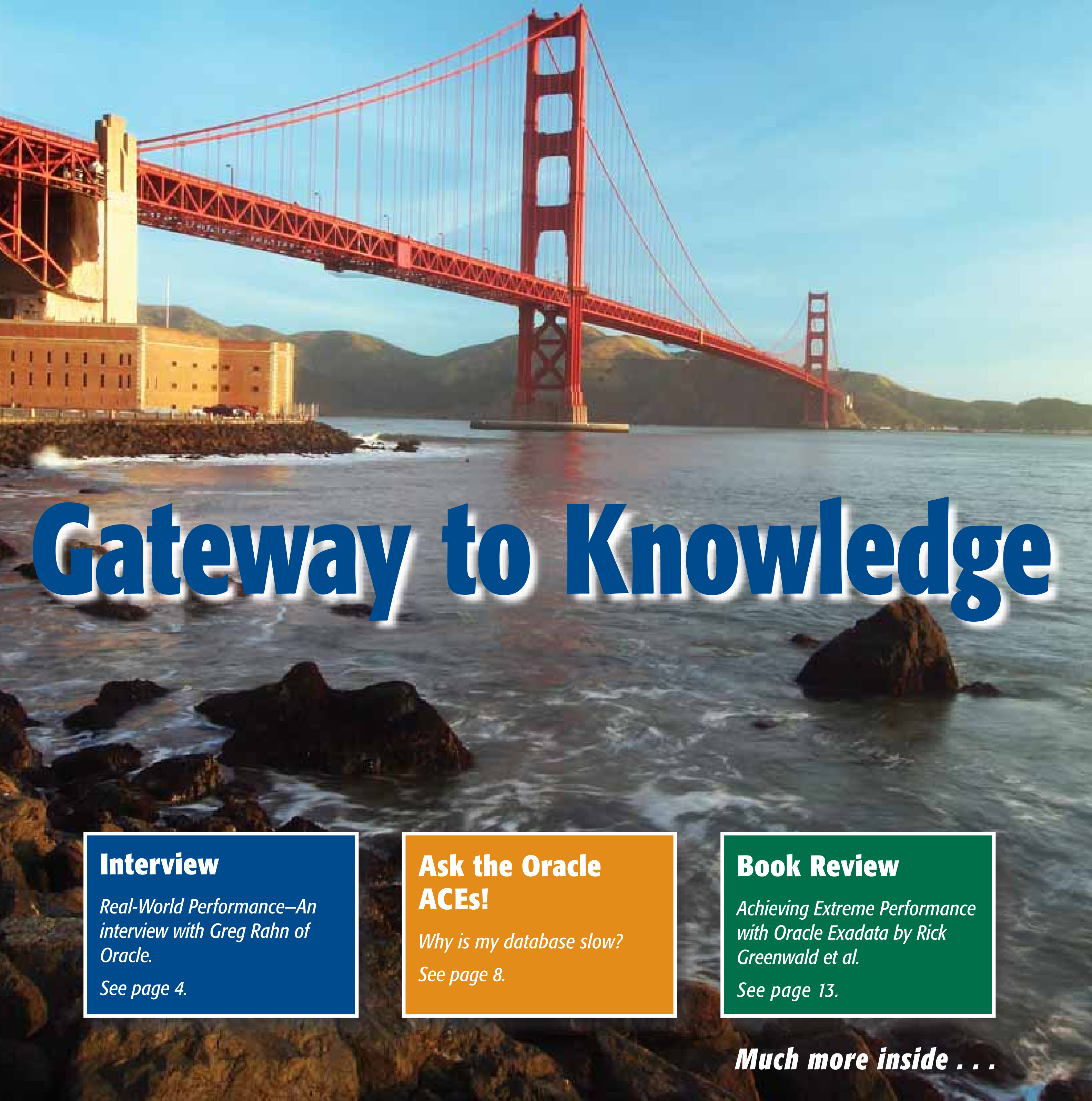
Official Publication of the Northern California Oracle Users Group

NoCOUG JOURNAL

Vol. 25, No. 2 · MAY 2011

$15



Gateway to Knowledge

Interview

Real-World Performance—An interview with Greg Rahn of Oracle.

See page 4.

Ask the Oracle ACEs!

Why is my database slow?

See page 8.

Book Review

Achieving Extreme Performance with Oracle Exadata by Rick Greenwald et al.

See page 13.

Much more inside . . .

Greetings From Your New Editor!

I am thrilled and honored to take over the reins of this *Journal*! I especially want to thank Iggy Fernandez, who has been the editor for the last five years. He set very high standards, and I'll be working very hard to follow in his footsteps.

NoCOUG's speakers, authors, and vendors have given us all a tremendous amount of practical skills and real-world advice. Their generosity and commitment have benefited us all. Even the end users of our database are happier because of the skills they have passed on to us! I am convinced that sharing and collaboration are the key to professional growth and success. With this inspiration in mind, I'll do my best to ensure that this *Journal* continues to be a vital source of knowledge. ▲

—*NoCOUG Journal* Editor

Table of Contents

ADVERTISERS

Publication Notices and Submission Format

The *NoCOUG Journal* is published four times a year by the Northern California Oracle Users Group (NoCOUG) approximately two weeks prior to the quarterly educational conferences.

Please send your questions, feedback, and submissions to the *NoCOUG Journal* editor at **journal@nocoug.org**.

The submission deadline for the upcoming August 2011 issue is July 15, 2011. Article submissions should be made in Microsoft Word format via email.

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

2011 NoCOUG Board

ADVERTISING RATES

The *NoCOUG Journal* is published quarterly.

Size	**Per Issue**	**Per Year**
Quarter Page	$125	$400
Half Page	$250	$800
Full Page	$500	$1,600
Inside Cover	$750	$2,400

Personnel recruitment ads are not accepted.

journal@nocoug.org

Conference #98

by Iggy Fernandez

Iggy Fernandez

It's amazing that NoCOUG is almost 25 years old. Our spring conference on May 19 at the Oracle Conference Center in Redwood Shores will be our 98th conference. NoCOUG member Daniel Kukla has been coming to NoCOUG conferences since 1988. How many conferences have you attended?

NoCOUG provides educational content and networking opportunities that you can't find anywhere else. Here are some inspirational quotes to keep you coming back to NoCOUG for another 25 years.

"Anyone who stops learning is old—whether this happens at twenty or at eighty. Anyone who keeps on learning not only remains young but becomes constantly more valuable—regardless of physical capacity." —*My Philosophy of Industry and Moving Forward* by Henry Ford

"And if you don't know Which to Do
Of all the things in front of you,
Then what you'll have when you are through
Is just a mess without a clue . . ."

—*The Tao of Pooh* by Benjamin Hoff

By taking advantage of the great educational opportunities at NoCOUG, you become constantly more valuable, you learn "which to do of all the things in front of you." On May 19, you'll have the opportunity to listen to Oracle gurus such as James Morle in the comfort of the Oracle auditorium. James Morle likes to say: "*Is Comic Sans* really *a font or a description of many people's storage tier?*" Come and find out what he means on May 19. I hope to see you there. ▲

Real World Performance

with Greg Rahn

Greg Rahn

Greg Rahn is a database performance engineer in the Real-World Performance group at Oracle Corporation. He joined the Real-World Performance group in 2004 and has been working with Oracle databases since 1997. As the name might suggest, the Real-World Performance group deals with database performance and scalability issues that come from the real world—the customer's world. He is the author of the blog **http://structureddata.org/**.

You work in the Real-World Performance Group at Oracle. Can you tell us a little about what you do?

The Real-World Performance Group is a team of database performance engineers within Server Technologies (database development) whose focus is Oracle database performance as customers see and perceive it. We frequently work on competitive POCs for customers as well as performance related escalations.

We love war stories. Can you tell us about a really difficult real world performance problem you were called to assist with?

Several years ago I was at a client who was having performance problems. One of the application engineers made a list of issues on a white board, all of which were related to query performance and execution plan issues. After some thought and several days of analysis and experimentation I told them I was pretty sure there was a single root cause for all of those issues. They gave me the "you're kidding me, right" look. After implementing my suggestions, those query performance problems disappeared. Now you're probably wondering what was the single root cause, right? All the bad execution plans came down to non-representative stats and the technique I used to determine and resolve this was comparing the cardinality of the row sources to the actual number of rows returned. This technique was presented by the Real-World Performance Group at Oracle OpenWorld that year and has since become known to most as "tuning by cardinality feedback."

What is the most common mistake you see customers make when designing their data warehouse?

One of the most common mistakes I see is that customers attempt to design and tune their Oracle data warehouse as if it were an OLTP database. They under size the I/O bandwidth and then apply OLTP tuning techniques, like adding lots of indexes, to try and improve performance. Oracle data warehouses need to leverage the features designed for data warehousing: nologging direct path loads, parallel execution, partitioning, set-based processing, etc. and need a hardware platform designed to support those features.

If there was one thing you could make sure every DBA in the world knew, what would it be?

One thing I think every DBA needs to know is how to be an effective troubleshooter. Part of the challenge is that being good at troubleshooting takes discipline and attention to detail. Too frequently I see DBAs chase symptoms and use a trial-and-error approach instead of drilling down, gathering and analyzing performance metrics and trying to identify root cause.

> *One of the most common mistakes I see is that customers attempt to design and tune their Oracle data warehouse as if it were an OLTP database.*

Many DBAs have trouble using "parallel" correctly. They believe that adding "parallel" to a query or index build will make it faster, but end up with a worse performance problem than they had before. What advice can you give DBAs who are struggling with getting the best performance out of their multi-core systems?

My experience is that DBAs have challenges with Parallel Execution (PX) because their system is not designed to support it. I often hear comments like "PX uses all my CPU or I/O". In order to make effective use of PX, one needs to have a balanced system (appropriate ratios of CPU and I/O). Since PX can consume significant portions of CPU and I/O it is important to leverage database resource management with it as well.

Exadata Smart Scan offloads some of the data filtering to the Exadata storage cells. This improves performance because

(continued on page 6)

Optimizing Oracle – Foundations

A Two-Day Seminar with Jonathan Lewis

August 16–17, 2011 • Register at http://www.speak-tech.com/seminars/

Warning: This is not just another two days packed with sound-bites and quick tips that might work on your particular system. You will learn why tuning is hard but trouble-shooting is easy. You will learn techniques for producing a well-tuned system on day one, and as well as the most common issues that affect performance after a system has gone into production. You will learn how to spot these problems, and measure their impact. Methods, workarounds, dirty tricks and parameters for dealing with classic performance problems will be covered. You will learn how to carefully examine costs, risk, and benefits, and gain familiarity with the key dynamic V$ and X$ performance views.

