Official Publication of the Northern California Oracle Users Group

NoCOUG JOURNAL

Vol. 21, No. 2 · MAY 2007 $15

Let Your Knowledge Blossom at NoCOUG

Big discount on Oracle Performance Training Day! See page 26.

Fireside Chat with Craig Shallahamer

See page 4.

Why Oracle?

Four well-known Oracles answer our questions. *See page 11.*

SQL Challenge

Try your hand at the SQL challenge and win an iPod Shuffle. *See page 18.*

Much more inside . . .

Teamwork at NoCOUG

You may have noticed that the *Journal* now sports a more colorful appearance. The *Journal* is now full color throughout, thanks to Darrin Swan, our Vice President, who negotiated a great deal with our new printer.

It does take a lot of teamwork to produce the *Journal*, beginning with the authors who generously give of their time and effort and the vendors who do a quality job. Lisa Loper, our President, writes the Presidential Message; Roger Schrag, our Director of Conference Programming, provides the conference agenda and abstracts; Diane Lee, our Vendor Coordinator, obtains artwork from the advertisers; Joel Rosingana, our Director of Membership, provides the member address list; and Nora Rosingana, our Staff Accountant, pays the vendors.

You too can help produce the Journal. *We are looking for additional technical reviewers to join the newly formed technical review team of Ted Syrett and Danny Chow. We are also looking for somebody to contribute book reviews on a regular or semi-regular basis. If you would like to help, please drop a line to* **journal@nocoug.org**.

—Iggy Fernandez, *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 2007 issue is May 31, 2007. Article submissions should be made in Microsoft Word format via email.

Challenges!

by Lisa Loper

People seek a challenge just as fire seeks to flame, says a Chinese proverb. Every spring, my tennis teammates and I gear up for the competitive league season. I'm always a little nervous going into every match, but I do it because I enjoy putting myself into a challenging situation and coming through to the other side.

Challenges—whether we choose them or not—help us push ourselves, tap into our resources, and develop confidence in what we can achieve. Best of all, when we face challenges, we gain a sense of accomplishment and the courage to take on the next challenge.

Life in the Bay Area can be overflowing with challenges, with its limitless variety of activities, industries, and professional, social, and environmental causes to contribute to. As Oracle professionals, chances are that your workload is a challenge in itself. You might be pulling more than your share of weight in your work environment, dealing with the pressures of seamlessly building and supporting mission-critical database systems, and more.

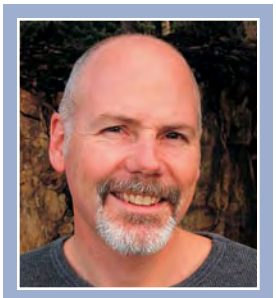
But if your work didn't have challenges, it would likely get boring really quickly. Without challenges and the opportunity to learn new things, people tend to stagnate. That's probably one of the reasons I took on the role of NoCOUG president. Sure, I have plenty to do in my regular professional life, but I wanted the opportunity to learn something new, to continue contributing to our Oracle Users Group, and to see how I handle the challenge and responsibility of looking at the big picture. With recent past presidents like Joel Rosingana, Roger Schrag, and Darrin Swan, I have big shoes to fill. In these first few months as president, I am definitely being challenged, and I am learning a lot so far!

Lisa Loper

Speaking of challenges, Steve Lemme, our Spring Conference keynote speaker, will share some of his insights about how Oracle professionals can plan to meet the challenges of the future. I interviewed him for the *NoCOUG Journal* last year and I found his insights invaluable. As a longtime board member of the Independent Oracle Users Group, Lemme is a big advocate for DBAs. His lessons on how DBAs and developers play a critical part in their respective roles at work are well worth hearing. Don't miss his keynote: *Fusion: The New Frontier—Oracle Administration in the Future.*

The Spring Conference is also full of great technical sessions from returning popular presenters, as well as new ones. It takes place on Thursday, May 17, at Lockheed Martin in Sunnyvale. You'll find the complete agenda with session descriptions on our website at *www.nocoug.org*. See you there! ▲

Fireside Chat with Craig Shallahamer



Craig Shallahamer

Oh the posh, posh traveling life, the traveling life for me;
First cabin and captain's table, regal company.
Whenever I'm bored I travel abroad but ever so properly;
Port out, starboard home, posh with a capital P-O-S-H, posh!
—Potts Senior in *Chitty Chitty Bang Bang*

Craig Shallahamer's smiling face has been spotted before at NoCOUG. He has spoken at NoCOUG conferences on several occasions, including keynotes in winter 2003 and fall 2004. As you can probably tell from the smile, he's easygoing—I once went up to him after he had finished presenting at a NoCOUG conference at the Computer History Museum and asked him to autograph and inscribe a book not written by him, and he smilingly complied with this fairly idiotic request. So, when I heard that he was finally writing a book, I decided to track him down to find out more about it. Because what you can't tell from his smiling face is that he is one of the most authoritative independent Oracle performance specialists in the industry.

