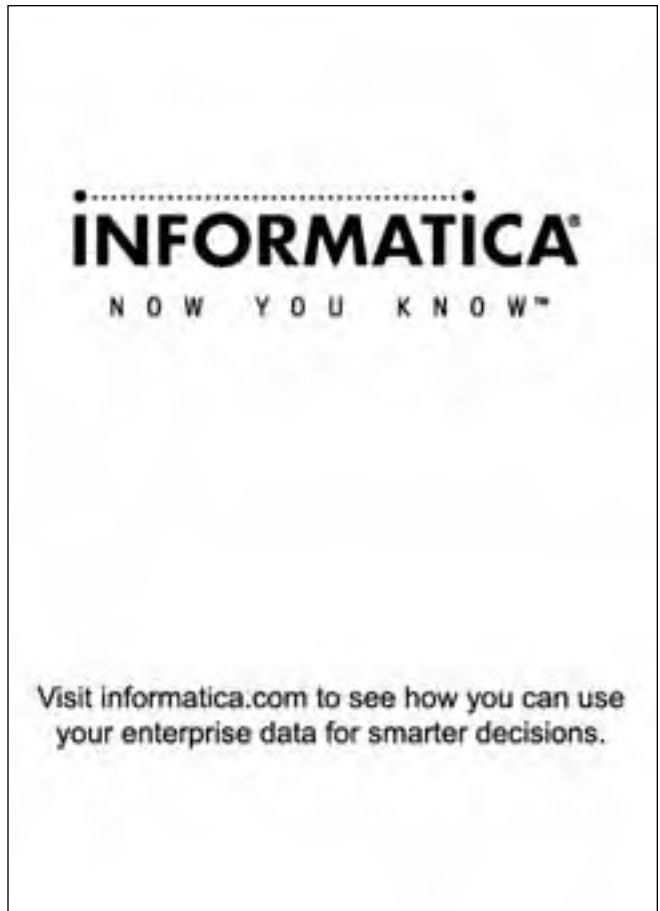


Be the HERO

ROI. Revenue generation. Extending usage. Issues every IT department worries about; especially now.

Be the hero with total and cost-effective enterprise solutions from PROMATIS INCOME Suite. Build business intelligence portals quickly with pre-configured Knowledge Bases. Control and organize knowledge in multiple languages. Integrate all components through the web. Know everything's working with wireless monitoring. Others have pieces; only PROMATIS has it all.


 Make Your Business Fly
get-INCOME.com



INFORMATICA
 NOW YOU KNOW™

Visit informatica.com to see how you can use your enterprise data for smarter decisions.

keys in the index, the number of distinct keys, the number of blocks in the index leaf set and the average number of both leaf blocks and data blocks per distinct key value. In addition, column value histograms can be captured, and these can be used to estimate the selectivity of specific key values.

These statistics are not maintained in real time, but captured on demand using either the SQL command ANALYZE (now deprecated) or the supplied package DBMS_STATS. We can see intuitively that with this kind of information available CBO should be able to make fairly accurate estimates of the number of block visits required to execute each discrete execution step, and at its simplest CBO works out for each of the possible approaches to the query which one generates the lowest total.

Other Factors

Life is not, in reality, quite that simple as there are a number of other factors that influence how CBO works out the notional I/O cost. The simplest to understand is optimizer_mode, which is set at instance level but which can be over-ridden at session level.

At a more complex level, a series of additional parameters (again settable at both index and session level) allow even more variation in how the notional I/O cost is computed. For example, the larger the value of the parameter sort_area_size the more likely CBO is to elect to use a sort because large work areas reduce the disk traffic caused by sorting. If the sort area is large enough then the sort can be performed entirely in memory with no I/O cost whatever. There is even a parameter optimizer_index_cost_adj that takes a value in the range 1 to 10,000 (default 100) and allows index usage to be made more or less attractive (low values make indexes more attractive).

In some simple cases we can get the changes that we need to the query plan simply by changing these parameters, though in many cases we will also need to change the statistics themselves.

Cursor Sharing

The problem query that started the author's investigation contains a number of literal values supplied at run time and therefore the statement is almost certain to be slightly different each time that it is used. Oracle allows DBAs to get around this problem by setting the parameter cursor_sharing = force (the default value is exact).

With force set, any literal value in a SQL statement is replaced with a "bind variable" and thus each of

```
select ORD# from ORDS where OTYPE = 'STD';
select ORD# from ORDS where OTYPE = 'GOV';
select ORD# from ORDS where OTYPE = 'STAFF';
```

will be executed as

```
select ORD# from ORDS where OTYPE = : "SYS_B_0";
```

This has both advantages and disadvantages. The main advantage is that the statement need only be parsed and optimized once, and for a simple query parsing can take longer than executing the query, so the parse saving is significant. The main disadvantage is that if the selectivity of the

key varies the optimizer will not be able to use a different approach for key values of different selectivity. (In the example above 'STD' might select several hundred thousand rows whereas 'STAFF' might only select a few hundred.) This problem is partly overcome in Oracle9i by having the optimizer look at the "bind value" for the first execution, and to use that value to determine the execution path.

Package DBMS_STATS

Paper #

In Oracle9i this Oracle supplied package contains about 35 procedures. Simple usage examples include:

```
dbms_stats.gather_schema_stats ('SCOTT');
dbms_stats.delete_index_stats ('SCOTT',
'EMPPK');
dbms_stats.set_table_stats ('SCOTT', 'EMP'
, numrows => 14
, numblks => 1
, avrglen => 48);
```

The procedures not only allow statistics to be gathered (and deleted) on tables, indexes and column values, but also allow these statistics to be inspected, changed, saved and restored. Statistics can also be moved from one object to another and from an object in one database to an object in another database.

Plan Stability

This relatively recent feature of Oracle sets out to ensure that the same plan is always used for a particular SQL statement even if the optimizer parameters or the object statistics have been changed in the interim. The basic mechanism is to use the SQL statement create outline to create a "stored outline" which contains both the SQL statement itself and a series of Oracle generated hints that should always generate the same query plan as was generated when the outline was created. Creating outlines is quite simple, but managing them can quickly become a major problem if a large number are present within a single database.

Because of the time constraints under which this paper was written it has not been possible to include a full discussion of Plan Stability and the stored outline mechanism, but three observations may be worth noting: First, the author's discussions with DBAs at user sites has persuaded him that the facility is little used and has yet to really impact on the consciousness of the Oracle community. Second, although its use adds a considerable CPU overhead to 'hard parses' this is rarely a significant problem because the feature is intended for high throughput transaction-based applications in which hard parses should be rare except in the period immediately after instance startup. Last, and rather surprisingly, the application of stored outlines to statements being parsed can only be enabled dynamically through alter system statements. It cannot be set in an init.ora or spfile. This is apparently deliberate but the reasoning behind it is unknown to the author.

Getting CBO to the Required Plan

The Oracle manual "Oracle8i Designing and Tuning for Performance" tells us that CBO uses hash joins when the two



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.



“row sources” to be joined together are large and of about the same order of magnitude in size. Forcing the sample query to use hash joins was a relatively simple matter of adjusting the statistics on the four tables that underlie the two views in the query so that CBO felt the conditions for using hash joins achieved. The table sizes in the production system (or at least in the statistics on the production system) were:

```
Table Rows Blocks
-----
DM_QUAL_COMP_R 250 1
DM_QUAL_COMP_S 125 1
DM_SYSOBJECT_R 398,790 1,392
DM_SYSOBJECT_S 271,966 9,313
```

It was realized early in the process that making small changes was invariably ineffective. However, by changing the table sizes to

```
Table Rows Blocks
-----
DM_QUAL_COMP_R 20,000 1,600
DM_QUAL_COMP_S 10,000 800
DM_SYSOBJECT_R 398,790 1,392
DM_SYSOBJECT_S 10,000 800
```

(and also making some changes to index selectivity to avoid index use) it was relatively easy to derive the required execution plan of three hash joins. However this had the obvious drawback that every query that accessed any one of these tables might now use a new execution plan because the statistics had been changed.

Localizing the Impact

The easiest solution to preventing the changed statistics from affecting other queries was to create an outline for the query as soon as it had been tuned, and then to revert the statistics back to their previous values. This still had the effect of destabilizing a production system during the trial and error process of deriving the required plan.

The method adopted was to export both the production schema and the production statistics to a test instance using a combination of the DBMS_STATS package and Oracle’s EXPort and IMPort utilities. Not a single row of application data was transferred, just the schema definition (using EXP with Table Data set to No) and the table and index statistics, by separately exporting the table STATISTICS_TABLE used by DBMS_STATS as both a destination to which to save object statistics and a source from which to restore them. With a production schema and production statistics on the test instance, and having taken some care to ensure that the optimizer environment was the same, we were able to conduct the trial and error process without affecting the production system. We already knew from hint-based tuning what plan we wanted. Once we achieved this, an outline could be built for the statement and could be transferred to the production system. The query being tuned was never actually executed on the test system.

It was suggested to the author that under Oracle9i the supplied package DBMS_OUTLN_EDT could be used. In this scenario all that would be necessary, once the desired plan was known, would be to create an outline with the existing plan and then edit it to achieve the desired plan. Unfortunately, a

brief examination of the package showed that it was nowhere near powerful enough to achieve the required transformations.

Ensuring Outline Use

In order to ensure that the outline was used in production it was necessary not only to enable outlines, but also to set cursor_sharing = force so whatever the literal values appeared in the statement, it would match the statement text stored in the outline table. Sadly, despite the success of the approach, it has not yet been deployed on the production system because of the need for this change, which has been deemed to require a performance test under production load using a test instance. This test has not yet been performed, and as the production server has sufficient CPU to cope with the inefficient query plan at current load volumes, IS have decided not to implement the solution until the load rises to the point at which the inefficiency starts to impact throughput.