Management Summary

Why should you send your DBAs and senior developers to this course? Because you want to make the best use of their time, and you want to install systems that don't exceed budget, waste time, and perform badly. If you've paid the Oracle license fees, you need people who understand how it works so that they can avoid the worst design errors, and get the best value for money from your investment. Once your DBAs understand them, and have the tools to help them investigate what's going on, they will be able to solve problems more quickly, and be in a much better position to identify safe and cost-effective opportunities for improving your systems.

Overview

The first day is mainly about how the Cost Based Optimizer does its arithmetic and how you can compare your data with the optimizer's "picture" of your data. The last session of the day will be about finding and interpreting execution plans.

The second day starts with a session on the problems that we can run into when looking for, or interpreting, execution plans, followed by some time looking at indexing strategies, the way Oracle can use indexes, and some common misunderstandings about the strengths and weaknesses of Oracle's B*Tree implementation. We end the day by revisiting the optimizer to ask (and answer) the question—how do we make sure the optimizer gets a good picture of what our data looks like.

Day 1

CBO Session 1: Why isn't Oracle using my index?

CBO Session 2: Mechanisms of Joins

CBO Session 3: Selectivity, Joins, and hints

Explain Plan Session 1 and 2: Finding and Reading Plans

Day 2

Explain Plan Session 3: Problems with plans

Indexes: The use of indexes

Optimal SQL: Methods and practices for optimal SQL

CBO Session 4: Maximizing the truth

Pricing

$600 for members

$700 for non-members

Jonathan Lewis is an independent Oracle Consultant at JL Computer Consultancy and a specialist in core Oracle database technology since 1988. Has worked for around 20 of the FTSE 100 and a similar number of Fortune 100 companies, typically on high-profile performance problems. He the author of Practical Oracle 8*i*, Building Efficient Databases, *published by Addison-Wesley, and* Cost Based Oracle – Fundamentals, *published by Apress, with contributions to three other books to his name.*

Exadata features like Smart Scan, Hybrid Columnar Compression, Storage Indexes, and Parallel Execution make it fun to play with big data on Exadata

(continued from page 5)

the blocks are not read into the SGA, but how do we ensure read consistency?

The Oracle database and Exadata software have checks to determine which, if any, blocks have changed since the query started. If need be, a full block is sent to the database grid and a CR copy is created to ensure read consistency.

Too frequently I see DBAs chase symptoms and use a trial-and-error approach instead of drilling down, gathering and analyzing performance metrics and trying to identify root cause.

I often hear that one should limit the number of connections to the database to be less than the number of cores on the DB server. The intention is to prevent logon storms. However, I'm not sure this is the best tuning advice in an IO bound system. Any recommendations to development teams on how to tune their connection pools?

I'm not sure the number of connections should be limited to the number of CPU cores, but if the application is making effective use of the connections, one generally does not need more than a multiple or two of the CPU count (say 2-3x CPU count). Having too many database connections for the connection pool can result in a logon storm if there is a failure and the more connections there are, the longer that failover will take. My advice is to use the fewest connections possible and also to keep the pool static; do not let it grow. The reason for this is that when a database slows down, the last thing you want to do is create more connections to that database.

Some DBAs got Exadata for Christmas. What do you recommend they'll learn or unlearn in order to get the best value from their new toy?

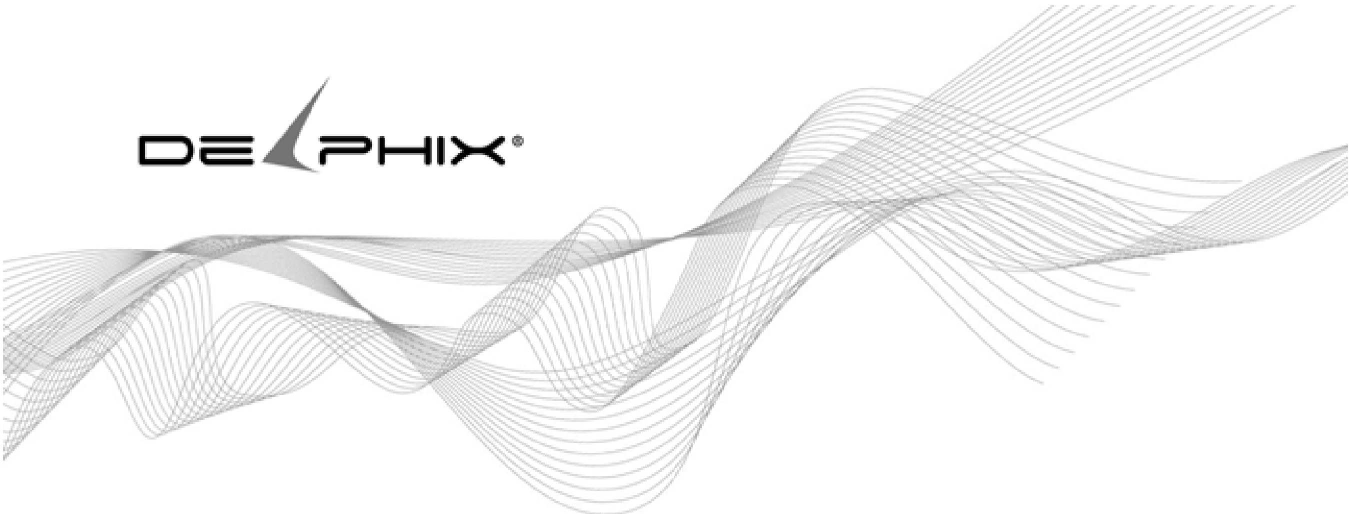
Exadata is a very powerful and scalable platform and to get the most out of it you need to make sure you are leveraging its features. I think the biggest change for many DBAs is ability to efficiently do what was previously impossible or very challenging in their data warehouse environment. One specific that comes to mind is running queries on very large data sets with a very high degree of parallelism (DOP). By high, I mean a DOP in the hundreds. Also, for most data warehouses POCs that I have worked on, I rarely use indexes any more—optimized parallel table scans are very fast. The combination of Exadata features like Smart Scan, Hybrid Columnar Compression, Storage Indexes, etc. in conjunction with Parallel Execution make it fun to play with big data on Exadata.

DBAs feel very replaceable these days. Oracle is making the database easier to use with each version, and these days it appears that even junior DBAs can successfully administer large and complex systems. Many of the skills that "old school" DBAs are proud of are no longer very useful for most companies and most databases. What can senior DBAs do to keep adding values to their companies and prevent their jobs from moving to cheaper junior colleagues?

For most data warehouses POCs that I have worked on, I rarely use indexes any more—optimized parallel table scans are very fast.

One thing that I think most senior DBAs need to do more of is coach and mentor database developers on how to get the most from their Oracle database. There seems to be a knowledge gap on how to write database applications and how to write well performing and scalable database applications. I think this gap can be closed by getting senior DBAs involved early in the development cycle so that common mistakes can be avoided. ▲

Interview conducted by Gwen (Chen) Shapira



Why is My Database Slow?

Ask the Oracle ACEs!

Karen Morton: Why is my database slow? The easy answer to that question is "It depends." The question isn't really scoped well enough to allow a better initial answer. I believe that when a question is formulated that way, there is another question that the performance analyst must uncover. The real issue, and the real question, is about determining which specific business task is slow. While it is certainly not impossible that the entire database is performing slowly, it is more likely that the person complaining about the database being slow has a particular business task they are attempting to execute, and it is performing outside their normal response-time expectations.

When asked this question, my response would be to ask more questions in return.

1. What specifically is executing more slowly than you would expect?
2. What is your current response time versus your normal/expected response time?
3. When did you notice it was "slow"?
4. Did it get worse suddenly, or has the response time been steadily declining?

The main issue, as I see it, is making sure you are working on the "right" problem. There's not much worse than spending time to fix something you believe will solve the problem, only to find out later that what you fixed had absolutely no effect on the reported problem.

One of the biggest roadblocks to answering the "why is my database slow?" question is the tendency to guess or assume. The key is to take the time initially to determine, with as much accuracy as possible, the specific problem that needs to be addressed so hit-or-miss guessing can be avoided.

As I mentioned above, asking more probing questions is the key. There is a great short article at **www.shmula.com/ask-why-five-times-about-every-matter/382/**, which I think is worth reading, that speaks to this point. The article states: "Basically, it is a simple approach of asking 'why' several times until you arrive at an atomic but actionable item." That's exactly it!

Once you have determined the specific problem you need to diagnose, then you need to collect the data you will need to diagnose the problem most accurately. You may be able to use AWR, ASH, or StatsPack data to help get specific details about the poorly performing task. You may find that using extended SQL trace to capture the response time data for this task will give you the data you need (this is my preference when I can have it).

The key is to take the time initially to determine, with as much accuracy as possible, the specific problem that needs to be addressed so hit-or-miss guessing can be avoided.

With the data on hand, you can analyze what is causing the slowness. But the question was "why," not "what," right? However, knowing what is wrong is the primary step in determining why. Once you nail down what is wrong, either it will be fairly obvious why it is a problem and how to fix it (bad SQL, improper instance parameter settings, etc.) or it will give you the data and direction you need to drill down to find the why (bad network device, IO subsystem configuration, etc.) and to correct it.

Diagnosing the root cause of why the database is slow is a process. The process is based on continuing to ask questions (google "ask why 5 times") that will lead you to a single item upon which to take action. So, the ultimate answer to why is "My database slow comes from continuing to ask why." ▲

Karen Morton is a consultant and educator specializing in application optimization in both shoulder-to-shoulder consulting engagements and classroom settings. She is a senior DBA performance and tuning specialist for Fidelity Information Services. For over 20 years, Karen has worked in information technology. Starting as a mainframe programmer and developer, she has been a DBA and a data architect and now is a researcher, educator, and consultant. Having used Oracle since the early '90s, she began teaching others how to use Oracle over a decade ago. Karen is a frequent speaker at conferences and user groups, an Oracle ACE, and a member of the OakTable network (an informal association of "Oracle scientists" who are well known throughout the Oracle community). She blogs at **karenmorton.blogspot.com**.

John Kanagaraj: As the database lead for the Application Performance Testing and Management team in my organization, this question is asked of me almost daily! My first response is usually: "How do we know it is the database that is slow?" This question helps me launch a dialog that can be facilitated using the two simple diagrams below.

Both diagrams show the execution profile of an online application, with the user on the left using a PC (signified by the "User Desktop" box) to access an application housed in the "Data Center." The vertical lines in each of the boxes represent the various components involved in the application—Browser, Webserver, JVM or Application server, Database, OS and Storage subsystem. The desktop and data center are connected by a network—either a LAN or a WAN. The blue oval represents the end to end response time experienced by users of this online application when they submit a transaction. The various shaded boxes show the time spent in the various technology components listed above. The total response time is a collection of the times spent in each of these components. With me so far? Now let's look at two scenarios, both of which have the same end to end "application response time," but have totally different performance characteristics under the covers.