Do try to catch Craig's talk the next time he visits NoCOUG—I promise it'll be worth your while. If you can't wait, go check out his website, **www.orapub.com***—there's lots to learn there, and freebie scripts. You might want to add yourself to his mailing list—if the Oracle community were MySpace, he would have the most friends, and there's got to be a good reason for that.*

With that short introduction, heeeeeere's Craig!

Introductions

When did it all begin? How did it unfold? Where is it leading?

I joined Oracle in 1989 as a Forms 2.3 developer. I knew nothing about Oracle technology and little about relational databases. After a few projects I took a six-month tour with the Oracle Hong Kong office. It was an amazing experience, but when I returned home I was having problems finding my niche in the consulting organization. It was during this time I decided to go full on into Oracle technology. So I decided to take a class led by Bob Rudzki and Cary Millsap of Oracle's National Technical Response Team. They were essentially a national SWAT team for really messed upon Oracle Application sites. After the class I asked if I could join their group. They agreed, and my life was changed forever.

Where it's going, I'm not exactly sure yet. But there is definitely a shift occurring toward forecasting and an intense focus on HoriZone.

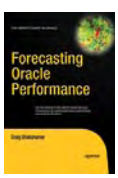
What about OraPub? Is it true that you have more than 20,000 Oracle professionals on your mailing list? What do you discuss in your communications? By the way, why aren't I on the list? What do I have to do to make the cut?

When I was working at Oracle, I started getting lots of emails from people asking for copies of my technical papers. While honored by their requests, in 1994, using dial-up from a hotel, this became overwhelming. Fortunately, the Internet came along and I could post my technical papers, that is, my "Oracle Publications," hence the name **OraPub**. Through no great strategic plan of my own, I personally registered, paid for, and owned the URL. By the time I left Oracle in 1998, Orapub.com was getting thousands of downloads. It was like manna from heaven, and allowed me to transition smoothly out of Oracle.

My mailing list is quite large, but honestly, there are so many younger DBAs out there now, so many Internet performance related sites, and others posting my papers on their websites, so it is difficult to let the new folks know about the work I've done and am doing. So 25,000 may sound like a lot, but there are a lot more out there.

Anyone can get on my list. This, of course, leads to a lot of bogus emails, so I wrote a script to search through my list and remove anyone with a bogus name or email, like Iggy. Actually, I don't have a script like that. Anyone who downloads a paper, tool, or registers for one of my classes will be on my list.

And how about HoriZone? We'll be spending a lot of time talking about it later, but do give us a little skinny, a little dope to whet our appetites.



Anyone who has spent lots of time firefighting performance problems recognizes that if there had been even a small dose of planning, the problem probably

"Anyone who has spent lots of time firefighting performance problems recognizes that if there had been even a small dose of planning, the problem probably would not have occurred."

would not have occurred. After a while you figure out the general idea is to anticipate where the risks are, how bad it will be, when the bad thing may happen, and what to do about it. This is one area of *risk management*. Essentially, I took my experiences and turned them into a product. The result is the initial release of HoriZone, which focuses on Oracle forecasting. It's essentially an application sizing risk-management product.

And the new baby? What is its name? Is it on store shelves yet? We'll be spending a lot of time later talking about that, too.

I am so proud of this book. It is entitled *Forecasting Oracle Performance*, and Apress is my publisher. It contains essentially everything I've learned over the past 20 years about forecasting. (Well, almost everything … I obviously have not put any HoriZone proprietary technology into the book.) I know it may sound weird, but I just loved reading the final revision! I hope other people enjoy it and find it practical and down to earth. The book is due in stores in April.

On the Traveling Life

How are you coping with the traveling life? What are the upsides and downsides you've found?

I can still remember the first time I called Oracle Travel after I joined Bob Rudzski and Cary Millsap's team. The adventure of traveling was so exciting. That excitement is still there, but in a differing way. When I first started, I would always try to get a window seat. Now I want an aisle seat. For me the love of traveling is being directly exposed to different people, cultures, and just taking it all in. But unless my family comes with me, a lot of the joy stays home. Here's a good upside; A couple of years ago, I did a class in Sydney, Melbourne, and Perth, all in a row over a three-week period. My entire family came with me, and we had a great time! So if I plan my schedule right, traveling can be turned into an awesome and unique opportunity.