Postscript

It is worth noting that Jonathan Lewis has proposed a sneaky alternative approach to the method outlined above for creating the “preferred” stored outline for a query. In this simplified approach, an outline is first created for the original query, and following tuning using hints, a second outline is created for the hinted version of the query. Once both outlines have been created their hint entries in the table OUTLN.OL\$HINTS then (assuming sufficient privilege) these entries can be swapped over using simple update command of the general form

```
update OUTLN.OL$HINTS set
OL_NAME = decode(OL_NAME, 'VERSION1', 'VERSION2', 'VERSION1')
where OL_NAME in ('VERSION1', 'VERSION2');
```

Once the referential links have been reversed in this way, the stored outline for the hinted query will be used whenever the original query is parsed. However, neither Jonathan Lewis nor the author can recommend the use of this approach in a production instance unless, or until, it has been approved by Oracle Corporation, and in particular Oracle Support. It currently seems unlikely that such approval will be forthcoming.

Conclusions

It is clear from the exercise described in this paper that careful modification of object statistics can be an effective way of persuading Oracle to use a better query plan. The technique is not powerful enough to overcome the potential impact of every “query from hell” but in those cases where it is effective it has low implementation cost and is also totally non-intrusive in that no changes are required to any code or script that has been supplied by the application vendor.

However, despite supporting the claim of being non-intrusive, full deployment of the technique may require that a number of changes be made to the instance environment. In particular, it will often be necessary to both enable the use of stored outlines and to cause Oracle to replace literal values with bind variables. ▲

This article is excerpted and reprinted with permission from the IOUG Live! Conference, April 2002.

Oracle Urban Legends

Robert G. Freeman, CSX Technology / 5 Kids Consulting

Abstract Summary

The American Heritage Dictionary defines an Urban legend as follows: “An apocryphal story involving incidents of the recent past, often including elements of humor, and horror, that spreads quickly and is popularly believed to be true.” The Oracle DBA world is full of urban legends. Legends spring from many different sources. Sometimes they arise from invalid information presented in a book or Oracle documentation. Sometimes they arise from some product undocumented feature or bug that was present in the past but has long since been corrected and sometimes it’s not clear where the legends sprang. Urban legends also tend to spring from a lack of knowledge of Oracle and its features. In this paper I will review some of the more common Oracle urban legends, and will attempt to refute them or at the very least provide you with a mind set to not believe everything you see, read, or hear. Indeed, this is the bottom line with regards to urban legends of any kind. Never believe anything until you have investigated it, and determined its validity. Never believe the first thing you read or hear. With regards to Oracle (and software in general) old rules never apply with newer releases and newer technology.

Oracle Urban Legends—A Highlight

There are a number of Oracle urban legends that abound. Perhaps an entire book could be dedicated to the subject. In this paper, I have chosen a few to address in some detail. I will address the legend itself, discuss the impacts of the legend on the Oracle realm today and provide some thoughts on why this legend should die.

The legends included in this paper cover:

- ▶ The books are always right.
- ▶ Rollback Segment Myths.
- ▶ Tablespace datafiles in hot backup mode are not written to.
- ▶ An odd belief that Oracle DATE columns take up 42 bytes of storage, and thus should not be used.
- ▶ Setting TIMED_STATISTICS=TRUE will kill database performance.
- ▶ You need to recreate the control file to increase MAXDATAFILES of a database.
- ▶ NOLOGGING causes all redo generation to stop for a given object.
- ▶ Multiple extents cause performance problems.
- ▶ Hit ratios are the best way to measure performance.
- ▶ There is a magic “Go faster” initialization parameter in Oracle.
- ▶ Increasing the ENQUEUE_RESOURCES parameter can result in a reduction of enqueue waits.

What are the causes of urban legends? Moore’s Law plays a big part. It states that technology capability dou-

bles every 18 months. There are those of us who think this is a bit conservative, and that the rate now seems to be about every 9 to 12 months. With such a rapid pace of growth, it’s no wonder we are left with some odd notions about how things work from time to time. I hope, with this abstract and the accompanying presentation, that I will illuminate one Oracle urban legend that you might have been clinging to.

The Books are Always Right

The Legend: Joe the DBA says, “I read such and such in this Oracle book, it is published by <insert your favorite publisher here> and <insert your favorite author here> is the author. It says that you should never <insert supposed factoid here>, so that must be true.”

The Truth: Go into any bookstore these days and you will find a great number of books dedicated to the subject of the Oracle database. The Oracle book business has been a very profitable business (though, as with the rest of the country, the technical book business has suffered of late). Also, with the pace of technical change, books must be produced at a very fast rate. For example, one of my current books, *Oracle9i New Features*, a 250-page work, was written in about 40 days. Indeed, a very fast pace. The rapid pace of writing, however, lends itself to errors and this is the stuff that urban legends are made of. One missed thought that survives the editing process (or perhaps is created as a result of the editing process), can bring life to an urban legend, and has. Slowing down the pace of writing, is not something that is easy to do. Because of rapidly changing technology, you must strike when the iron is hot. Failure to do so is risking sales of the book and having a book that is not current or topical. The rapid pace of book development is also a symptom of Moore’s Law (with ludicrous drive enabled). In the end, book writing is about making money, maybe not so much to the author but certainly to the publisher.

Take for example books written when Oracle8i was first released. Many of these were written using the beta version of Oracle8i and were published without the authors having had much time (if any) to look at the final product or the features that were contained in it. As a result, some of these books contained incorrect information. Some features in the Oracle8i beta did not initially make it to the production version of the database, and the implementation of certain features (such as DDL statements used to manage those features) changed. Thus, these books contained errors that might have frustrated the reader. Yet, there was also a great deal of good that arose from those books. Other books written using Oracle version 8.1.5 did not include information on the features introduced by later versions of the Oracle8i product. In addition, they might report functional-

ity or errors present only in one version that are not present in later versions. Again, the stuff of urban legends. Authors themselves are the victims of urban legends. Often times, an urban legend is repeated as fact by an author. This is not with malicious intent mind you; the author genuinely believes what he or she is writing is true. Generally, because of the accepted truth of the legend, the statement is rarely put to the test. Often the editing process will not challenge these myths, and they make it into print. I am personally aware of one case where the myth was challenged by a technical reviewer and after some rather exhausting research it was indeed discovered that the fact being reported was indeed not correct, nor was it ever truly correct. Beyond the rapid pace is the writing process. After the author writes a chapter it goes through a number of editing processes. At any of these stages in the lifetime of the text, unintended changes can be introduced that might go unnoticed by the author or technical editors. Trust me, once you have looked at the same page of text ten or twenty different times, your eyes tend to glisten over and you miss things. Editors can inadvertently change a word here, add a word there that changes the meaning of the text, thus providing a breeding ground for yet another urban legend. Even the Oracle documentation has been known to be incorrect, if you can believe that! In fact, errors in older versions of Oracle documentation are the source of more than a couple of Oracle Urban legends. In the end, books are wonderful resources and the authors are pretty smart (and dedicated) people. They are not perfect and are certainly not the last word on any topic (yours truly included!)

Rollback Segment Myths

There are several urban legends that address the topic of Oracle rollback segments. These have impacted how the DBA creates rollback segments, and manages them. Let's look at a few of these legends now.

Oracle Allows Only One Transaction Per Rollback Segment Extent

The Legend: A given rollback segment extent can only be used by a single transaction.

The Truth: I have not been able to determine the exact source of this urban legend. Some have told me it was a hold over from the Oracle6 days, but I have not found any supporting Oracle6 documentation that supports this belief. The bottom line is that a given rollback segment extent can indeed handle multiple transactions at the same time. In fact, in certain cases, multiple transactions can live within the same block in a rollback segment (but the transactions cannot be concurrent). It remains true that a given transaction cannot span more than one rollback segment (with the exception of certain PARALLEL processes, such as a parallel UPDATE statement, which can use many different rollback segments).

You Should Have 20 Extents Per Rollback Segment

The Legend: An optimal database will have 20 extents per rollback segment initially.

The Truth: This legend actually springs from Oracle. This figure appeared in Oracle documentation and even

appeared in Oracle OCP exams. While there is nothing wrong with having 20 extents per rollback segment, what is more important is the INITIAL and NEXT settings for that rollback setting. The number of initial extents in the rollback segment have little to do with performance, whereas the INITIAL and NEXT settings, coupled with the OPTIMAL setting, can make huge impacts on the presence of the irritating snapshot to old errors—which leads us to another urban legend and a more detailed discussion of rollback segments in general.

You Should Have 1 Large Rollback Tablespace for Large Transactions

The Legend: A separate large tablespace should be maintained, with a single large rollback segment, for large, long running transactions.

The Truth: On the face of it, the flaws in this legend should be clear. First of all, this is an incredible waste of space, isn't it? To dedicate space solely to the rollback generated by a single (or a series) of nightly loads, which is (are) not available during the day for other operations, is a waste of resources. An option that I like a great deal, and use successfully, is what I call the large rollback segment extent allocation model. In this model, you create larger extents in your rollback segments and allocate as much space to the rollback segments as you can.

Let's look at this model in a bit more detail. For the purposes of this example, assume I have calculated that at absolute peak my longest expected transaction will be no more than some 20mb in size. Further, I do not expect any more than an additional 40mb in concurrent activity to be occurring at any one time, though I expect that at peak I will have some 200 concurrent transactions active, as this is a high activity OLTP database.