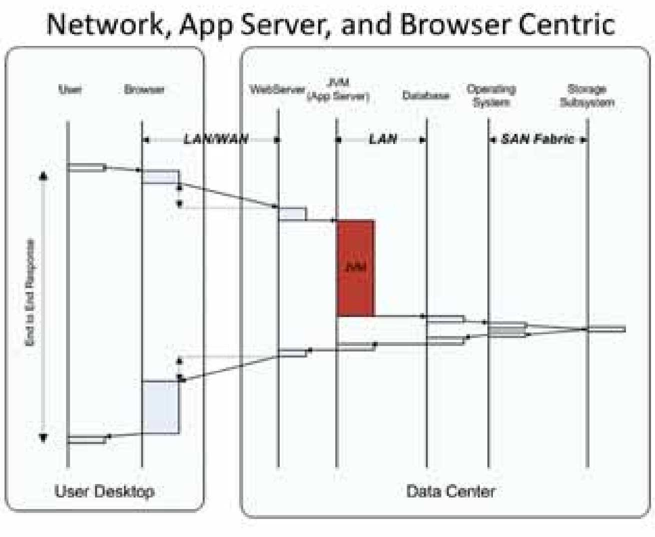


Figure 1

In figure 1, it is apparent that the *majority* of the time is spent in components other than the database, i.e. in the network, the JVM or App server layer as well as within the browser itself. The largest component of time shown above is the time spent in the JVM layer. This could be due to time spent in JVM "full garbage collection" cycles due to inefficient or incorrect memory allocation and usage, thread synchronization and thread locking issues, etc. There is also a significant component of time spent in the network, probably due to network latencies. When a large number of small pieces or packets are sent back and forth on the network, with each packet representing a round trip, these network latencies are quickly amplified. Accessing a data center located in the other coast over the WAN could typically add 70 to 80 millisecond latency to each round trip, with the time spent on the network amplified proportional to the *number* of round trips. In other words, a "chatty" application developed and tested against a local database (typical network latency of 1 to 2 milliseconds) will perform well for "local" users and blow up when used by other "remote" users in other geographies. "Rich Web Applications" could also require significant memory requirements and extra processing work for the browser, resulting in slower responses for PCs with lesser CPU speeds and RAM. In this case, although the question was "Why is my *database* slow?", it is clear that the problem is not with the *database* but is due to issues in other layers. It is also apparent that fine tuning the database will not fetch significant improvement in response times in this case.

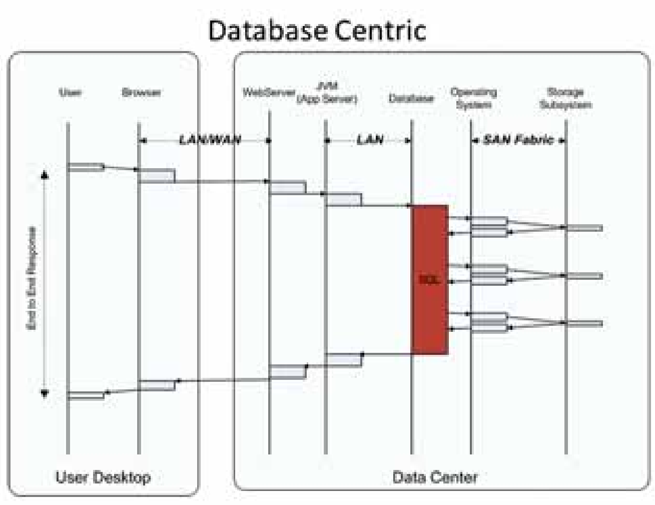


Figure 2

The picture in Figure 2, however, is quite different. It is clear that the majority of the work is in the database layer, with SQL processing resulting in multiple OS and I/O storage calls. The end user response time can be improved by tuning the database layer in this case, and the DBA/SQL Tuner can address this by a combination of SQL tuning, SQL rewrite or application redesign that aims to reduce *buffer gets* (logical I/O) and thus reduce physical I/O. Once this is done, database/instance tuning and storage and I/O layout optimization may also be attempted if that is indeed warranted.

In both cases, we kept the picture as simple as possible—the application above was serviced by a single Web server/App Server/Database stack, all residing in the same data center. This could very easily have been a multiple Web/App server con-

Understanding the end to end architecture . . . employing the right tools . . . and partnering with development teams . . . is the first and essential step in solving application performance issues.

necting to a multi-node RAC database via SOA services and other databases in other remote data centers linked up through Database links. As you can imagine, the application landscape can be and usually is much more complex than the picture listed above.

Although we did not directly answer the question "Why is my database slow?", the point we make here is this: Understanding the end to end architecture, the various technology layers and their interaction with each other, employing the right tools to monitor and measure response times for each of these components, partnering with the application development and support teams to understand and document application architecture, behavior and requirements is the first and essential step in solving application performance issues. Once this is done, you need to combine all this information into a cohesive picture to first understand and then solve such performance issues using the right skills and techniques. Given all this, maybe we should turn the question around to gently ask "Why is the *Application* slow?" and launch the dialog I have laid out above. ▲

John Kanagaraj is an IT Architect with Cisco Systems Inc in California's Silicon Valley. John is always ready to share the knowledge he has gained from two decades of UNIX and Oracle via technical articles and presentations. He currently serves as the Associate Editor of the IOUG's SELECT Journal. *John is also a long-time Oracle ACE, author and technical editor of various Oracle books, frequent speaker at IOUG/OAUG's COLLABORATE, Oracle Open World, RACSIG, NoCOUG and other Regional and International Oracle User Groups. He can be reached via email at* **ora_apps_dba_y@yahoo.com** *or via his LinkedIn page at* **http://www.linkedin.com/in/johnkanagaraj**.

Kristofferson "Karl" Arao: Indeed, it's a short question, yet challenging to answer. We always get this type of interesting question from the end users, and, as usual, they always want their applications to be fast. But where will you start? If you understand the user problem well, if you are systematic with the troubleshooting, and if you know how to use the right tools for the right problem, finding "why" the database is slow will always be a fun and interesting exercise.

That is pretty much my troubleshooting principle—Understand the problem, have a Systematic and Layered approach, and then Fix.

Let me detail each of them:

Understand the problem

So, let's say it's a "sudden" database slowdown. Would you instantly increase the buffer cache? Or get the top SQL and start tuning it? Well, that's a big **no**. I would first interview the parties involved and let them speak about the problem that's happening.

In that way, I would know if it's for a particular session or a system-wide slowdown. This will also give me an idea what they are trying to do, as well as insight into their important business tasks, processes, and modules that I may have to check and drill down on, so it will be easier for me to quantify the problem they are experiencing.

> *If you understand the user problem well, troubleshoot systematically, and use the right tools, finding why the database is slow will always be a fun and interesting exercise.*

The bottom line is, you don't have to guess. It just takes facts, numbers, and figures (F.N.F.) to solve a performance problem. So the objective is to know how to get the F.N.F. right, and then naturally you will come up with a performance story.

Systematic and layered approach

Now that I have identified the areas the tuning efforts should focus on, it is critical not to "assume," even though I may probably know what the problem is all about. In this case, I would continue to be curious about the problem and let the performance numbers guide me on my troubleshooting. Also, to effectively diagnose the cause of the performance degradation, I would always divide and conquer on the different subsystems and then diagnose them "separately" and in "combination." The different subsystems would be the following:

a) **Oracle**
   - ➤ database configuration
   - ➤ instance-wide performance
   - ➤ session-level performance

b) **Operating System**
   - ➤ CPU, IO, memory, network, process info

c) **Application**
   - ➤ SQLs and anything specific to the application

By doing the performance analysis from multiple perspectives, holistic problem isolation will be achieved. All of the data points will be analyzed, and bottlenecks found will be correlated to the different subsystems and specific user activity. The result of the correlation will explain "why" the database is having such performance. It will also allow me to clearly explain to the end users how one subsystem is affecting the other, and detail which resource hoggers contribute a lot to the response time of the database server.

This systematic and layered approach is very applicable for trivial and non-trivial performance problems, and once you get used to this process, troubleshooting will be fun and you will never, ever go wrong. And yes, these are all based on facts, numbers, and figures.

Fix

The systematic and layered approach has clearly shown the issues and opportunities for performance improvement. And since we have looked at it in a holistic way, naturally we will have multiple solutions for each subsystem, attacking the same performance problem.

Having the correlated performance analysis and solutions from different subsystems will also allow us to communicate better with the rest of the team and have stress-free discussions with the people in charge of those subsystems. And you'll be surprised that they will be more involved because everything is based on facts, numbers, and figures.

Ultimately, applying the targeted solutions will result in great workload reduction, response-time optimization, and huge savings on system resources. ▲

*Kristofferson "Karl" Arao is a senior technical consultant at Enkitec, where he is currently busy exploring Exadata performance. Some of his experiences as a DBA include proactive database maintenance, database installation and configuration for high availability and disaster recovery, migration and upgrades, backup and recovery, database hardening, capacity planning, and performance tuning. He shares his experiences, adventures, and discoveries in his blog (***karlarao.wordpress.com***) and owns a Wiki site (***karlarao.tiddlyspot.com***) where he shares his quick guides and documentations on technologies. Karl Arao is an Oracle ACE (***bit.ly/karlarao***), OCP-DBA, and RHCE. He presents at local Oracle events and DBA roundtables, and has also presented at the Oracle Open World Unconference, Oracle Closed World, and Hotsos. He is also a proud member of the Philippine Software Industry Association Technology Council.*

Gwen Shapira: Ask any respectable performance expert why your query takes so long to run, and she will ask you to trace the query and check what wait events is it waiting on. Wait events? That is time spent waiting in queues. Is your query is waiting for "DB File Sequential Read"? That is the IO queue. How about "Library Cache Latch"? A latch could have several sessions waiting, each burning CPU. Is your CPU is at haevily utilized? Use VMStat to check the run queue length: the processes waiting in line for the CPU. Whenever you have a performance problem, you have sessions waiting in queues.

Some people, when confronted with a problem, think "I know, I'll use a queue." Now they have an unbounded number of problems.

Ted Dziuba, in his blog post "The Case Against Queues" wrote: "Some people, when confronted with a problem, think 'I know, I'll use a queue.' Now they have an unbounded number of problems."

Obviously we can never completely get rid of queues. The rate of new requests fluctuates, and the rate at which the server can handle tham also fluctuates. You can minimize queues, but you can never eliminate them.

The application has a better idea of which requests are more critical, and so it can prioritize accordingly.

However, I'd say that the database is probably a bad place to manage all the queueing in your system. Sure the database will need to handle some of it, but most databases I see handle much more queuing than necessary, with poor results for all.

Here is how it usually works:

1) Something changes in the database and causes a slight slowdown. Maybe an ad-hoc query is too expensive, or there is an issue with the network card. It does not need to be something huge.