You must have a lot of travel stories to tell. Has anything interesting happened to you lately? Funny? Frightening? Tell us a story. Tell us two. We love stories.

Of all the travel experiences I've had related to Oracle, by far the funniest (at least, now it's funny) was during my first international trip. Oracle flew me business class on Singapore Airlines to Hong Kong. Since this was my first international trip, I didn't know not to stuff myself with everything that was offered to me. The flight attendants were so sweet, I just hated to say no. So when I arrived in Hong Kong, I was worked. I got to my hotel and took a walk around. When I got back to the hotel I started to feel really bad. Whenever I start feeling bad, I start drinking water. So that's exactly what I did: drank water, lots of it. The water in my room was soon gone, so I started drinking water out of the tap. But instead of getting better, I started to get worse! I got this pounding headache and my stomach was killing me. I can still remember thinking, "I'm gonna die! And if I do die, it could be days before they find me!" I felt so bad, I couldn't even walk to the bathroom. I had to literally crawl. But somehow I went to sleep. To my surprise, when I woke up I felt okay. Later I found out that while the water doesn't contain viruses or bad things like that, it is heavily chlorinated. So I was slowly poisoning myself by filling my body with chlorine. No wonder I had a pounding headache and a raging stomachache. It was by far the most horrible travel experience I've ever had. But even writing about it now makes me laugh.

"When a company invests strategically in IT, both their business and their brand are at stake. If the systems go down, or performance prevents people from using the system, tons and tons of money gets flushed down the toilet."

How about some travel tips? Airports to avoid? Best laptop computer for road warriors?

First rule: Do not run in the airport. It's not worth it. Second rule: Eat lots of salad—it's better for you than the other stuff. Third rule: Beat jet lag by working out the day you arrive—it's worth the pain.

About three years ago, I took my Windows PC and slowly and methodically sawed it in half! Well, I didn't actually do that, but I wanted to. I did, however, switch to a Mac. It's not the best road warrior laptop, so I wrap it in bubble wrap and treat it like a baby when I travel.

Craig's New Baby

My mother once said that all babies are the same. They're all so cute, and they all noisily demand lots of our attention. All new parents think that their new baby is extra-special. I've got a shelf full of books already. Why yours? And what's the title again?

I suspect the forecasting section of your bookshelf is pretty bare. As far as I can tell, there are currently no other Oracle forecasting books on the market. There's a good reason for that, too: People familiar with capacity planning, performance modeling, and forecasting know Oracle makes their lives just plain difficult. Just one general example is Oracle's optimization algorithms—while they may increase performance, they also make honestly assigning resource consumption extremely difficult. If you have spent any time reading capacity planning books, you've probably felt like you learned some good stuff, but how that good stuff can be applied in a real-life production Oracle system was conveniently left out. Arghh! I also think the math scares most people. I was very careful in this book to methodically lead the reader through the entire forecasting process—and not skip any steps! I also tried really hard to help the reader translate the technical forecasting data into management information. Forecasting can be fun, but if

it cannot be fed into the management decision-making process, you won't be doing it very long.

So I wrote the book to be more of a practitioner's handbook. I tried to keep the math to a minimum, focus on real Oracle systems, and use examples that people can easily relate to and actually do themselves. The book is stuffed full of examples. In a way, the book is written how I like to teach people about forecasting. I lecture, we discuss, I present an example, we discuss some more, and then the student does a case study. The book is just like that.

In the description of the book, you promise to teach us how to prevent our employers from becoming fodder for the front page of* The Wall Street Journal.[1] *Would you remind us of some high-profile cases?

Ah . . . no. My legal team is expensive! But here's the situation: When a company invests strategically in IT, both their business and their brand are at stake. If the systems go down, or performance prevents people from using the system, tons and tons of money gets flushed down the toilet. To make matters worse, if the company is well known, they may get written up in *The Wall Street Journal*. Getting written up because your system is down is an event trigger to place your résumé on Monster.com.

Noel Yuhanna reported at Oracle OpenWorld that Oracle professionals spend a ridiculous percentage of their time in performance tuning and neglect other aspects of database administration, such as security. It seems to me that performance tuning and capacity planning are tightly intertwined, because a lack of attention to measuring workloads and ensuring that there is enough capacity creates the need for "tuning" down the road. Why do we think a system will perform if not planned correctly, and if we have no insight into the components of the workload? Are there technological barriers to quantifying, measuring, and tracking workloads?