Thus, in this example I am looking at a maximum required rollback segment size of about 60mb. Knowing that I will have an occasional long running operation, I like to add some fluff to the total size that I'm going to allocate to rollback segments, so I'd add another 20mb to my overall rollback segment sizes, for a total of 80mb. This is mostly to avoid snapshot to old errors. Finally, let's build just a bit more cushion when I'm allocating rollback segment space, so let's say we have decided that a 120mb rollback segment tablespace will be sufficient. Now, we know how big our rollback segment tablespace is, how do we allocate our rollback segments? Rollback segment allocation is a subject that is discussed and cussed by DBAs the world over. The primary considerations when creating and sizing rollback segments are:

- The number of concurrent transactions that will occur on the database
- The size of those transactions
- The nature of those transactions

Creating Rollback Segments

An individual rollback segment is limited to a fixed number of concurrent transactions. This is because each transaction assigned to a rollback segment requires a transaction slot in the rollback segment. These slots are pre-allocated in the first block of each rollback segment. The num-

ber of allocated transaction slots are limited by the database block size. Thus, the larger the database block size, the larger the number of concurrent transactions a rollback segment can handle. Based on the database block size, I like to create a number of rollback segments such that one rollback segment will only be expected to deal with 50% of its maximum concurrent transaction load, with a minimum of 4 rollback segments in any event. If my database had a database block size of 8k, then the total number of transaction slots available in any one rollback segment will be 97. Thus, a rollback segment will be able to handle up to 97 concurrent transactions. Since I want to limit this to half of its capacity, that would calculate out to 48 concurrent transactions per rollback segment. As I listed earlier, I expect some 120 concurrent peak transactions to be active, so I divide 200 by 48 and I come up with 5 extents (after a bit of rounding). Having decided that I want 5 rollback segments, I will then take this number and divide it into the amount of space I was going to allocate initially to rollback segments (60mb). Thus, each individual rollback segment will be some 12mb in size. Now I need to define the storage clause for that rollback segment. In this case, I would create some 10 extents in the initial rollback segments (I prefer fewer rather than larger numbers of extents to reduce overall extend and shrink operations). Thus, I would set INITIAL and NEXT to about 1.2MB per extent. In actual practice I'd likely round it up INITIAL and NEXT to 2m per extent because I like to keep my extent sizes even numbers.

Oracle9i Considerations

With the advent of the UNDO tablespace in Oracle9i, the old method of creating separate rollback segment tablespaces is no longer supported. This is because Oracle only supports a single UNDO tablespace in Oracle9i. Also, use of automated undo management does not support the ALTER TRANSACTION USE ROLLBACK SEGMENT statement. In Oracle9i you will need to set the UNDO_RETENTION parameter to avoid the snapshot to old messages. Additionally, with Oracle9i's new feature, resumable space management, the risks of running out of undo space is even further mitigated. Now, as long as you are monitoring your operation, if it should run out of undo space, you can correct the problem without the process actually failing. Corrective action might include eventual release of other undo extents that were previously locked and unavailable, another rollback segment shrinking to optimal, or the diligent DBA adding space to the undo tablespace.

Tablespace Datafiles in Hot Backup Mode are Not Written To

The Legend: It is often cited that datafiles for tablespaces in hot backup mode are not written to by Oracle. In fact, some books have even reported this was the case.

The Truth: Even when in hot backup mode, Oracle is constantly writing to the database datafiles.

This is easily enough verified by observing the change in the date time stamps over the course of the database backup. What is not updated while the datafiles are in hot backup mode is the individual datafile headers. These headers include checkpoint information that is not updated because

of the potential fuzziness of the datafiles on the backup image. Thus, the data in the datafiles is written to, but the file headers are not.

Oracle Data Columns Take Up 42 Bytes of Storage and Should Not Be Used

The Legend: This is an older legend that appears to have sprung from a single statement in an old Oracle DBA book.

The Truth: Oracle DATE data types take up 7 bytes. One byte each for the day, month, year, hour, minute and second.

Setting TIMED_STATISTICS=TRUE Kills Database Performance

The Legend: Setting TIMED_STATISTICS=TRUE is a huge performance hog that will bog down your database.

The Truth: While in earlier versions of Oracle this was true (and that fact is the birth of this urban legend), in later Oracle database versions (with the exception of a few notable bugs which have been fixed) this is certainly not the case. The overhead of the use of TIMED_STATISTICS is now negligible. This fact, coupled with the valuable tuning information that the inclusion of timed statistics can give you, almost makes enabling timed_statistics a requirement these days. Also note that for some time now that Oracle has supported dynamic changes to the timed_statistics parameter. As proof of the negligible performance impacts, I offer this query executed on a 300001 row table. Each execution was done after the database was restarted. The first with timing on, the second with timing off.

```
SQL> select name, value from v$parameter where
name like '%timed%';
NAME VALUE
-----
timed_statistics TRUE
SQL> set timing on
SQL> select count(*) from system.testme;
COUNT(*)
-----
300001
Elapsed: 00:00:17.06
/** Database cycled, timed_statistics is
disabled **/
SQL> select name, value from v$parameter where
name like '%timed%';
NAME VALUE
-----
timed_statistics FALSE
SQL> select count(*) from system.testme;
COUNT(*)
-----
300001
Elapsed: 00:00:18.01
```

As with everything else, there are no absolutes. There may be specific code paths in Oracle that suffer when timed_statistics is enabled, and certainly there is an infinite number of test permutations that might be attempted. My experience of late generally shows that these results are consistent with a variety of database operations.

To increase the Maximum Number of Datafiles in the Database Past Maxdatafiles You Must Re-Create the Control File

The Legend: You need to create a datafile one more than allowed by the MAXDATAFILES parameter that you

defined in the CREATE DATABASE statement. Thus, you must re-create the control file to facilitate this administrative action.

The Truth: This is another legend that started with Oracle of old. In fact, this was true up until just recently. Now all you need to do is simply increase the db_files parameter and you are in business! Note however that this parameter is not dynamic.

Setting a Table to Nologging Will Eliminate Redo Generation for that Table

The Legend: Redo generation will be significantly reduced by using the NOLOGGING attribute of a table.

The Truth: In fact you can only suppress redo generation for only specific types of operations against that table. These operations include:

```
DML Operations
direct-path INSERT (serial or parallel)
Direct Loader (SQL*Loader)
DDL Operations
CREATE TABLE ... AS SELECT
CREATE INDEX
ALTER INDEX ... REBUILD
ALTER INDEX ... REBUILD PARTITION
ALTER INDEX... SPLIT PARTITION
ALTER TABLE ... SPLIT PARTITION
ALTER TABLE ... MOVE PARTITION
```

Thus, even if an operation uses the NOLOGGING hint (or the table is defined as NOLOGGING), if that operation doesn't support NOLOGGING operations, redo will be generated. Specifically, there is no way to eliminate redo generation on DELETE or UPDATE operations, and INSERT operations must use the direct load method (APPEND hint) or CTAS. Even if you are performing an operation that supports NOLOGGING redo generation is not eliminated altogether. Oracle will still generate redo for data dictionary related activities (such as recursive SQL required to add a new object or extent to the data dictionary). Also, Oracle generates redo that will cause a range of blocks to be marked invalid during recovery. In general NOLOGGING operations appear to reduce redo from anywhere between 60 and over 95 percent depending on a number of factors. Here is an example of the impacts of a NOLOGGING operation and proof that even NOLOGGING will generate redo:

```
- Perform a normal insert into a table
SQL> select a.sid, b.name, a.value
1 from v$sesstat a, v$statname b
2 where a.statistic#=b.statistic#
3 and b.name = 'redo size'
4 and sid=8;
SID NAME VALUE
-----
8 redo size 564
1 rows selected.
SQL> insert into testmenew select * from
testme;.300001 rows created.
SQL> commit;
Commit complete.
SQL> select a.sid, b.name, a.value
1 from v$sesstat a, v$statname b
2 where a.statistic#=b.statistic#
```

```
3 and b.name = 'redo size'
4 and sid=8;
SID NAME VALUE
-----
8 redo size 25161544
1 rows selected.
- Now create the table using a direct path
insert.
SQL> select a.sid, b.name, a.value
1 from v$sesstat a, v$statname b
2 where a.statistic#=b.statistic#
3 and b.name = 'redo size'
4 and sid=8;
SID NAME VALUE
-----
8 redo size 564
SQL> insert /*+ APPEND */ into testmenew
select * from testme;
300001 rows created.
SQL> commit;
Commit complete.
SQL> select a.sid, b.name, a.value
1 from v$sesstat a, v$statname b
2 where a.statistic#=b.statistic#
3 and b.name = 'redo size'
4 and sid=8;
SID NAME VALUE
-----
8 redo size 56876
```

Note in this example that each INSERT operation, by default, generates redo. However, when we add the APPEND hint to the insert operation (making it a direct load operation which does not log redo), redo logging is significantly reduced but not eliminated. The NOLOGGING attribute when creating or altering a table is somewhat of a red herring in many cases. NOLOGGING really impacts only a few table and index partition related operations and index build operations. Direct load operations are done without logging (INSERT /*+ APPEND */ or CTAS) do not generate redo sufficient to recover these objects and their data regardless of the LOGGING attribute of the associated table (thus, a backup after a CTAS or APPEND INSERT operation is a good idea).

Multiple Extents Cause Performance Problems

The Legend: Objects with more than about 10 extents will suffer from performance problems.

The Truth: In testing last year under Oracle8i for another IOUG-A presentation, we found that there was little or no impact on performance on an object of up to even 20,000 extents. There were some performance impacts on dictionary-managed objects with regards to TRUNCATE and DROP TABLE statements, but INSERT, UPDATE and DELETE performance was not impacted by the number of extents. Defragmenting a database is an expensive and dangerous operation. Most of the benefits seen from a defragmentation party are not actually due to the defragmentation of the table itself, but rather from the rebuilding of the indexes and, in some cases, an increase of the average number of rows per block in the table. The later case is taken care of by properly setting the objects PCTUSED and PCTFREE parameters. The earlier case is

better handled by performing an online rebuild of the index. Also, consider building a compressed index. With a compressed index, you define how many columns of the index key to compress, and Oracle will only record those column values once per leaf block. This can result in significant savings of space in an index that contains sparse values in one or more of its key columns.

Hit Ratios are the Best Way to Measure Performance

The Legend: The database buffer cache hit ratio is the best way to measure the performance of your database.