2) Due to the small performance problem, queries take slightly longer to run, and there are more queries running at the same time. and therefore

3) At almost every single I've ever worked on, the application server notices that queries are taking longer to run and allocates more connections to the connection pool. This allows even more queries to run at once.

4) Now the database has to deal with more queries than usual on the slightly slow system, so the queries take even longer to run. They start queuing on the CPU, disks, library cache, and buffer cache.

5) As a result the application opens even more connections.

You can see how things quickly escalate—and when you are called on the scene, all you can see is the piles of sessions waiting in every possible queue. The original cause for the slowdown could have be solved by now, and in any case you can't see it in all the chaos. Remember, the database server only has N CPUs, unless your queries are expected to do a lot of IO, you'll gain no benefit by throwing much more than N queries at it. What do you think the database will do with them? Of course they will queue up and wait.

This escalating chaos scenario happened because the application let the database handle queues that should have been handled by the application.

Why by the application? Because the application has a better idea of which requests are more critical, and so it can prioritize accordingly. The application can decide that some requests can wait for later. It can return partial data to the customer and give a nice message that says "We are busy at the moment, please try again later." A database that is slightly slower will need its resources to recover from the problem. Or maybe you, the DBA, need database resources to diagnose and solve the problem, especially if the solution is "Add an index."

So you see, the database is slow because the application is not using it correctly. At least some of the time. ▲

Gwen Shapira is a senior DBA who spent the last 11 years of her life carrying a pager. She enjoys playing with data, presenting at conferences and telling people how to make their databases go faster. Luckily, she works for Pythian and gets paid to purse her favorite activities.

Achieving Extreme Performance with Oracle Exadata

A Book Review by Brian Hitchcock

Details

Author: Rick Greenwald, Robert Stackowiak, Maqsood Alam, Mans Bhuller

ISBN: 978-0-07-175259-6

Pages: 432

Publication Date: February 2011

Price: $60

Publisher: McGraw-Hill

Summary

Overall review: A very good introduction to Exadata.

Target Audience: Anyone with an Oracle database looking for performance improvement or database consolidation.

Would you recommend this to others? Yes.

Who will get the most from this book? Anyone who needs to understand what is new with Exadata.

Is this book platform specific? Yes—but that doesn't mean what it used to mean.

Why did I obtain this book? NoCOUG asked me to review this book.

Overall Review

Exadata is both a new thing and a return to the past. Dedicated hardware preconfigured to run specific vendor software—sounds like old times, doesn't it? But there are new twists. The use of large amounts of flash memory and clever software makes Exadata a very new thing. While the familiar Oracle database runs on Exadata, the way the software and hardware work together presents many new concepts and terms that need to be understood. Much about Exadata will be familiar to Oracle DBAs, but there is much that is totally new. This book covers the new concepts very well. While there aren't any case studies or metrics from real-world implementations, the material presented is well worth your time as you prepare for the new world of Exadata.

Before we go any further, we need to define some terms. The overall product is the Oracle Exadata Database Machine. This product contains multiple database server nodes, multiple storage server cells, and networking gear. The product also contains Exadata Storage Server Software that is distinct and separate from the Oracle database software. Taking the full rack Exadata configuration as an example, it contains eight database server nodes and 14 storage server cells. The terms "database node" (server) and "storage server" (cell) will come up often.

Chapter Reviews

- Introduction
- Chapter 1: Oracle and Tightly Integrated Hardware and Software Platforms
- Chapter 2: Oracle 11*g* Enterprise Edition Features
- Chapter 3: Exadata Software Features
- Chapter 4: Oracle Exadata Database Machine Platform Hardware Components
- Chapter 5: Managing the Exadata Database Machine
- Chapter 6: High Availability and Backup Strategies
- Chapter 7: Deploying Data Warehouses on the Oracle Exadata Database Machine
- Chapter 8: Exadata and OLTP
- Chapter 9: Consolidating Databases with the Oracle Exadata Database Machine
- Chapter 10: Migrating to the Exadata Database Machine

Introduction

While the introduction is very short, it makes several interesting points. First, that this is the first comprehensive book on the Oracle Exadata Database Machine. Second, that Exadata is all about simplicity.

> *Dedicated hardware preconfigured to run specific vendor software—sounds like old times, doesn't it? But there are new twists. The use of large amounts of flash memory and clever software makes Exadata a very new thing.*

And third, that Exadata "just works." These are interesting statements, and we will see how the rest of the book explains each of them.

Chapter 1: Oracle and Tightly Integrated Hardware and Software Platforms

We start with a brief history lesson. The evolution of Oracle database machines started in 2008. Perhaps that isn't a long enough period of time to really be called history, but there have been versions of this product from HP, Sun, and now Oracle and Sun. On the other hand, a timeline is shown for Oracle database development from 1977 that shows how this development has been coming for a long time. This also signals quite a change in focus for Oracle. Oracle has always been the database that runs on almost all combinations of server and storage platforms. With Exadata, Oracle is the database that runs on only one. This could be seen as "back to the future." This turn of events follows the endless increase in the volume of data that businesses need to process. Business also wants to access more and more data in less time to support more sophisticated data analysis. This pushes traditional database server and storage platforms beyond their limits.

Exadata addresses the need for a large move forward in database server performance to satisfy current demands. In addition to a change to dedicated hardware, Exadata also signals a change in the way customers build their systems. Up to now, each customer needed to have access to various personnel to decide what server and storage platforms they should select. Exadata eliminates this. The server hardware, software, and storage are defined for you. We are told that this eliminates risk for the customer. It also simplifies support for Oracle by greatly reducing the number of combinations of hardware and software that customers can have. Some will see this as a lack of choice, a lack of control, and for those who are employed to make these system design decisions, they may see it as a head-count reduction. The integration between the Exadata hardware and software and the Oracle database software is described. The addition of Smart Flash Cache and various new indexes (too choose a few at random) illustrates the many new things that Exadata brings to the table. Further, we are told that applications don't have to be certified for Exadata. Therefore, any application should run on Exadata without modification. I will call this transparency; i.e., moving to Exadata is transparent to your application. Your application users won't notice any difference. We will watch as this concept is further refined throughout the rest of the book. The first refinement is that the database for the application must be 11*g*R2. The impact of Exadata on the traditional DBA role is discussed. We are assured that all existing DBAs will simply be assigned to more interesting work. Finally, we have some thoughts on where we are headed next. Most (if not all) applications have complex middle tiers that also require ever-increasing levels of performance. The Exalogic platform will do for WebLogic and Java-based applications what Exadata is doing for databases. If you don't know what WebLogic is, consider this a wake-up call. Oracle's focus is moving quickly to the middle tier, and the future there is WebLogic.

Chapter 2: Oracle 11*g* Enterprise Edition Features

This chapter describes many of the features of the Oracle 11*g* Enterprise Edition database. This information is useful for readers who are approaching Oracle for the first time, but I think most experienced DBAs will not need 65 pages describing 11*g* features. For example, transactions and locks are explained. Having said this, there are pieces of useful information here and there, and this chapter does explain how each of the major features of 11*g* is enhanced by Exadata. We find that RAC is not a requirement for using Exadata. This means that your application doesn't have to be running on RAC before it can move to Exadata, but we are told that RAC and Exadata are made for each other. You probably want to use RAC for your application to get the most from the Exadata hardware and software. Note how the transparency is not as complete as we first thought. Our application must be running on 11*g*R2 database, and it should be set up for RAC.

> ***The way the software and hardware work together presents many new concepts and terms that need to be understood. Much about Exadata will be familiar to Oracle DBAs, but there is much that is totally new.***

Next is ASM. Exadata uses ASM exclusively. An existing application that doesn't use ASM can move to Exadata, but you will need to learn how to manage ASM. Another interesting bit of trivia is the Fast Recovery Area. (Yes, it has changed its name. It was the Flash Recovery Area.) The future is full of subtle changes. Partitioning is explained next, and we are told that Exadata really performs best when applications use partitioning. Parallel execution is described in great detail, as is Data Guard.

Again, this information isn't bad, but for most readers, I think this chapter should be skimmed for the parts that relate specifically to Exadata.

Chapter 3: Exadata Software Features

This chapter is the best part of the whole book. This is where we learn what is really new and different about Exadata. For all the interest in the Exadata hardware, it is the software that really offers some amazing performance improvements. Assuming your application can make use of all of them, the Exadata software really is a significant step forward. If you take away only one thing from this book (and/or this book review), it should be this: Exadata is hardware and software. And this software is separate from the Oracle database software. There is a lot of material here, and I have chosen to briefly describe each of the most significant software features. Again, this software is not the 11*g*R2 software. This software is called the Exadata Storage Server Software.

Smart Scan: During normal query processing, the database instance will examine all the blocks of a table that contain relevant data, and then eliminate data based on the query predicate. Smart Scan processes the blocks at the storage level, before they are returned to the database instance, and eliminates those that aren't needed to satisfy the query. Smart Scan also eliminates columns that aren't needed. All of this will dramatically reduce the amount of data that is returned to the database instance. Smart Scan also reduces the amount of data returned for each table that will be used in a join using Join Filtering. Using a Bloom filter to determine which rows will be needed in each table that will be joined; rows that won't be joined are eliminated, further reducing the amount of data that the database instance will have to deal with.

Exadata provides significant performance gains but will require significant organizational changes as well.

Note that the effects of Smart Scan can be seen in the explain plan that now displays new operations, such as TABLE ACCESS STORAGE FULL.

Encryption: Smart Scan can process encrypted data directly without the processing overhead of decryption.

Hybrid Columnar Compression: As mentioned earlier, the amount of data to be processed is always increasing. Compression reduces the amount of storage required, as well as reducing the I/O bandwidth. Exadata can store table data by columns. Only the columns needed for the query will be processed. The table is stored as a set of Compression Units, and each unit stores the data for one or more columns for a set of rows.

While the compression is done by the database instance, decompression can be done at the storage level.

Storage Indexes: These indexes track the high and low values for columns in the rows stored in 1Mb storage regions. The Exadata Storage Server Software can use these indexes to eliminate storage regions that aren't needed for a query. This, in my opinion, is similar to partition pruning in the database.

Smart Flash Cache: Exadata has large amounts of flash memory storage. To make the best use of this flash storage, Exadata provides software that will decide what to keep in flash cache. Data that it determines will be used again is cached.

Each of these software features has been described very briefly. Detailed descriptions of each are provided in this chapter. Also note that for each feature, there are conditions under which the feature won't apply or won't be used by Exadata. There are queries where some or all of these features won't be used or won't improve performance.