The Truth: There can be a number of causes of poor hit ratios that are perfectly reasonable and do not indicate that the database is performing poorly. Data warehouses may well have poor hit ratios, particularly as a result of long running reports. Also, it is the nature of some OLTP databases that very few blocks in the database are actually read with any great frequency. The hit ratios of these kinds of databases are apt to be lower than one might like. Typically, the response to such a hit ratio problem is adding memory to the database buffer cache. With smaller databases this might correct the problem (as the end result is to essentially cache the entire database in memory), yet with larger databases the cost of adding enough memory to the system to make an effective difference is prohibitive, at best. Last year at IOUG-A this topic seemed to be very popular, and it seems that the new trend is to tune based on wait statistics. By reviewing the wait events in `v$sysstat`, `v$sesstat` and `v$waitstat`, the DBA can determine the largest causes of database delays. With regards to tuning application SQL, reviewing your indexing strategy and the use of newer data warehouse features of Oracle such as materialized views, can have more positive benefits than just adding memory to the database buffer cache.

There is a Magic “Go Faster” Initialization Parameter in Oracle

The Legend: There is a dark, mysterious, `init.ora` parameter that only DBAs know about that causes the Oracle database to go faster all of a sudden.

The Truth: The truth is that there are some very basic settings in the database parameter file. No magic settings exist, though there are on occasion settings that can make a difference in special cases. Most often, the biggest gains in performance are to be realized by:

- Making sure that basic database parameter values are correctly set. This includes the database buffer cache, the shared pool, and the redo log buffer.
- Making sure that the database itself is properly built. Issues such as redo logs that are too small, or rollback segments that are not properly adjusted can lead to significant performance problems.
- Making sure the IO is properly distributed. This becomes harder on the newer disk systems, but is just as important. Nowadays this means much more than just making sure that data is on one disk, and indexes are on another. Indeed, it often doesn't matter which disk the data is on, but rather how the data is striped among a set of disks. The stripe size, the number of

disks, the speed of the interfaces and the disks, will all make huge differences in performance.

- Monitoring your database for wait events and tuning based on those events that cause the most impact.
- Properly tuned application SQL is critical to database performance. One badly tuned SQL statement can take the entire database down at most inopportune times.

Increasing the `enqueue_resources` Parameter Can Result in a Reduction of Enqueue Waits

The Legend: Enqueue waits can be corrected by modifying the `enqueue_resources` parameter.

The Truth: An enqueue wait in the `v$system_event` or `v$session_event` views represents the time that an Oracle database session has waited for a lock on another object to be released. Often the Junior DBA will mistakenly attempt to increase the `enqueue_resources` parameter in the database parameter file in an attempt to correct high enqueue wait timeout values. The `enqueue_resources` parameter establishes the number of resources that can be concurrently locked by the lock manager. This parameter rarely needs to be altered, as Oracle derives a value for this parameter at instance startup and will acquire additional enqueues from the shared pool should the number requested exceed the number allocated when the instance was started. Waits on the enqueue event is an application tuning issue, and are indicative of an application acquiring locks on a row which is not released before other sessions attempt to lock the same row. As other sessions attempt to acquire locks on the already locked row, the enqueue wait times will accrue.

Conclusions

Urban legends can cause the DBA unnecessary work. They are often not just harmless stories or experiences. They can cause the DBA to follow courses of action that ultimately are not effective. Likewise, because of urban legends, the DBA may choose to not follow a given course of action that might be very effective. Moore's law has a real impact on our life as DBAs.

In the end, it is up to each of us as DBAs to constantly refresh our knowledge. We must be cautious about what we believe and what we read. Of course, each situation is different, certain rules that apply in one case, do not apply in another. Rules of thumb, more today than ever, actually break quite easily. The DBA has a number of resources to pull from in the quest for better database performance, and the DBA must constantly be updating the knowledge base from which administrative juices spring. Never accept something just because that is the way “it is,” or the way it has always been. You must know the way it is now, otherwise you as a DBA risk becoming an urban legend yourself. Finally, these were some of the more popular Oracle urban legends. Fellow DBAs suggested many of these urban legends. Thanks to them for their suggestions. You probably have your own favorite urban legend too. Please share it with us at masteringoracle.com! ▲

This article is reprinted with permission from the IOUG Live! Conference, April 2002.



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 Joel Rosingana at joelros@pacbell.net.

Long-term event sponsorship:

LOCKHEED MARTIN

CHEVRON

ORACLE

Thank you! Year 2002 Gold Level Support Vendors:

- BMC Software
- Cast Software, Inc.
- Database Specialists, Inc.
- dbDoctor
- Embarcadero Technologies
- Informatica
- LECCOTECH
- Promatis
- Quest Software, Inc.
- XIOTech Corporation

For information about our Gold Level Vendor Program, contact Ganesh Sankar, Vendor Relations, at: bgs2k2@yahoo.com.



TREASURER'S REPORT

Judy Lyman, Treasurer

Beginning Balance

April 1, 2002 \$ 63,055.61

Revenue

Advertising	100.00	
Sponsorships	-	
Membership Dues	49,600.00	
Meeting Fees	200.00	
Vendor Receipts	2,500.00	
Interest	46.61	
Vendor Mailing	-	
Total Revenue		\$ 7,806.61

Expenses

Regional Meeting	2,274.75	
Journal	5,561.74	
Membership	2,127.88	
Administration	1,220.00	
Website	875.00	
Board Meeting	487.46	
Miscellaneous	5500.00	
IRS	430.00	
FTB Tax	10.00	
Total Expenses		\$ 13,486.83

Ending Balance

March 31, 2002 \$ 65,4042.78

(continued from page 4)

You can verify your operating system version and see which patches are installed with the following commands:

```
$ uname -a
$ showrev -p
```

If you are running Solaris 8, no specific patches are required. However, if you are running the 64 bit version of Solaris 8, you should be using the 07/01 release of the operating system. (I am running Oracle9i release 2 on the 04/01 release of 64 bit Solaris 8 without problems, but Oracle Corporation recommends the 07/01 release.) You can check which release of Solaris you are running with the following command:

```
$ cat /etc/release
```

2. Make sure that the following software packages have been installed.

Required Packages
SUNWarc
SUNWbtool
SUNWhea
SUNWlibm
SUNWlibms
SUNWsprout
SUNWtoo
SUNWilof
SUNWxwfont

You can use the following command to verify that a package has been installed:

```
$ pkginfo -i <package name>
```

3. You will need to perform the installation from an X window environment. You cannot perform the installation from a character mode environment such as a telnet or ssh session. There is a facility for performing non-interactive installations (“silent” installs), but we won’t be covering that technique here. Besides, it appears that even the silent install still needs access to X libraries. Your X environment can be the console on the database server, but it does not need to be. You can also use a Windows X emulator like Hummingbird Exceed, but see page 1-15 of the Oracle9i Installation Guide for UNIX for possible issues with Hummingbird Exceed. I ran the installation from the Common Desktop Environment (CDE) on my Solaris 8 desktop and had no problems, but when I tried to use the X environment on my Linux workstations running older versions of Red Hat, I ran into trouble with missing fonts.

4. The following executables must be present somewhere on your path: make, ar, ld, nm.

5. Make sure that your hardware is sufficient. You’ll need at least 512 Mb RAM, a swap space of at least 1 Gb or equal to the amount RAM (whichever is larger), and a bare minimum of 4 Gb of disk space. This will let you perform a typical Standard Edition software installation and create a starter database. A production implementation will almost always require more RAM and more disk space. My server has 1 Gb RAM, and it was swapping heavily during the installation. The following commands will allow you to check RAM and swap space:

```
$ /usr/sbin/prtconf | grep size
$ /usr/sbin/swap -l
```

6. Make sure that the Solaris kernel has parameters set sufficiently high for Oracle. The Oracle architecture makes extensive use of shared memory segments for sharing data among multiple processes and semaphores for handling locking. Many operating systems, including Solaris, do not by default offer sufficient shared memory or semaphores for maintaining an Oracle database. Happily, you can change kernel parameters in Solaris simply by editing the /etc/system file and rebooting the server.

Kernel Parameter	Setting To Get You Started	Purpose
SHMMAX	4294967295	Maximum size of a single shared memory segment
SHMMIN	1	Minimum size of a single shared memory segment
SHMMNI	100	Maximum number of shared memory segments in entire system
SHMSEG	10	Maximum number of shared memory segments one process can attach
SEMMNS	2000	Maximum number of semaphores in entire system
SEMMSL	1000	Maximum number of semaphores per set
SEMMNI	100	Maximum number of semaphore sets in entire system

The first four kernel parameters configure shared memory segments. The recommended settings shown here should be appropriate for almost any Oracle database implementation. The SHMMAX setting may seem excessive, but there is no penalty for setting SHMMAX larger than you actually need.

The last three kernel parameters configure semaphores. Each Oracle instance requires one semaphore for each process, plus ten extras. Additionally, the largest instance requires a second semaphore for each process. If you will only be setting up one database on your server, the upshot is that you will need two semaphores for each process plus ten extras.

The recommended settings for the first two semaphore kernel parameters, SEMMNS and SEMMSL, should be appropriate for most Oracle implementations. For systems with large numbers of concurrent database connections, you may need to increase these values. The recommended setting shown here for SEMMNI should be appropriate for just about any Oracle database implementation.

In general, if your Solaris kernel already has any of these parameters set larger than recommended here, you should not reduce the settings. If you do change any kernel parameter settings in /etc/system, then reboot the server so that the new settings will take effect.

I added the following lines to the end of my /etc/system file:

```
set shmsys:shminfo_shmmax=4294967295
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=100
```

```
set shmsys:shminfo_shmseg=10
set semsys:seminfo_semms=2000
set semsys:seminfo_semmsl=1000
set semsys:seminfo_semmsi=100
```

7. Create a Unix group that will be used by the Oracle software owner and database administrators. You can call it anything you like, but the standard is “dba”. If you will be installing Oracle on multiple servers on your network, you might want to keep the groupid the same on all servers. You can use the admintool, or you can create your dba group with a command like:

```
$ groupadd -g 300 dba
```

8. Create a Unix user that will be the Oracle software owner. You can call it anything you like, but the standard is “oracle”. If you will be installing Oracle on multiple servers on your network, you might want to keep the userid the same on all servers. Note that this user’s home directory will not be the ORACLE_HOME or where the actual Oracle software is installed; this user’s home directory should be in the same place as other users’ home directories. You should make dba the primary group, and the login shell should be Bourne, Korn, or C shell. You can create your oracle user with the admin tool, or with commands like:

```
$ useradd -c 'Oracle software owner' -d /home/oracle \
-g dba -m -u 300 -s /usr/bin/ksh oracle
$ passwd oracle
```

9. Create a Unix group and user that will be used by the Apache HTTP listener integrated into the Oracle9i database. Running the Apache HTTP listener as the Oracle software owner or a member of the dba group can compromise security. You can call the group and user anything you like. At this time there seems to be no clear standard for what to call this group and user. You can create your group and user with the admin tool, or with commands like:

```
$ groupadd -g 60300 apache
$ useradd -c 'Oracle Apache user' -d /home/apache -g apache \
-m -u 60300 -s /usr/bin/ksh apache
$ passwd apache
```

10. Create mount points for the Oracle software and the Oracle database. Each mount point should correspond to a separate physical device or set of devices. You’ll need at least one mount point. Typically you use one mount point for the Oracle software and one or more mount points for each database. A nice convention is to call the mount points /u01, /u02, and so on. Because mount points are typically owned by root and the Oracle installer will run as the oracle user and not as root, you should create some subdirectories now to avoid permission problems later. Create an app subdirectory below the software mount point, and oradata subdirectories below the mount points to be used for database files. (You can put software and a database on the same mount point if you wish.) Make these subdirectories owned by the oracle user and dba group, and give them 755 permissions.

11. Oracle supports files larger than 2 Gb, but your shell must not impose a 2 Gb file size limit for this feature to work. Note that Oracle does limit database data files to 4,194,304 Oracle blocks. You choose the Oracle block size at the tablespace level, and the options range from 2 Kb to 32

Kb. This means that Oracle data files are limited in size to a maximum of 8 Gb to 128 Gb, depending on the block size. It is also important to note that certain Oracle utilities, notably the export utility, are still limited to 2 Gb files. Use the following commands to ensure that the shell imposes no limits on file size:

```
$ ulimit -Sa
$ ulimit -Ha
```

12. If you downloaded a trial version of Oracle off of the Internet, then use cpio to unpack the distribution. If you have the software on CD ROM, then mount the first CD ROM now. Most Solaris systems will automatically mount CD ROMs, but alternatively you can use a command like:

```
$ mount -r -F hsfs device_name /cdrom
```

13. Create the /var/opt/oracle directory and make it owned by the oracle user. After installation, this directory will contain a few small text files that briefly describe the Oracle software installations and databases on the server. These commands will create the directory and give it appropriate permissions:

```
$ mkdir /var/opt/oracle
$ chown oracle:dba /var/opt/oracle
$ chmod 755 /var/opt/oracle
```

Install the Oracle Software and Create a Starter Database

These steps install the Oracle software on your server and create a “starter” database. In this section, we will prepare the oracle user’s environment, run the Oracle Universal Installer, and tidy up a few minor loose ends. All of the steps in this section, except where noted, are run as the oracle user.

1. Edit the oracle user’s login file on the database server so that the environment will be configured automatically on login. If you are using Bourne or Korn shell, then edit .profile. You can also use C shell and edit .cshrc, but the syntax will be different from the examples you see here. For now, we will hardcode certain things. But after the Oracle software is installed we will come back and eliminate all hardcodings. Here is what I added to my .profile for the install:

```
umask 022
# Substitute your Oracle software mount point in the line below.
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=$ORACLE_BASE/product/9.2.0
# Substitute the name of your Oracle database below.
export ORACLE_SID=dev920
# Fill in the following line as you wish, but make sure that
# $ORACLE_HOME/bin, /usr/ccs/bin, /usr/bin, /etc, /usr/openwin/bin,
# and /usr/local/bin are all in the PATH (in that order).
# If you will be installing Pro*C/C++ then the C compiler must be
# on your path. The usual compiler executable on Solaris is located
# in /opt/SUNWpro/bin.
export PATH=...
# Ensure that TWO_TASK is not set.
unset TWO_TASK
```

2. Log out and log back in as the oracle user from an X window so that the environment is set correctly.

3. Set your DISPLAY variable to the IP address of your X server plus the X server and screen numbers. If you are working from a workstation (and not the console of the database server where you are installing Oracle) do not accidentally set the DISPLAY variable to the IP address of your database server. You can set your DISPLAY variable with a command like:

```
$ export DISPLAY=myworkstation:0.0
```

4. If you are not using the console of the database server, then ensure that the X server on your workstation will allow your database server to open windows on your display. The easiest way to do this is to issue an xhost command from a session on your workstation. (Don't get confused and issue the command in a window that is logged onto your database server.) You can issue a command like:

```
$ xhost +mydatabaseserver
```

5. Use ftp to transfer a small file from your database server to a remote host to prove to yourself that TCP/IP networking is installed, configured, and working properly.

6. Ensure that the mount points you plan to use for the Oracle software and starter database have sufficient free space. The starter database will be created entirely on one mount point. For a Standard Edition installation, allow 2.4 Gb for the software mount point and 1.3 Gb for the database mount point as bare minimums. If you are installing the Enterprise Edition of Oracle9i, you will need more space.

7. If you have an active installation of Oracle8i on the database server, then you should make a backup copy of the jre and oui directories under ORACLE_BASE. I found that installing Oracle9i overwrites the JRE and code used by the Oracle8i installer. This means that after you install Oracle9i, you might have difficulties using the Oracle8i tools (such as Database Configuration Assistant or Net8 Configuration Assistant) to manage your Oracle8i databases. After I installed Oracle9i, my Oracle8i tools continued to work properly. But according to postings on Metalink, other users were not as lucky.

8. Double check that you are logged in as oracle and not root. Then change to your home directory and start the Oracle Universal Installer with these commands:

```
$ cd  
$ <full path to first CD ROM>/runInstaller
```

We'll walk through the installer prompts one at a time:

- a. The Welcome window appears. Click Next.
- b. If the Unix Group Name window appears, enter the name of your dba group and click Next. You won't see this window if you have previously installed Oracle8i or Oracle9i

software on the database server, or if your dba group is called "dba". (You won't see this window the next time you run the installer because Oracle saves this information in the /var/opt/oracle/oraInst.loc file.)

c. The File Locations window appears. Leave the Source field unchanged. The Destination field will show the ORACLE_HOME value you set in your environment. Give your ORACLE_HOME a name if you like, and change the mount point if you don't like what you see. Click Next.

d. The Available Products window appears. Choose Oracle9i Database and click Next.

e. The Installation Types window appears. We will perform a "typical" install to get a basic set of Oracle software installed and a starter database. You can rerun the installer again later and choose Custom to install additional products individually. For now, choose Standard Edition or Enterprise Edition. The Enterprise Edition of Oracle9i has some very sophisticated features missing from Standard Edition, and the opportunity to purchase additional options that might be valuable to a large enterprise. However, the Enterprise Edition is much more expensive than Standard Edition. It is very important that you choose the edition that matches your license, as this will be difficult to fix later. I use the Standard Edition. Click Next.

f. The Database Configuration window appears. We will go the easy route here and choose a General Purpose database and click Next. Alternatively, you could choose Customized and tailor the starter database to your own specifications. Or you could choose Software Only and not create a starter database at all. (You can always run the Database Configuration Assistant later to create, drop, or reconfigure databases.)

g. If you have any existing Oracle databases on your server that are at a version prior to what you are now installing, the installer will ask if you would like to run the Database Upgrade Assistant at the end of the installation to migrate or upgrade these older databases to the current version. Make your decision and click Next. (We won't be covering the Database Upgrade Assistant here.)

h. The Database Identification window appears. You need to specify both a global database name and an SID (instance



RESOURCE CORNER

Learn From the Masters

The 2002 University Master Classes are here! The International Oracle Users Group organizes these classes for DBAs of all levels. Experts in the industry teach the essentials every beginner must know and the advanced techniques every DBA desires.

New for this year, the University Master Classes are offered over four days in a modular format designed for maximum flexibility. The next course offering begins October 7 at the Rio Hotel & Casino in Las Vegas, Nevada. You can mix and match modules based on your individual skill level. For more information on the classes and schedules, check out http://www.ioug.org/ioug_p/dynamic_pkg.show?v_name=DISPLAY_UMC_INTRO. ▲

name) for the starter database that will be created. The SID will default to the setting of the ORACLE_SID environment variable, but you can override it here if you wish. You should give your database a global name that is the same as the SID, with your domain name appended. Click Next.

i. The Database File Location window appears. Enter the name of one of the mount points you chose for holding your database, followed by “/oradata”. For example, if your mount point is called /u02, then enter /u02/oradata. The starter database will have all of its files in one directory under this mount point. This may or may not be a good design for high performance and availability, but is fine for a starter database. Click Next.

j. The Database Character Set window appears. Choose the Unicode character set (AL32UTF8) or another character set from the dropdown list if you prefer not to use Unicode. Note that AL32UTF8 complies with the Unicode 3.1 standard, while the UTF8 character set used in Oracle8i complies with the older Unicode 2.0 standard. Click Next.

k. The Summary window appears. Review all of the selections you have made to confirm they are correct. Click Install.

l. If you are installing from CD ROM, you’ll be prompted to mount the second and third CD ROMs at various points during the install. Use a separate window to eject the current CD ROM and mount the next.

m. During the installation a Setup Privileges window will appear. (The installation process took about half an hour to get to this point on my server.) The installation will be paused at this point, waiting for you to run a script as root. The script will be called root.sh and can be found in the ORACLE_HOME directory. You should open another window, log in to the database server as root, review the root.sh script thoroughly, run the script, and click OK in the Setup Privileges window.

n. A Configuration Tools window appears a minute later and the Oracle Net Configuration Assistant launches to configure networking so that your database will be able to accept requests from remote clients. No action is required on your part, and this step completes quickly.

o. The Database Configuration Assistant launches to create a starter database. A progress window will show you how the database creation is going. Database creation took about five minutes on my server, but will take substantially longer if you chose a customized database configuration. When database creation is complete, a window will appear telling you that most user accounts on the database have been locked and that you must change the passwords for the SYS and SYSTEM database users. You may enter the new passwords and click OK. Alternatively, you may click the Password Management button and unlock accounts and set passwords for all database users as you wish.

p. The Agent Configuration Assistant launches to configure Oracle’s “intelligent agent”, a monitoring and job-running agent that you control through Oracle’s Enterprise Manager tool. No action is required on your part, and this step completes quickly.

q. The Apache HTTP listener will now start. No action is required on your part, and this step completes quickly.

r. The End of Installation window appears. You may click

Exit to exit the installer or Next Install to begin another installation. You might click Next Install, for example, to perform a custom installation to install individual products that did not get installed as part of the “typical” installation—such as Pro*C/C++.

s. It is important to note that the default Enterprise Edition install loads certain extra cost options, such as table partitioning, onto your database server. If you are not licensed to use these options, then you should deinstall them. To deinstall products, click the Deinstall Products button on the Welcome window.

t. Exit the installer when you have completed installations and deinstallations.