Chapter 4: Oracle Exadata Database Machine Platform Hardware Components

This chapter discusses the Exadata hardware in detail. The need for balance between the CPUs' ability to process data and the storage system's ability to supply that data is examined. This issue led to the fixed ratios seen between the database nodes and the storage server cells in the Exadata systems. It is interesting to read about how flash memory can wear out and how the Exadata hardware is designed to work around this by supplying 32Gb of flash storage for each 24Gb of flash cache that appears to be available. The evolution of the database server nodes and the storage server cells is shown for each of the iterations of the Exadata machine. And we are told that sizing the Database Machine should be done only by consulting Oracle. The CPU and storage capacity, as well as the physical dimensions of the various Exadata machines, are also presented.

Chapter 5: Managing the Exadata Database Machine

Now we learn about the tasks you will need to do after your Exadata machine is first installed. This includes creating databases and migrating databases from other systems. An interesting note appears, pointing out that the focus here is not on the administration of the database servers. We are told that the database servers are managed similarly to a regular 11*g*R2 database utilizing RAC and ASM. This implies that if you want to move your database(s) to Exadata, they must be on RAC and ASM, in addition to being on 11*g*R2. Details of how Exadata formats the disks (LUNs, cell disks, grid disks) and presents them to ASM are covered, as well as detailed options for flash cache. We also see a new command line utility, CellCLI, which is used to configure and manage the Exadata Storage Server Software. You now have a new utility to learn. Further, we are told that while we will have root access to the storage servers, performing any modifications to the storage servers is prohibited.

The practical impacts of all of this are amusing to think about. Will your Exadata machine be monitored or inspected? What will happen to those who violate this rule? It turns out that CellCLI works only within each of the multiple storage cells (servers) in the Exadata machine. To manage multiple cells, we use the dcli utility. Many more things are covered, including the overall Exadata setup process and security that allows creation of realms of disks for separate databases and monitoring.

Chapter 6: High Availability and Backup Strategies

We start out covering Data Guard. Exadata Hybrid Columnar Compression can cause problems if your standby server isn't also an Exadata machine. If the primary stores data using

Exadata flash memory capacity is designed to hold about 18% of capacity of the storage servers. This ratio has been found to represent the "working set," which is the fraction of the data in a database that is used by an application at any given time.

Oracle Database Administration, Development, and Performance Tuning... *Only Faster.*

Taking care of your company's data
is an important job.
Making it easier and faster is our's.

DATABASE ADMIN & DEVELOPMENT
PERFORMANCE TUNING & SQL OPTIMIZATION
DATABASE CHANGE MANAGEMENT
ORACLE

DB PowerStudio™

Introducing DB PowerStudio for Oracle. It provides proven, highly-visual tools that save time and reduce errors by simplifying and automating many of the complex things you need to do to take care of your data, and your customers.

Whether you already use Oracle Enterprise Manager (OEM) or some other third-party tool, you'll find you can do many things faster with DB PowerStudio for Oracle.

> **Easier administration** with DBArtisan®, the industry's most respected database administration tool with graphical editors and wizards to streamline routine tasks and reduce errors. In addition, database analytics identify performance, capacity, and storage management issues before they become problems

> **Faster performance** with DB Optimizer™, an agent-less graphical profiling tool that quickly pinpoints performance issues and provides a Visual SQL Tuner so you can quickly understand and optimize poor-performing SQL

> **Faster development** with Rapid SQL™, a full-featured IDE that streamlines SQL and PL/SQL scripting, query building, object management, and version control

> **Simplified change management** with DB Change Manager™, a tool for coordinating database changes across the development lifecycle, enforce database configuration standards, and mask test data to comply with privacy mandates

Best of all, DB PowerStudio tools work across all versions of Oracle from a single interface. And with Embarcadero's enterprise-friendly software licensing and delivery technologies, it's easier to access, track, and manage tools licenses to minimize costs and maximize productivity.

Don't forget Data Modeling! Embarcadero also provides **ER/Studio**®, the industry's best tool for collaborative data modeling.

Go Faster Today. >>> Get Free Trials and More at **www.embarcadero.com**

embarcadero®

EHCC, this data will be delivered to the standby in the compressed form. If you were to fail over, you would need to uncompress this data on the standby. We are told that the best practice for EHCC and Data Guard is to have an Exadata machine for both the primary and the standby. The use of Golden Gate log-based replication is also discussed.

Next is patching. Note that Oracle supplies all patches for all aspects of Exadata software. Users are prohibited from applying OS patches directly, for example. Note that this means the patching process for Exadata will be separate from your patching process for your non-Exadata systems. Backup and recovery are covered. It turns out that you don't back up the OS for the storage servers, and, if needed, you will restore from the supplied CELLBOOT USB flash drive. Who keeps this flash drive? Lots of new procedures to implement. RMAN is the only way to make database backups on Exadata. Again, if you aren't using RMAN now, you need to be ready to support it for Exadata. Oddly, I found the description of RMAN image copies and backup sets to be the best I've run across.

Chapter 7: Deploying Data Warehouses on the Oracle Exadata Database Machine

A review of data warehouse schema (stars, snowflakes) options and history precedes a discussion of how the features of Exadata help with data warehouse processing. We learn that there are actually more CPUs in the storage servers than in the database nodes, and that many math and analytic functions are processed within the storage servers. This is another example of how Exadata (hardware and software) breaks up data processing tasks to improve performance.

ROLTA
TUSC
Enterprise IT Solutions
The Value of Experience...
✓ Oracle (Applications & Technology)
✓ IT Infrastructure
✓ Profitability & Cost Mgmt
✓ Rolta Software Solutions
ORACLE Platinum Partner

Chapter 8: Exadata and OLTP

Similar to the previous chapter, here we learn how Exadata features help with OLTP processing. Not surprisingly, Exadata Smart Flash Cache has the most impact on OLTP. I didn't know that the Exadata flash memory capacity is designed to hold about 18% of capacity of the storage servers. This ratio has been found to represent the "working set," which is the fraction of the data in a database that is used by an application at any given time. This is how Oracle decided how much flash memory to supply, versus physical disk storage. The flash memory in the Exadata server is also protected against power supply issues that could result in a loss of data by an integrated supercapacitor! Not just a really big capacitor, but a supercapacitor! Perhaps it will take your Delorean back in time?

> ***While you don't have to use RAC on Exadata, the whole point of Exadata is having many database and storage nodes to share workload.***

Chapter 9: Consolidating Databases with the Oracle Exadata Database Machine

Much of the marketing around Exadata emphasizes its sheer performance.

But it also has a lot of capacity. While many customers may have huge databases that can fill one or more Exadata racks, many customers may have many smaller databases that could be consolidated into a single Exadata machine. This chapter tells us that we can consolidate OLTP and data warehouse databases into a single Exadata machine but we have to be careful. Also, if your existing application is CPU bound, consolidating into Exadata needs to be done with care. Specifically, we are told that the CPU sizing process is critical when consolidating CPU-bound applications. Memory and I/O sizing are covered, as well as how to isolate databases from each other for security or performance reasons. While reading this book, I wasn't clear on whether RAC was a requirement of Exadata. In other words, I didn't know if my existing application had to be running on RAC to be ready to move to Exadata. In this chapter, this is discussed. The point is that while you don't have to use RAC on Exadata, the whole point of Exadata is having many database and storage nodes to share workload. If your application uses a single database node in the Exadata machine, you may not be realizing the full potential offered by the Exadata hardware and software.

Chapter 10: Migrating to the Exadata Database Machine

This chapter covers the practical issues around actually migrating Oracle and non-Oracle databases to Exadata—things like how to gather metrics for the existing databases,

(continued on page 18)

Teamwork at NoCOUG

(PHOTO COURTESY OF CALCYCLING)

NoCOUG is a successful organization with more than 500 members, and there's no way it could run without teamwork. We have a full and active Board of Directors, plus other volunteers who contribute regularly. All the people on the NoCOUG team contribute in both big and small ways, depending on what they have time for. And it's all of us working together as a team that makes for the great conferences, training days, and other benefits.

But volunteering your time is far from without rewards. In fact, volunteering with NoCOUG offers opportunities to meet and talk with speakers, authors, and other professionals in the Oracle field, as well as other activities. In fact, if your day-to-day job has become routine or doesn't offer you the chance to use some of your other skills—interacting with people, writing, organizing events, etc.—volunteering is a great way to utilize those skills. It's surprisingly fun once you get started. You'll find we are a welcoming bunch of people, and most volunteers say their favorite aspect of volunteering is the people they meet. So, if you would like to get involved but don't know where to start, here are some quick things you can do that don't take much time:

- Contribute an article to the *NoCOUG Journal*
- Volunteer to speak and share your knowledge at a conference
- Recruit a knowledgeable Oracle colleague to speak at a conference or contribute an article
- Help with registration on the day of our quarterly conference
- Assist with marketing our conferences and training days

And, there are plenty of other opportunities to help out. Remember, it takes a lot of teamwork to keep our successful organization growing and providing value to its members. So, if you want to be part of a great team, just send an email to **contact_nocoug@nocoug.org** and let us know how you want to get involved.

What are you waiting for. Join the NoCOUG Team! ▲

"I have met many wonderful people during my years of volunteering. Some long-term friendships have developed. Every quarter brings new friends. This is what I enjoy about NoCOUG volunteering.

"A big reason for volunteering is the give-back factor. I have received a great deal of valuable experience as a NoCOUG member. I feel it's important to give something back to the organization."

—Joel Rosingana
NoCOUG Past President

"Volunteering with NoCOUG helped me round out my professional skill set by augmenting my technical skills with the 'soft' skills that I think are so very important. During these last eight years, I have held a variety of positions ranging from webmaster to president and I've learned a lot that I would never have learned just being an Oracle DBA or application developer. I've learned a lot about working with people, about how the whole is so much more than the sum of the parts, about how to get things done that are too big to do all by myself."

—Roger Schrag
NoCOUG Past President

If you are using snapshots for backups, Exadata doesn't support anything like this. You can't connect a third-party storage system to Exadata to support snapshots. You must transition to RMAN for backups.

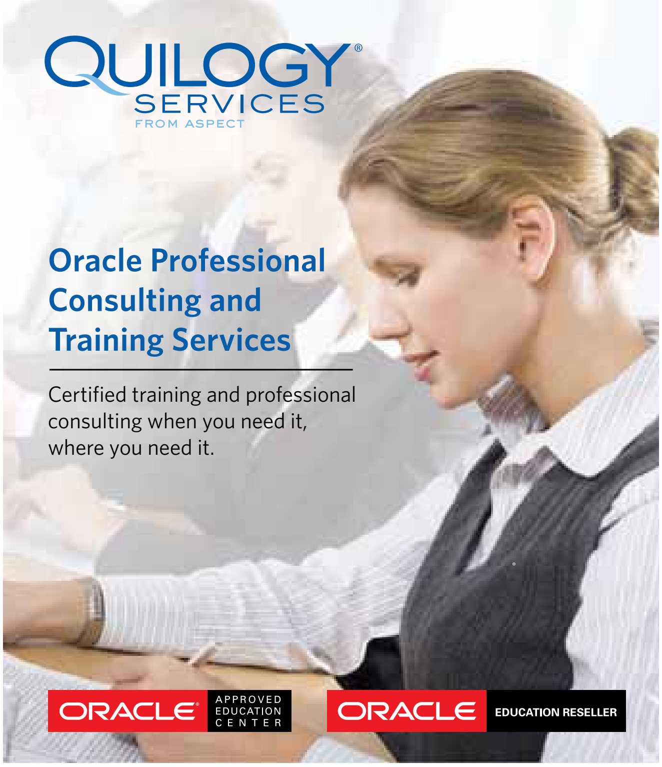
(continued from page 16)

documenting any data loads and extracts, how to take into account the various compression options offered by Exadata when sizing disk capacity, and so on. I found some interesting things here. We are told that the process of migrating to Exadata is no different from migrating to a regular 11*g*R2 database on Linux or Solaris configured with RAC and ASM. Depending on what your current configuration is, this migration could be very complex whether Exadata is involved or not. If you are using snapshots for backups, Exadata doesn't support anything like this. You can't connect a third-party storage system to Exadata to support snapshots. You must transition to RMAN for backups. When sizing the new environment, you should size for performance and then for capacity. Sizing for performance is harder, and many times only the sizing for capacity is done. It is easy to calculate the disk space of your existing system. However, even this is not as easy as it seems, since you can compress your data on Exadata and you may need a lot less disk space than you have now.

Conclusion

This book is very good at presenting the Exadata Database Machine. Assuming you are familiar with recent versions of the Oracle database, I would suggest you focus on chapters 3, 4, and 5. These contain most of the discussion of what is really new about Exadata and how it works. There are two topics that I think need more coverage for those considering a move to Exadata. First is what I have been calling transparency. Will your existing application run on Exadata without any changes, i.e., will the move to Exadata be transparent to your applications and your users? Let's assume this is true. However, to be able to move to Exadata, your applications have to run on 11*g*R2. If you aren't on 11*g*R2 already, this migration could be a large project. Further, Exadata requires that you support ASM and RMAN. This could be another migration. Finally, to get the most from Exadata, your applications should make use of partitioning and RAC. Yet another migration? It may well be true that your existing databases could run Exadata without any modifications, once they were "modified" to run on 11*g*R2, ASM, RAC, etc.

To move to Exadata, your applications have to run on 11gR2 . . . to get the most from Exadata, your applications should make use of partitioning and RAC.



Second is the support organization. Who will support ASM, Exadata storage cells, CellCLI, and dcli utilities? Will this be handled by your existing DBA staff or the SA staff? Who will support the Exadata patching process that spans OS, storage, and database? We normally patch development, then test, and then do production. What does this mean for Exadata? While we could create three databases in the Exadata machine, how about patches that affect the Exadata machine overall? Do we no longer need development and test environments to validate these high-level patches? Exadata provides significant performance gains but will require significant organizational changes as well. ▲

The statements and opinions expressed here are my own and do not necessarily represent those of Oracle Corporation.

Brian Hitchcock worked for Sun Microsystems for 15 years supporting Oracle databases and Oracle Applications. Since Oracle acquired Sun he has been with Oracle supporting the On Demand refresh group and most recently the Federal On Demand DBA group. All of his book reviews, presentations and his contact information are available at **http://www.brianhitchcock.net**.

SERVING UP THE BEST IN DATA INTEGRATION

DBMoto™ delivers TOP PERFORMANCE in Change Data Capture and Data Integration for the most demanding tasks, without stressing your system or budget!

HiT® SOFTWARE

A BackOffice Associates, LLC Company

DBMoto VERSION 7 · *NEW! INFORMIX LOG-READING!*

DBMoto 7 offers Change Data Capture, transaction log-reading for:

Oracle, Informix, MSFT SQL Server, DB2, MySQL, and others!

- Multi-Server Synchronization
- Easy administration, quick to deploy
- DBMoto Verifier™ "compare" tool to assess replication results
- Customizable functions and automated "smart" management features
- Cost-effective – use your own IT team!

If you need quick, reliable, low-cost data integration between Oracle and other relational databases or a Netezza appliance, DBMoto is the most efficient solution available today!

FREE WHITE PAPER!

"Data Integration Cost Considerations"

www.hitsw.com/WP19L

DATABASES SUPPORTED:

Oracle · IBM DB2 for i, z/OS, LUW · Microsoft SQL Server · MySQL · Netezza · Informix · Sybase ASE · SQL Anywhere · Ingres · Cloudscape · PostgreSQL

Evaluate DBMoto for FREE! Download a fully functional version at **www.hitsw.com/DL19A**

T +1.408.345.4001 www.hitsw.com info@hitsw.com

Multi Dimensional Clustering (MDC)

by Iggy Fernandez

Clustering is even more fundamental to database performance than indexes. Imagine a huge library or bookstore with an up-to-date catalog that allowed you to pinpoint the location of any book in stock. Suppose that you wanted to find all the books on Oracle published by Apress but they were scattered on random bookshelves throughout the store. You wouldn't patronize that library or bookstore for very long. In fact, poor data clustering is a big reason why the Oracle optimizer may decline to use an index.

To demonstrate the value of clustering, consider a fact table with three independent dimensions—year, region, and color. Let's generate 500,000 records that we can reuse multiple times in the course of our experiment and insert them into the fact table. Finally, let's create an index on the fact table and collect optimizer statistics.

```
CREATE TABLE data
(
  pk integer PRIMARY KEY,
  year integer NOT NULL,
  region integer NOT NULL,
  color integer NOT NULL,
  sales number NOT NULL
);

INSERT INTO data
SELECT
  level AS pk,
  ceil(dbms_random.value*10) AS year,
  ceil(dbms_random.value*10) AS region,
  ceil(dbms_random.value*10) AS color,
  dbms_random.value*10000 AS sales
FROM dual
CONNECT BY level <=500000;

CREATE TABLE fact
(
  pk integer PRIMARY KEY,
  year integer NOT NULL,
  region integer NOT NULL,
  color integer NOT NULL,
  sales number NOT NULL
);

INSERT INTO fact
(pk, year, region, color, sales)
SELECT * FROM data;

CREATE INDEX fact_i1 ON
fact(year, region, color);
EXEC dbms_stats.gather_table_stats(
  'IGGY', 'FACT');
```

Now let's execute a simple query that specifies the value of all three dimensions. We see that Oracle requires almost as many consistent gets (459) as the number of records (501), meaning that, on average, each data block visited by Oracle held only one record of interest, a terrible waste of effort. The situation is analogous to the badly organized library or bookstore that I mentioned earlier. (In all the examples in this article, each query was run twice to eliminate the overhead of parsing but only the results of the second execution are reported.)

```
set autotrace on
SELECT count(*), sum(sales) FROM fact
WHERE year=10 AND region=10 AND color=10;

COUNT(*)    SUM(SALES)
----------  ----------
501         2375907.81

Execution Plan
-------------------------
SELECT STATEMENT
   SORT AGGREGATE
         TABLE ACCESS BY INDEX ROWID FACT
               INDEX RANGE SCAN FACT_I1

Statistics
-------------------------
459 consistent gets
```

Next, let's redo the exercise from the beginning but, this time around, let's insert the data in sorted order.

```
INSERT INTO fact
(pk, year, region, color, sales)
SELECT * FROM data
ORDER BY year, region, color;
```

The results are magical. The query plan does not change but consistent gets are reduced from 459 to 8 because every block Oracle touched now holds many records of interest. This is precisely the clustering effect we were looking for.

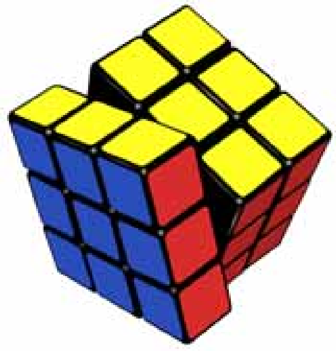
```
SELECT count(*), sum(sales) FROM fact
WHERE year=10 AND region=10 AND color=10;

COUNT(*)    SUM(SALES)
----------  ----------
501         2375907.81

Execution Plan
-------------------------
SELECT STATEMENT
   SORT AGGREGATE
         TABLE ACCESS BY INDEX ROWID FACT
               INDEX RANGE SCAN FACT_I1
```

```
Statistics
-------------------------
8 consistent gets
```

However, the clustering effect breaks down quite a bit when we specify only one of the dimensions. Suppose that we need to retrieve the sales total for the color 10. The average number of rows per block works out to be 181. There are 49723 records for color 10 and therefore we would only need 274 consistent gets in the ideal case plus some more to read the index. However, we find that we need to read 612 consistent gets, substantially more than in the ideal case. Note also that the index access path changes to INDEX SKIP SCAN.

```
SELECT 500000 / COUNT(DISTINCT(
  dbms_rowid.rowid_block_number(rowid)
)) AS avg_rows
FROM fact;

AVG_ROWS
----------
181.093807

SELECT count(*), sum(sales) FROM fact
WHERE color = 10;

COUNT(*)    SUM(SALES)
----------  ----------
49723       247646618

Execution Plan
-------------------------
SELECT STATEMENT
   SORT AGGREGATE
         TABLE ACCESS BY INDEX ROWID FACT
               INDEX SKIP SCAN FACT_I1
Statistics
-------------------------
612 consistent gets
```

Enter Multi Dimensional Clustering (MDC). The goal of MDC is to tightly cluster all rows with the same values of the dimension columns. Each of the dimensions in the above example can have ten different values. Imagine a 10x10x10 cube made up of 1x1x1 cells. All the rows in a cell should have the same values of all three dimension columns. All ten cells in any row or column of cells should have the same values for exactly two of the dimension columns. All one hundred cells in any horizontal or vertical slice of cells should have the same values of just one of the dimension columns.



Picture of an almost solved Rubik's Cube by Wikipedia Contributor Kleiner

A "cell" is therefore a collection of database blocks tracked by a "block index." A block index is also called a "sparse index" because it does not contain any entry for each row in the table; instead it contains one row for every block and is therefore much smaller than a regular index. When a query visits a block it will only find rows that satisfy the query criteria. Thus, the query only has to visit the minimum possible number of blocks.