9. In \$ORACLE_HOME/bin you will find a shell script called oraenv. This script can be called from .profile to set up a user’s environment. Unfortunately, there are a few variables that the script does not set—some handy, some very important. Make a backup copy of the script and then edit it, adding the following lines to the very end:

```
# Begin customizations
ORACLE_BASE=`dirname $ORACLE_HOME`
ORACLE_BASE=`dirname $ORACLE_BASE`
DBA=$ORACLE_BASE/admin
# Substitute the database character set you chose in following line.
NLS_LANG=american_america.AL32UTF8
export ORACLE_BASE DBA NLS_LANG
# End customizations
```

10. In the same directory you’ll also find a shell script called coraenv that can be called from .cshrc. If you use C shell, you will want to back up and edit coraenv with similar changes to the oraenv script.

11. The root.sh script copied oraenv and coraenv from \$ORACLE_HOME/bin to your local bin directory. You just updated these scripts in \$ORACLE_HOME/bin. Copy the updated versions to your local bin directory.

12. In \$ORACLE_HOME/bin you’ll find a script called dbstart. This is a utility that you can run to start up databases on the server. Later we will add a call to this script from /etc/rc2.d so that the databases start up automatically whenever the server reboots. Unfortunately, the dbstart script has a bug that will cause it to fail with the error message “Can’t find init file for Database” in certain situations. One way to fix this bug is to add the following line immediately after line 55:

```
SPFILE=${ORACLE_HOME}/dbs/spfile${ORACLE_SID}.ora
```

and change line 117 to read:

```
if [ -f $PPFILE -o -f $SPPFILE ] ; then
```

13. In \$ORACLE_HOME/bin you’ll find a script called dbshut. This is a utility that you can run to shut down databases on the server. Unfortunately, it shuts down databases with normal priority. This means that if any users are logged into a database, the shutdown will hang until they log out. You might want to change this script to shut down databases with immediate priority. To do this, find the lines in the script that contain just the word “shutdown”. Change these to read “shutdown immediate”.

Fine Tune the Starter Database

These steps modify the configuration of the starter database to tailor it to your needs and to make it better comply

with industry-proven best practices. You can skip this entire section if initially you just want to work with the starter database as is. In this section we will change configured database options, adjust file locations and server parameters, create application users and tablespaces, and configure Oracle Net. All of the steps in this section are run as the oracle user.

1. Set up your environment the same way you did when you ran the Oracle installer: Log in as the oracle user on the database server from an X window. Set your DISPLAY variable appropriately. Make sure that your ORACLE_HOME, PATH, and other variables are set correctly based on your login file.

2. You may run the Database Configuration Assistant to configure database options that were not pre-configured in the starter database, or remove options that were included in the starter database which you don't need. Launch the Database Configuration Assistant with the following commands:

```
$ cd $ORACLE_HOME/bin
$ ./dbca
```

Choose to configure an existing database and select your starter database from the list. You will be presented with a list of database options. The options that are checked have already been configured in the database. Add a check mark beside those options you wish to add, and uncheck those options which you wish to remove. Note that some options will be grayed out. This can happen for three reasons:

a. The option has already been configured in the database and cannot be removed once configured. In this situation

you could choose to delete the entire database and create a new one with only the options you wish.

b. Software products required to support the database option have not been installed. Run the Oracle Universal Installer again and install the appropriate products before trying to configure the option in the database.

c. The option is only available with the Enterprise Edition and you have installed the Standard Edition software.

3. The starter database comes with 100 Mb online redo logs. These might be much larger than you need. You cannot resize online redo logs, but you can drop and recreate them with commands like:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> ALTER DATABASE DROP LOGFILE '/u02/oradata/dev920/redo01.log';
SQL> HOST rm -i /u02/oradata/dev920/redo01.log
SQL> ALTER DATABASE ADD LOGFILE GROUP 1
2 '/u02/oradata/dev920/redo01.log' SIZE 10m;
SQL> ALTER DATABASE DROP LOGFILE '/u02/oradata/dev920/redo02.log';
SQL> HOST rm -i /u02/oradata/dev920/redo02.log
SQL> ALTER DATABASE ADD LOGFILE GROUP 2
2 '/u02/oradata/dev920/redo02.log' SIZE 10m;
SQL> ALTER DATABASE DROP LOGFILE '/u02/oradata/dev920/redo03.log';
SQL> HOST rm -i /u02/oradata/dev920/redo03.log
SQL> ALTER DATABASE ADD LOGFILE GROUP 3
2 '/u02/oradata/dev920/redo03.log' SIZE 10m;
```

Do not answer an rm prompt affirmative unless the corresponding ALTER DATABASE DROP LOGFILE command completed successfully without an error message. If you get an error that a log file is in use when you try to drop it, switch the database to the next online redo log with the command:

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
```

db Doctor
Empowering Database Administrators

Health Watch for Oracle

AUTOMATED, PROACTIVE, SYSTEMIC DATABASE REPORTING

Save time and avoid pain with

- 80+ Automatic Reports
- 90 Day Performance History Archive

visit us on the web, or call for a free trial

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

LECCO SQL Expert
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.

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

- Oracle
- Sybase
- MS SQL Server

Where the experts go for performance. **LECCOTECH**
www.leccotech.com

4. The starter database has all data files and online redo logs in the same directory. If you would like to move any of these files to another directory, use commands like the following:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> SHUTDOWN IMMEDIATE
SQL> STARTUP MOUNT
SQL> HOST mv -i /u02/oradata/dev920/users01.dbf
/u03/oradata/dev920/users01.dbf
SQL> ALTER DATABASE RENAME FILE
2 '/u02/oradata/dev920/users01.dbf' TO
3 '/u03/oradata/dev920/users01.dbf';
SQL> HOST mv -i /u02/oradata/dev920/redo01.log
/u03/oradata/dev920/redo01.log
SQL> ALTER DATABASE RENAME FILE
2 '/u02/oradata/dev920/redo01.log' TO
3 '/u03/oradata/dev920/redo01.log';
SQL> ALTER DATABASE OPEN;
```

Note that this procedure does not work for control files. Relocating database control files will be covered in a later step.

5. In the starter database, all data files have the “auto-extend” feature turned on. This means that when a data file becomes full, it will automatically grow larger as needed. The problem with this is that an application can get out of control and fill up an entire disk partition. It also means that you need to manage your free space at the operating system level. Many DBAs prefer to manage free space at the database level by pre-allocating space to data files and not using the auto-extend feature. You may resize data files and disable auto-extend with commands like:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> ALTER DATABASE DATAFILE '/u02/oradata/dev920/system01.dbf' AUTOEXTEND OFF;
SQL> ALTER DATABASE DATAFILE '/u02/oradata/dev920/undotbs01.dbf' AUTOEXTEND OFF;
SQL> ALTER DATABASE DATAFILE '/u02/oradata/dev920/users01.dbf' AUTOEXTEND OFF;
SQL> ALTER DATABASE DATAFILE '/u02/oradata/dev920/indx01.dbf' AUTOEXTEND OFF;
SQL> ALTER DATABASE DATAFILE '/u02/oradata/dev920/tools01.dbf' AUTOEXTEND OFF;
SQL> ALTER DATABASE TEMPFILE '/u02/oradata/dev920/temp01.dbf' AUTOEXTEND OFF;
SQL> ALTER DATABASE DATAFILE '/u02/oradata/dev920/system01.dbf' RESIZE 300m;
SQL> ALTER DATABASE TEMPFILE '/u02/oradata/dev920/temp01.dbf' RESIZE 100m;
```

Note that if you are using the Enterprise Edition, you should not resize the system01.dbf data file smaller than 500 Mb.

6. Oracle uses a server parameter file or “spfile” to store configuration settings that affect the instance. The parameter settings in the starter database are not bad, but you will probably want to make some changes. Unfortunately, you cannot edit the spfile. Instead, you must export the contents of the spfile to a plain text file called a “pfile”. You can then edit the pfile and convert it back to an spfile for use on your starter database. (This might sound confusing, but is actually pretty straightforward.)