> ***If a schema has no IOTs or clusters, that is a good indication that no thought has been given to the matter of optimizing data access.***

Here's a demonstration of MDC in action. First we create a table cluster. (One use of a table cluster is to co-locate rows from multiple tables that have the same value of the cluster key; Oracle uses such clusters in the data dictionary and in its TPC-C benchmarks. However, a table cluster can also provide the multidimensional clustering we are looking for.) We then add a table to the cluster and insert 500,000 rows into it. Each dimension can have ten possible values.

```
-- Create a multidimensional cluster
CREATE CLUSTER mdc
(
  year integer,
  region integer,
  color integer
);

-- create the required "block index"
CREATE INDEX mdc_idx ON CLUSTER mdc;

-- Add the fact table to the cluster
CREATE TABLE mdc_tab
(
  pk integer PRIMARY KEY,
  year integer NOT NULL,
  region integer NOT NULL,
  color integer NOT NULL,
  sales number(8,2) NOT NULL
)
CLUSTER mdc (year, region, color);

-- Insert 500,000 rows into the fact table
INSERT INTO mdc_tab
(pk, year, region, color, sales)
SELECT * FROM fact;
COMMIT;

-- Collect optimizer statistics
EXEC dbms_stats.gather_table_stats(
  'IGGY', 'MDC_TAB');
```

(continued on page 22)

> ***Clustering is even more fundamental to database performance than indexes. In fact, poor data clustering is a big reason why the Oracle optimizer may decline to use an index.***

The goal of Multi Dimensional Clustering is to tightly cluster all rows with the same values of the dimension columns.

(continued from page 21)

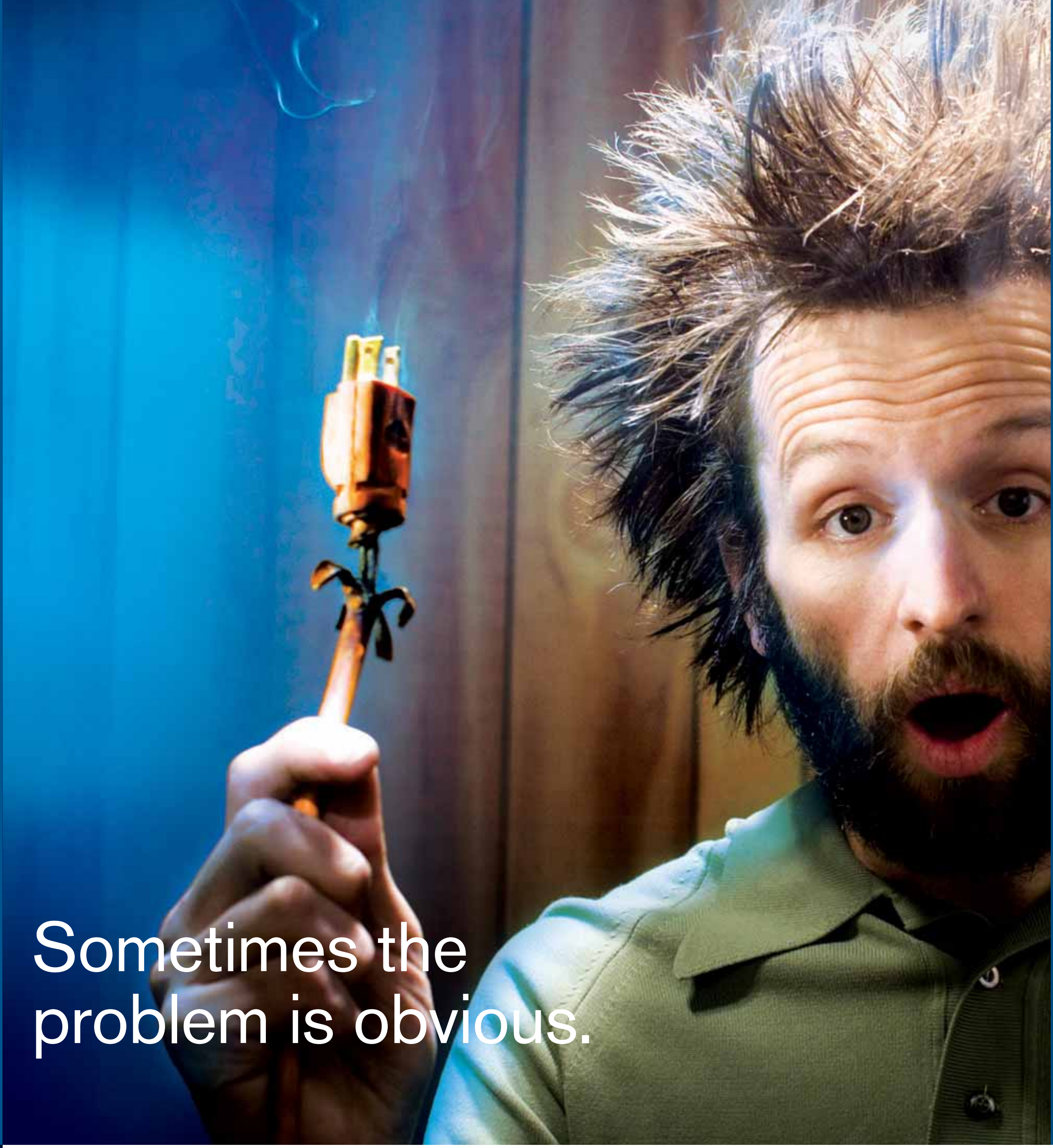
The following traces show that Oracle efficiently uses the block index to visit the minimum number of blocks. Irrespective of the dimension, Oracle only needs to visit about 200 blocks. Notice that the table access path is TABLE ACCESS CLUSTER.

```
SELECT COUNT(*), SUM(sales) FROM mdc_tab
WHERE year=10;

COUNT(*)    SUM(SALES)
----------  ----------
50058       248952500

Execution Plan
-------------------------
SELECT STATEMENT
  SORT AGGREGATE
      TABLE ACCESS CLUSTER MDC_TAB
          INDEX RANGE SCAN MDC_IDX

Statistics
-------------------------
201 consistent gets

SELECT COUNT(*), SUM(sales) FROM mdc_tab
WHERE region=10;
```

```
COUNT(*)    SUM(SALES)
----------  ----------
50010       250669898
Statistics
-------------------------
205 consistent gets

SELECT count(*), sum(sales) FROM mdc_tab
WHERE color = 10;

COUNT(*)    SUM(SALES)
----------  ----------
49723       247646618

Statistics
-------------------------
203 consistent gets
```

Index Organized Tables (IOTs) are another good way to cluster data. In *Effective Oracle by Design*, Tom Kyte quotes Steve Adams as saying: *"If a schema has no IOTs or clusters, that is a good indication that no thought has been given to the matter of optimizing data access."* ▲

Iggy Fernandez is the author of Beginning Oracle Database 11g Administration *[Apress, 2009] and* Expert Indexing for Oracle Database 11g *[Apress, 2011]. He blogs at* **http://www.iggyfernandez.wordpress.com**.


Sometimes the
problem is obvious.

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 memberships. If your company is able to offer sponsorship at any level, please contact NoCOUG's president, Iggy Fernandez, at **iggy_fernandez@hotmail.com**. ▲

Long-term event sponsorship:

CHEVRON

ORACLE CORP.

Thank you! Year 2011 Gold Vendors:

- 3PAR
- Confio Software
- Database Specialists, Inc.
- Delphix
- Embarcadero Technologies
- Rolta TUSC

For information about our Gold Vendor Program, contact the NoCOUG vendor coordinator via email at: **vendor_coordinator@nocoug.org**.

TREASURER'S REPORT

Naren Nagtode, *Treasurer*

Beginning Balance		
January 1, 2011		**$ 40,529.29**
Revenue		
Membership Dues	22,310.00	
Meeting Fees	295.00	
Vendor Receipts	3,250.00	
Advertising Fee	–	
Training Day	11,550.00	
Sponsorship	–	
Interest	4.76	
Paypal balance	–	
Total Revenue		**$ 37,409.76**
Expenses		
Regional Meeting	700.00	
Journal	750.00	
Membership	550.07	
Administration	1,466.25	
Website	–	
Board Meeting	549.23	
Marketing	–	
Insurance	–	
Vendors	–	
Tax	–	
Training Day	1,163.33	
IOUG-rep	–	
Miscellaneous	–	
Total Expenses		**$ 5,178.88**
Ending Balance		
March 31, 2011		**$ 72,760.17**

NoCOUG Spring Conference

Session Descriptions

For the most up-to-date information, please visit **http://www.nocoug.org**.

–Keynote–

Aspiring for Excellence in Database Performance

Andrew Holdsworth, Oracle Corp. 9:30–10:30

The theme for this keynote is to "Aim High." During this presentation, Andrew Holdsworth, Senior Director, Oracle Real World Performance Group, will provide updates and insights into how Oracle Database professionals can exploit the full potential of the Oracle Database and Exadata Hardware combination.

To date, many customers have successfully implemented Oracle Database on Exadata Hardware to achieve order of magnitude performance gains when running existing applications. During this discussion, Andrew will overview the true potential of the Software/Hardware platform—when used optimally and creatively—to produce innovative solutions to old and new data processing problems.

The best practices content delivered today will hopefully inspire Oracle professionals to aspire for excellence, or "Aim High" when solving database challenges rather than applying the standard industry approach of "good enough"—because "good enough" is rarely acceptable tomorrow. This is the philosophy of Oracle's Real World Performance Group.

–Auditorium–

Sane SAN 2011

James Morle, Scale Abilities 11:00–12:00

It has been ten years since the publication of my original Sane SAN whitepaper and much has changed in the area of high performance database storage. In this major overhaul of the original topics of Sane SAN 2000 we look at the pros and cons of the various technologies available on the market today, and at approaches that maximise the performance of the storage tier and deliver predictable response times to the database. We also look at emerging trends in an attempt to steer new architectures in a direction that is compatible with the imminent paradigm shift in persistent storage driven by solid-state storage.

Oracle SQL Developer Datamodeler

Kris Rice, Oracle. 1:00–2:00

Oracle SQL Developer Data Modeler is a feature-rich product supporting logical, relational, multi-dimensional, data type and data flow models, and physical data modeling for Oracle and non-Oracle databases. In this session we follow a flow of work, starting from scratch with an empty model and touching all aspects of the product, discussing how the models relate to each other and the salient points of each. The latest release is tightly integrated with the open source, source code control software, Subversion, allowing users to track changes to models and objects. SQL Developer Data Modeler 3.0 also includes a number of other new features including the ability to create user defined design rules and transformation scripts. These and other new features will be demonstrated in this session.

PHP: Develop and Deploy Mission—Critical Applications with Oracle DB 11*g*

Luxi Chidambaran, Oracle 2:30–3:30

This session will illustrate how PHP, together with Oracle Database 11*g* are being used to build mission critical enterprise and web applications. The session will start with basics of PHP connectivity to Oracle. We will then describe how customers leverage Oracle Database Resident Connection Pooling (DRCP) to handle the demanding scalability requirements of web facing PHP applications. The session will also touch upon key facilities in PHP that Oracle provides for HA, online application upgrades, result caching, and end-to-end tracing.