Shut down the database and export the contents of the spfile into a pfile that you can edit with commands like:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> CREATE
PFILE='/home/oracle/dev920params.txt'
2 FROM SPFILE;
SQL> SHUTDOWN IMMEDIATE
```

7. Make a backup copy of the pfile you created in the previous step and edit the pfile to change parameters as you wish, based on your needs and your server’s capabilities. You can always change parameters again in the future, so you are

not locking yourself into anything right now. Here is the pfile that I ended up with:

```
*.background_dump_dest='/u01/app/oracle/admin/d
ev920/bdump'
*.compatible='9.2.0.0'
*.control_files='/u02/oradata/dev920/control01.
ctl', '/u02/oradata/dev920/
control02.ctl', '/u02/oradata/dev920/control03.ctl'
*.core_dump_dest='/u01/app/oracle/admin/dev920/cdump'
*.db_block_size=8192
*.db_cache_size=24m
*.db_domain='dbspecialists.com'
*.db_name='dev920'
*.instance_name='dev920'
*.java_pool_size=0
*.job_queue_processes=10
*.max_dump_file_size=10240
*.open_cursors=300
*.os_authent_prefix=''
*.pga_aggregate_target=24m
*.processes=50
*.remote_login_passwordfile='EXCLUSIVE'
*.shared_pool_size=64m
*.sort_area_size=1048576
*.timed_statistics=TRUE
*.undo_management='AUTO'
*.undo_retention=10800
*.undo_tablespace='UNDOTBS1'
*.user_dump_dest='/u01/app/oracle/admin/dev920/
udump'
```

(Note that I do not use the Oracle JVM, and that is why my java_pool_size is set to 0. You will need to set your java_pool_size to 20m or more if you plan to use the Oracle JVM or other Oracle features that use the JVM.)

8. The starter database has three control files. The control file is a pretty small file that contains crucial configuration and synchronization information that Oracle needs in order to locate all the files that make up the database and keep them consistent. All three copies of the control file are kept identical; whatever Oracle writes to one control file it also writes to the other two. (Think of it like software mirroring.) It is a good idea to move at least one of the control files to another location. With the database shut down, you can go ahead and move the control files around as you wish. Be sure to change the control_files entry in your pfile accordingly.

9. Remove the existing spfile that the Database Configuration Assistant created, and the bogus pfile that it left behind, with the following commands:

```
$ rm -i $ORACLE_HOME/dbs/spfile$ORACLE_SID.ora
$ rm -i $ORACLE_BASE/admin/$ORACLE_SID/pfile/init.ora*
```

10. Create a symbolic link from the location where Oracle looks for the spfile to the location where you will actually maintain the spfile:

```
$ ln -s $ORACLE_BASE/admin/$ORACLE_SID/pfile/spfile$ORACLE_SID.ora \
$ORACLE_HOME/dbs/spfile$ORACLE_SID.ora
```

11. Now convert the pfile that you edited back into an spfile that Oracle can use with the following commands:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> CREATE
SPFILE=' $ORACLE_BASE/admin/$ORACLE_SID/pfile/spfile$ORACLE_SID.ora'
2 FROM PFILE='/home/oracle/dev920params.txt';
```

12. You are now ready to restart your database using your newly created spfile. Use the following commands to start

the database and view the parameters that are in effect. These settings should match what you put in your pfile a few steps back:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> STARTUP
SQL> SET PAGESIZE 100
SQL> SELECT name, value, isdefault
2 FROM v$parameter
3 ORDER BY isdefault, name;
```

13. You can follow the above few steps at any time to make further changes to the parameters. However, if you only have a few changes to make, there is a much easier way than exporting the spfile into a pfile, editing the pfile, and converting back to an spfile. You can simply:

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
SQL> ALTER SYSTEM SET parameter = value
2 SCOPE = SPFILE;
```

This will update the setting in your spfile, and the change will take effect the next time you restart the database. Many parameters are dynamic, meaning that you can change them on the fly without restarting the database. For dynamic parameters, you can omit the SCOPE = line above and Oracle will change the parameter setting immediately and in the spfile.

14. Adjust the configuration of the Oracle Net listener if necessary. You can edit the listener.ora file in \$ORACLE_HOME/network/admin to suit your needs, although you may find the default file to be totally acceptable. Depending on your network topology, you might want to change the hostname or IP. (In my case my server is multi-homed, but I only want the database to accept connections from the internal network.) You should leave the extproc settings as they are; extproc is part of the mechanism that allows PL/SQL to call out to procedures outside the database. My listener.ora file looks like this:

```
#
# Filename: listener.ora
#
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.3)(PORT = 1521))
      )
    )
  )

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /u01/app/oracle/product/9.2.0)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = dev920.dbspecialists.com)
      (ORACLE_HOME = /u01/app/oracle/product/9.2.0)
      (SID_NAME = dev920)
    )
  )
)
```

15. Prepare a tnsnames.ora file in \$ORACLE_HOME/network/admin on the database server and distribute it to all clients. Edit the default file to suit your needs. Change the hostname or IP if needed. My tnsnames.ora file looks like this:

```
#
# Filename: tnsnames.ora
#
EXTPROC_CONNECTION_DATA.DBSPECIALISTS.COM =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
    )
    (CONNECT_DATA =
      (SID = PLSExtProc)
      (PRESENTATION = RO)
    )
  )

DEV920.DBSPECIALISTS.COM =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.3)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = dev920.dbspecialists.com)
    )
  )
)
```

16. At this point the database has two tablespaces available to hold your application tables and indexes: USERS and INDX. However, I recommend that you create new tablespaces for holding application segments instead of using these two tablespaces. Create separate tablespaces with data files on separate physical devices for tables and indexes. You may want to split your application segments into several tablespaces, based on object size, permanence, volatility, I/O volume, or any of a number of other criteria. In the past, choosing storage parameters and allocation schemes for database objects was extremely complex. Now it is quite simple because you can have Oracle do the space allocation and management automatically and it will do a pretty good job. Here is a sample tablespace creation statement:

```
CREATE TABLESPACE small_tables
DATAFILE '/u02/oradata/dev920/small_tables01.dbf' SIZE 500m
EXTENT MANAGEMENT LOCAL AUTOALLOCATE
SEGMENT SPACE MANAGEMENT AUTO;
```

17. Create application roles if desired. Alternatively, you can use the default roles CONNECT, RESOURCE, and DBA.

18. Create your application users that will own the application schemas. Set the default tablespace to one of your application tablespaces designated to hold tables. Assign quotas on all of the application tablespaces where the user will need to be able to create schema objects. (You can use the keyword UNLIMITED.) You should not set any quota on the temporary tablespace. Do not plan to create any application objects in the SYS or SYSTEM schemas, or store any application objects in the SYSTEM or TEMP tablespaces. Here is a sample application user creation statement:

```
CREATE USER bob IDENTIFIED BY bob123
DEFAULT TABLESPACE small_tables
QUOTA UNLIMITED ON small_tables QUOTA UNLIMITED ON large_tables
QUOTA UNLIMITED ON small_indexes QUOTA UNLIMITED ON large_indexes;
```

19. Grant roles and/or system privileges to the application users. Note that if you grant the RESOURCE role to a user, that user will also receive the UNLIMITED TABLESPACE system privilege. This will let the user create objects in any tablespace they wish, regardless of quotas. I recommend you revoke UNLIMITED TABLESPACE from all application users you create. Sample statements to grant and revoke privileges are as follows:

```
GRANT connect, resource TO bob;
REVOKE unlimited tablespace FROM bob;
```


20. Review the overall security of your database. Oracle Corporation has published a handy ten-page listing of security checks that you should perform against your database. Download from the Oracle Technology Network at http://otn.oracle.com/deploy/security/oracle9i/pdf/9iR2_checcklist.pdf.

Complete the Server Configuration

These steps complete the configuration of your server for smooth Oracle operation. In this section we will change the oracle user's login script to eliminate hard-coding, create individual operating system accounts for each database user, and configure the server to start the database and listeners automatically whenever the server is rebooted.

1. Edit the `/var/opt/oracle/oratab` file to verify that the entry for your database is correct. Lines starting with a pound sign are considered comments and are ignored. Each non-comment line contains the name of one Oracle instance, its `ORACLE_HOME`, and a Y or N. A Y indicates that the database should be started automatically on server reboot, and an N indicates that it should not. The three fields should be separated by colons. A sample `/var/opt/oracle/oratab` file looks like this:

```
#
# /var/opt/oracle/oratab
# =====
#
dev920:/u01/app/oracle/product/9.2.0:Y
```

2. Edit the login file (`.profile`) for the oracle user to eliminate hardcodings and call the `oraenv` script to set the environment instead. The following will work with Bourne shell or Korn shell:

```
# Settings for Oracle environment
ORACLE_SID=dev920 # Put your instance name here
ORAENV_ASK=NO
export ORACLE_SID ORAENV_ASK
. oraenv
```

Note that this script assumes that the `/usr/local/bin` directory is on your path. Also, if you use C shell then you should edit `.cshrc` and have it source `coraenv`.

3. Create separate Unix accounts for DBAs and database users who will log onto the database server directly. You should only log in as oracle when installing or patching software. The Unix accounts for DBAs should be members of the `dba` group, and other users should not be members of the `dba` group. Give each of these accounts a login file like oracle's so that their environment initializes correctly when they log in.

4. The Database Configuration Assistant configured the HTTP listener that is integrated into the Oracle9i database and started it up for you. Unfortunately, the HTTP listener was started by the oracle Unix user. Running the HTTP listener as the oracle user and a member of the `dba` Unix group can be a big security hole. Earlier we created a separate Unix user and group to run the HTTP listener. At this time we need to stop the HTTP listener, clean up file ownership problems, and restart the HTTP listener as the apache user. The steps are as follows:

a. Run the following command as the oracle user to stop the HTTP listener:

```
$ $ORACLE_HOME/Apache/Apache/bin/apachectl stop
```

b. Review the `httpd.conf` file in `$ORACLE_HOME/Apache/Apache/conf` and comment out modules that you will not be using. (The default `httpd.conf` file includes everything under the sun.)

c. Change the ownership of certain files from oracle to apache so that the apache Unix user will not get permissions' problems when trying to run the HTTP listener. Which files need a change of ownership depends on which modules you kept active in the `httpd.conf` file. In my system I use the mod PL/SQL feature, but none of the others. Here are the commands I ran to change file ownership:

```
$ cd $ORACLE_HOME/Apache/Apache
$ su
$ chown apache:apache logs logs/*
$ cd ../modplsql
$ chown apache:apache log log/* cfg/wdbsvr.app
```

d. Edit the login file (`.profile`) for the apache user to call the `oraenv` script at login time. (This is the same as what we did for the oracle user's login script in a previous step.) The following will work with Bourne shell or Korn shell:

```
# Settings for Oracle environment
ORACLE_SID=dev920 # Put your instance name here
ORAENV_ASK=NO
export ORACLE_SID ORAENV_ASK
. oraenv
```

e. Start the HTTP listener from the apache Unix user by running the following command while logged in as the apache user:

```
$ $ORACLE_HOME/Apache/Apache/bin/apachectl start
```

5. To make the database and listeners start up automatically when the server reboots and shut down automatically when the server shuts down, you'll need to create a `dbora` file in `/etc/init.d` and link it to `/etc/rc2.d` and `/etc/rc0.d`. You'll need to do this as the root user. First create a file called `dbora` in `/etc/init.d` as follows:

```
#!/bin/sh
ORA_HOME=/u01/app/oracle/product/9.2.0
ORA_OWNER=oracle
HTTP_OWNER=apache
if [ ! -f $ORA_HOME/bin/dbstart ]
```



```

then
  echo "Oracle startup: cannot start"
  exit
fi
case "$1" in
  'start') # Start the Oracle databases and listeners
    su - $ORA_OWNER -c "$ORA_HOME/bin/dbstart"
    su - $ORA_OWNER -c "$ORA_HOME/bin/lsnrctl start"
    su - $HTTP_OWNER -c "$ORA_HOME/Oracle/Oracle/bin/apachectl start"
    ;;
  'stop') # Stop the Oracle databases and listeners
    su - $ORA_OWNER -c "$ORA_HOME/bin/lsnrctl stop"
    su - $ORA_OWNER -c "$ORA_HOME/bin/dbshut"
    su - $HTTP_OWNER -c "$ORA_HOME/Oracle/Oracle/bin/apachectl stop"
    ;;
esac

```

After creating the dbora file, you need to link it to /etc/rc2.d and /etc/rc0.d:

```

ln -s /etc/init.d/dbora /etc/rc2.d/S99dbora
ln -s /etc/init.d/dbora /etc/rc0.d/K10dbora

```

Note that this script starts the HTTP listener as the apache user. If you want your HTTP listener to listen on a privileged port, then you will need to have root start the HTTP listener and hand off ownership to the apache user by setting the User and Group parameters in the Apache configuration file \$ORACLE_HOME/Oracle/Oracle/conf/httpd.conf.

Conclusion

This paper walks you through the intricate details of getting Oracle9i up and running on a database server running SPARC Solaris. It may look complicated, but that's only because this paper goes down to a nitty-gritty level of detail.

Please keep in mind, though, that the requirements are different for every Oracle implementation. I am extremely confident that if you follow these steps to install Oracle9i release 2 (Oracle version 9.2.0) on a server running SPARC Solaris 2.6, 7, or 8, the process will go very smoothly for you. However, no single document can address every specific hardware configuration and every set of business needs. Please use this paper as a starting point to get Oracle up and running in your shop. To get the best performance and scalability, each system needs to be considered individually.

About the Author

Roger Schrag has been an Oracle DBA and application architect for over twelve years. He started out at Oracle Corporation on the Oracle Financials development team and moved into the roles of production DBA and database architect at various companies in the San Francisco Bay Area. Roger is a frequent speaker at Oracle World and the IOUG Live! conferences. He is also vice-president of the Northern California Oracle Users Group. In 1995, Roger founded Database Specialists, Inc., (<http://www.dbspecialists.com>) a consulting firm specializing in business solutions based on Oracle technology. In addition to consulting, the company offers flexible solutions including part-time DBA support (<http://www.dbspecialists.com/dbapro.html>). In 2001, the San Francisco Business Times named Database Specialists one of the Top 150 Fastest-Growing Private Companies in the Bay Area. ▲



Need Part-Time DBA Support?

When you need no-nonsense, straightforward Oracle expertise, call on Database Specialists. We'll roll up our sleeves and help you get things done. With our DBA Pro service, we can configure a part-time or remote DBA program that best suits your needs. You receive:

- A cost-effective and flexible extension of your IT team
- Proactive database maintenance and quick resolution of problems
- Increased database performance and minimized database downtime
- Constant database monitoring with Database Rx™
- Onsite and offsite flexibility
- Reliable support from a stable team of DBAs familiar with your databases

Call Us Today!

415-344-0500 • 888-648-0500
www.dbspecialists.com



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.

NoCOUG Summer Conference

Thursday, August 15, 2002
Session Descriptions

For up-to-date information, see www.nocoug.org

DBA TRACK

11:00-12:00

Oracle Backup and Recovery Essentials Every DBA Should Know

Rob O'Brien, *Product Marketing Manager, VERITAS Software*

This presentation will discuss in detail the technical fundamentals and essentials that every Oracle DBA should know and follow when creating their Oracle RMAN backup, restore, and recovery strategy. Alternative Oracle backup methods such as Block Level Incremental Backup, split mirror backup, and ServerFree backup will also be discussed. *(Intermediate Level)*

1:45-2:45

Making the move from ANALYZE to DBMS_STATS

Hamid Minoui & Rajesh Khanna, *UPS Freight Services*

Since the new statistics for manipulating and gathering packages became available in Oracle 8.1.6, we took the direction of moving away from using ANALYZE to use this facility for collecting CBO statistics. Our motivation to implement this strategy for all our databases will be discussed. *(Intermediate Level)*

3:15-4:15

Information Integration Using Oracle Streams

Bob Thome, *Group Product Manager, Distributed Databases, Oracle Corporation*

Oracle Streams is a new information integration feature and infrastructure introduced in Oracle9i Database Release 2. Information integration is critical for identifying and capturing information from applications and databases, and passing that information to applications, databases, and end users who value it. This session will present an overview of Streams, describe the underlying technology, and introduce some of the solutions you can build using Streams. *(Intermediate Level)*

N-TIER ARCHITECTURE TRACK

11:00-12:00

Oracle 9iAS Release 2 Update

Margaret Mei, *Senior Product Manager, Oracle Corporation*

This presentation introduces the theme and key messages Release 2 of Oracle9iAS. It discusses key new features for all major areas of the application server, including: Transactional Web Applications, Web Services, Integration, Portal, Wireless, Business Intelligence, Performance, Scalability and High Availability, and Management and Security. *(Intermediate Level)*

1:45-2:45

Top Tips for Web-Deployed Forms

Peter Koletzke, *Technical Director, Quovera*

Web-deploying your Forms applications is straightforward, but you must be aware of the things that work differently from client/server and the things that just don't work. This presentation provides the top tips and techniques you need to focus on when deploying forms on the Web. It supplies details on what to think about when designing an application, as well as how to optimize code and objects to run efficiently within a browser. The presentation also provides an overview of the architecture, as well as a discussion of resources that are available for information on the subject. *(Intermediate Level)*

3:15-4:15

Implementing Object-Oriented Inheritance in a Relational Database

Howard Hyde, *President of RADical Information Design Corporation (Intermediate Level)*

SERVER FEATURES FOR DEVELOPERS

11:00-12:00

PL/SQL Native Compilation, an Implementation Story

Roger Schrag, *Senior Consultant, Database Specialists, Inc.*

Starting in Oracle9i it is possible to compile your PL/SQL stored procedures into native machine code on your database server. Oracle documentation claims that natively compiled procedures can run up to ten times faster than interpreted PL/SQL code. We will begin this session with a brief overview of the native compilation feature, and discuss my experiences on a project where a production application including over 35,000 lines of complex PL/SQL code was natively compiled. We'll talk about the actual performance improvements observed, system stability, and the inevitable unexpected issues that popped up along the way. *(Intermediate Level)*

1:45-2:45

Oracle Text

Ron Hardman, *Senior DBA, CommerceOne.com*

Oracle Text is an integrated solution for advanced searches of standard text and many document formats. This presentation introduces Oracle Text and its features, compares functionality with interMedia, and demonstrates how Oracle Text can be used to enhance your application. *(Intermediate Level)*

3:15-4:15

An Introduction To Materialized Views

Willy Albino, *Senior Database Administrator*

In this session we will discuss materialized views, how they work, and the advantages and disadvantages of using them. We cover topics such as instance parameter settings, user privileges, the query rewrite feature, syntax, refresh modes and options, build methods, and dimensions. *(Beginner Level)*

NoCOUG Summer Conference

Thursday, August 15, 2002

Location: The conference building is next door to the Chevron Campus at 6101 Bollinger Canyon Road San Ramon, CA 94583. Please note that the address is different from the last meeting at Chevron. See complete directions to the meeting location at www.nocoug.org.

- 8:00-9:00 **Registration and welcome. Refreshments served.**
- 9:00-9:30 **Opening Remarks and Announcements/Vendor Introduction**
- 9:30-10:30 **Keynote: "Oracle9iAS Technical Overview" with Thomas Kurian, Senior Vice President, Oracle 9iAS, Oracle Corporation**
- 10:30-11:00 **Morning Break**
- 11:00-12:00 **Parallel Sessions**
DBA Track: Oracle Backup and Recovery Essentials Every DBA Should Know
N-Tier Architecture Track: Oracle 9iAS Release 2 Update
Server Features for Developers: PL/SQL Native Compilation, an Implementation Story
- 12:00-1:00 **Lunch Break**
- 1:00-1:40 **Roundtable Discussions for DBA and Developer Tracks**
- 1:45-2:45 **Parallel Sessions**
DBA Track: Making the move from ANALYZE to DBMS_STATS
N-Tier Architecture Track: Top Tips for Web-Deployed Forms
Server Features for Developers: Oracle Text
- 2:45-3:15 **Afternoon Break**
- 3:15-4:15 **Parallel Sessions**
DBA Track: Information Integration Using Oracle Streams
N-Tier Architecture Track: Implementing Object-Oriented Inheritance in a Relational Database
Server Features for Developers: An Introduction To Materialized Views
- 4:30- **Networking and Happy Hour: O'Kane's Irish Pub, 200 Montgomery St., San Ramon**
- Cost: **\$40 admission fee for non-members. Members free. Includes lunch voucher.**



View session descriptions at www.nocoug.org.

Thank you to Chevron, our meeting sponsor.

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