Getting Started with Oracle and .NET

Riaz Ahmed, Oracle. 4:00–5:00

This beginner-level session will introduce Oracle's offerings for .NET programmers, including Oracle Data Provider for .NET (ODP.NET), Oracle Developer Tools for Visual Studio, Oracle Providers for ASP.NET, and .NET stored procedures. Step-by step-demos will be used to illustrate how to get started developing Oracle Database .NET applications using each of these free products. New and upcoming features will also be described briefly in this session, including Entity Framework, LINQ, WCF Data Services, 100% managed Thin ODP.NET, and support for TimesTen.

–Room 102–

Edition-Based Redefinition: the Key to Online Application Upgrade

Bryn Llewellyn, Oracle 11:00–12:00

Large, mission-critical applications built on Oracle Database are often unavailable for tens of hours while the application's database objects are patched or upgraded. Oracle Database 11*g* Release 2 introduces revolutionary new capabilities that allow online application upgrade with uninterrupted availability of the application. Existing sessions can continue to use the pre-upgrade application until their users decide to finish; and, at the same time, new sessions can use the post-upgrade application. When no sessions are any longer using the pre-upgrade application, it can be retired. The application as a whole there-

(continued on page 26)

(continued from page 25)

fore enjoys hot rollover from the pre-upgrade version to the post-upgrade version.

The capability depends on these new kinds of object: the edition, the editioning view, and the crossedition trigger. Code changes are installed in the privacy of a new edition. Data changes are made safely by writing only to new columns or new tables not seen by the old edition. An editioning view exposes a different projection of a table into each edition to allow each to see just its own columns. A crossedition trigger propagates data changes made by the old edition into the new edition's columns, or (in hot-rollover) vice-versa. The capability as a whole is called edition-based redefinition—EBR for short. This session explains how it all works.

Data Warehouse Best Practices

Rekha Balwada, Oracle 1:00–2:00

Attend this session to learn best practices for deploying a data warehouse on Oracle Database 11*g*. Using detailed examples, this session will cover the best practices for schema design, hardware configuration, data loading, partitioning strategies, parallel query, and workload management. You will also learn how to monitor, diagnose, and correct common performance issues using both the old reliable tools such as explain plan as well as an introduction to the latest and greatest SQL monitor tool. By applying what you learn in this session you will be able to deploy a data warehouse that can seamlessly scale without constant tuning or tweaking.

Exadata—Embracing Change—What is familiar and what is new?

Brian Hitchcock, Oracle. 2:30–3:30

The Exadata Database Machine offers new levels of processing performance and a new set of concepts to learn. Based on my book review of "Achieving Extreme Performance with Oracle Exadata" (Oracle Press), this presentation covers what is new and what is familiar to an Oracle DBA. Due to the combination of hardware and software that make up the Exadata machine, there are many new things to learn including the storage cells, a new command line interface and the requirements a database must meet before it can be migrated to Exadata.

Goldengate Replication—Myth vs Reality

Jim Allen, Yodlee Inc. 4:00–5:00

Replication seems like the silver bullet to solve scalability, reliability and geographic issues. Oracle has deprecated Streams and acquired Goldengate as the preferred solution for hot-hot multimaster replication. However there are systemic considerations for multimaster systems in general and the current state of the art for Goldengate in particular that need to be considered from an application perspective before committing to this replication technology. To replicate DDL or not, partitioning of sequences and data domains geographically and the impact of replication lags especially with RAC are some of the issues explored in this talk. Learn which systems this powerful tool is best applied to and when the operational complexities make it problematic.

–Room 103–

RMAN 101: An Introduction to RMAN

Abhaid Gaffoor, Amazon. 11:00–12:00

In this session we will cover the basics of RMAN. We will look at the architecture of RMAN and basic backup and recovery guides. We will also look at the use of an RMAN catalog. In closing we will mention a few tips and tricks when using RMAN. At the end of the session you should have a basic understanding of RMAN and be able to do basic backup and recovery with your Oracle databases.

Advanced RAC troubleshooting

Riyaj Shamsudeen .1:00–2:00; 2:30–3:30

These presentations explore and troubleshoot performance issues encountered in RAC environments. Below are a few discussion points:

1. Trend analysis for global cache traffic and performance using AWR/statspack tables. Review of simple techniques to probe interconnect performance.
2. Understanding wait events in SQL trace files related to RAC such as gc cr grant 2-way, gc cr grant 3-way, gc multi block reads, etc.
3. Global cache object re-mastering.
4. Global cache waits and LGWR performance. LGWR performance is important in single instance databases and LGWR performance is even more critical in RAC instances.
5. Interconnect issues, lost packets, and statistics. Identifying interconnect issues with database and OS statistics.
6. Troubleshooting performance issues with MTU/jumbo frames, and jumbo frames mis-configuration issues.
7. Troubleshooting RAC background processes such as LMS/LMD, etc.
8. Troubleshooting parallel query, interconnect, and PQ setup related performance issues.
9. Concurrency-induced global cache issues and a few time-tested simple work-arounds.
10. Troubleshooting locking issues and deadlocks in RAC environments.
11. Bind peeking issues and RAC instances.

OMG! Identifying and Refactoring Common SQL Performance Anti-Patterns

Editor's Pick

Jeff Jacobs 4:00–5:00

Most presentations on SQL performance assume that the query developer is highly experienced in SQL and Oracle and focus on tracing and other technique. Sadly, in today's world, knowledge of SQL and Oracle are not highly valued, and many SQL performance problems are the result of inexperience, misguidance, "heard it/read it somewhere" and various source of mis-information. This presentation will address common "anti-patterns" in database design and SQL query writing that I've encountered over the years and how to fix them without tracing, changing init.ora parameters or techniques requiring DBA level privileges. It is focused on techniques that are typically available to developers. ▲

Database Specialists: DBA Pro Service

DBA PRO BENEFITS

- *Cost-effective and flexible extension of your IT team*
- *Proactive database maintenance and quick resolution of problems by Oracle experts*
- *Increased database uptime*
- *Improved database performance*
- *Constant database monitoring with Database Rx*
- *Onsite and offsite flexibility*
- *Reliable support from a stable team of DBAs familiar with your databases*

CUSTOMIZABLE SERVICE PLANS FOR ORACLE SYSTEMS

Keeping your Oracle database systems highly available takes knowledge, skill, and experience. It also takes knowing that each environment is different. From large companies that need additional DBA support and specialized expertise to small companies that don't require a full-time onsite DBA, flexibility is the key. That's why Database Specialists offers a flexible service called DBA Pro. With DBA Pro, we work with you to configure a program that best suits your needs and helps you deal with any Oracle issues that arise. You receive cost-effective basic services for development systems and more comprehensive plans for production and mission-critical Oracle systems.

DBA Pro's mix and match service components

Access to experienced senior Oracle expertise when you need it

We work as an extension of your team to set up and manage your Oracle databases to maintain reliability, scalability, and peak performance. When you become a DBA Pro client, you are assigned a primary and secondary Database Specialists DBA. They'll become intimately familiar with your systems. When you need us, just call our toll-free number or send email for assistance from an experienced DBA during regular business hours. If you need a fuller range of coverage with guaranteed response times, you may choose our 24 x 7 option.

24 x 7 availability with guaranteed response time

For managing mission-critical systems, no service is more valuable than being able to call on a team of experts to solve a database problem quickly and efficiently. You may call in an emergency request for help at any time, knowing your call will be answered by a Database Specialists DBA within a guaranteed response time.

Daily review and recommendations for database care

A Database Specialists DBA will perform a daily review of activity and alerts on your Oracle database. This aids in a proactive approach to managing your database systems. After each review, you receive personalized recommendations, comments, and action items via email. This information is stored in the Database Rx Performance Portal for future reference.

Monthly review and report

Looking at trends and focusing on performance, availability, and stability are critical over time. Each month, a Database Specialists DBA will review activity and alerts on your Oracle database and prepare a comprehensive report for you.

Proactive maintenance

When you want Database Specialists to handle ongoing proactive maintenance, we can automatically access your database remotely and address issues directly — if the maintenance procedure is one you have pre-authorized us to perform. You can rest assured knowing your Oracle systems are in good hands.

Onsite and offsite flexibility

You may choose to have Database Specialists consultants work onsite so they can work closely with your own DBA staff, or you may bring us onsite only for specific projects. Or you may choose to save money on travel time and infrastructure setup by having work done remotely. With DBA Pro we provide the most appropriate service program for you.

DatabaseSpecialists

CALL 1-888-648-0500 TO DISCUSS A SERVICE PLAN

NoCOUG
P.O. Box 3282
Danville, CA 94526

NoCOUG Spring Conference Schedule

May 19, 2011—Oracle Conference Center in Redwood Shores

Please visit **http://www.nocoug.org** for updates and directions, and to submit your RSVP.
Cost: $50 admission fee for non-members. Members free. Includes lunch voucher.

8:00 a.m.–9:00	Registration and Continental Breakfast—Refreshments served
9:00–9:30	**Welcome:** Iggy Fernandez, NoCOUG president
9:30–10:30	**Keynote:** *Aspiring for Excellence in Database Performance*—Andrew Holdsworth, Oracle
10:30–11:00	**Break**
11:00–12:00	**Parallel Sessions #1**
	Auditorium: *Sane SAN 2011*—James Morle, Scale Abilities
	Room 102: *Edition-Based Redefinition: the Key to Online Application Upgrade*—Bryn Llewellyn, Oracle
	Room 103: *RMAN 101*—Abhaid Gaffoor, Amazon
12:00–1:00 p.m.	**Lunch**
1:00–2:00	**Parallel Sessions #2**
	Auditorium: *Oracle SQL Developer Datamodeler*—Kris Rice, Oracle
	Room 102: *Data Warehouse Best Practices*—Rekha Balwada, Oracle
	Room 103: *Advanced RAC troubleshooting*—Riyaj Shamsudeen
2:00–2:30	**Break and Refreshments**
2:30–3:30	**Parallel Sessions #3**
	Auditorium: *PHP: Develop and Deploy Mission-Critical Applications with Oracle DB 11g*—Luxi Chidambaran, Oracle
	Room 102: *Exadata—Embracing Change—What is familiar and what is new?*—Brian Hitchcock, Oracle
	Room 103: *Advanced RAC troubleshooting*—Riyaj Shamsudeen
3:30–4:00	**Raffle**
4:00–5:00	**Parallel Sessions #4**
	Auditorium: *Getting Started with Oracle and .NET*—Riaz Ahmed, Oracle
	Room 102: *Goldengate Replication—Myth vs Reality*—Jim Allen, Yodlee Inc.
	Room 103: *OMG! Identifying and Refactoring Common SQL Performance Anti-Patterns* **Editor's Pick**—Jeff Jacobs
5:00–	**Networking and No-Host Happy Hour at Sofitel Hotel, 223 Twin Dolphin Drive, Redwood City**

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

The *NoCOUG Journal* design and production: Giraffex, Inc., S.F.