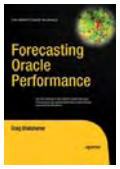
"She would put her head down and just barrel into traffic without double-checking to ensure there were no cars coming. I think most people do the same thing with Oracle."

I think for many people, putting an application into production, or dealing with change, is similar to learning how to drive. This past year I taught my oldest daughter how to drive. I noticed that when she decided to pull out into traffic, she would put her head down and just barrel into traffic without double-checking to ensure there were no cars coming. I think most people do the same thing with Oracle. They just put their heads down and hope it will work out fine in the end.

Forecasting can be complicated, and it will seem really complicated if you haven't received some training. So most people just hope everything works out. And you know, many times it does work out okay. But sometimes, you just can't take that chance. When there is too much at stake to just go for it, that's when forecasting (and all that surrounds it) becomes extremely valuable to a company.

[1] www.apress.com/book/bookDisplay.html?bID=10271.

There are definitely significant technical barriers to proactively managing Oracle performance. Just knowing classic capacity-planning mathematics or [having] mainframe capacity-planning experience is simply not enough on a real-life production Oracle system. It's just plain hard. It's difficult to gather all the workload information you want without clobbering your system; it's difficult to transform that data into useful information; it's difficult to create a precise enough forecast model; and it's difficult to translate the technical results into usable information that management can understand. As you might imagine, each of these topics is covered in my book and much of this done for you in my HoriZone product.

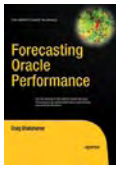


Database professionals pay more attention to "new features" and such, instead of mundane operational aspects such as service-level management and change management. I believe your book is the first Oracle book ever to touch on these topics. What's the scoop? Did you get religion?

I'll admit it was really difficult for me to get my hands around the concept of service-level management. I was never taught to view IT as *providing a service*. For me, it was a game, a puzzle, and a fun thing to do. I was never involved in quantifying the cost and benefits of the services IT provided (or didn't provide). For example, I would think, "I just reduced the query from 120 seconds down to two seconds. I'm a genius!" But the businessman thinks, "Now we can service twice as many customers as before! Therefore, our customer's satisfaction will increase while keeping [our] head count to a minimum, resulting in a margin increase of 2%!" It's a whole different way of viewing IT, and this is how IT management, and even more so the users and the business, view IT. When you get right down to it, we're servants.

Honestly, I stumbled on service-level management by failing to sell my forecasting product. I was focusing on the technical stuff like utilization, response time, and model forecast precision. I failed to relate these technical metrics to the business. So while the DBAs would see value, it made a tough business sell.

HoriZone



What's that all about? What's with the name? Who developed it? Can we take it for a test drive?

It seems strange to say this, but HoriZone was birthed over three years ago. When it was originally released, it was more a technical exercise than a product. But that's all changed now and, as I'll tell you about, there is a major change coming.

The name "Hor-i-Zone" is perfect because if you are standing on the beach on a clear day and look out and into the *horizon*, you are probably thinking about far-off things: the future, planning, risk, how to mitigate risk, etc. This is what

the product is all about: managing, predicting, and prioritizing risk. I actually architected and designed the core product technologies. A friend of mine partnered with me for over a year to take the raw product and turn it into a usable product. If you have seen even the static website (**www.horizone.orapub.com**), you'll quickly notice the wonderful design job he did. One of my focuses for this product has always been to make it elegant. The design and user interface go a long way in making this happen. My buddy did such a good job, I have to clearly explain the forecasting process so people understand just how much HoriZone does for them. Honestly, the interface makes forecasting so simple, it can seem simplistic.

"My best time-management tip is to recognize what you do best at different times of the day. It's amazing what I can do if I plan my day right."

The introductory HoriZone has developed into a very nice application sizing product. It's used to identify the future risk of overutilized resources, develop strategies to mitigate the risk, and provide a way to communicate this to management. For example, you can submit a scenario like, "If the workload doubles and we use the vendor's new CPUs, can the system handle the load? And if not, what's going to break and what can we do about it?" That's powerful stuff! The product has been available for about a year now.

But that's about to change. I'm working with a group of investors to build a management, sales/marketing, and product development organization. This will provide the needed backing and jolt so HoriZone can be properly transformed in a world-class, Oracle-focused, service-level management solution. As much as I'd like to tell you about the details, I'm noticing some of my marketing messages and themes are being closely watched and used by my competition. So I need to keep the details quiet for now.

So you want a test drive, huh? Okay. I'll work with one company to fully implement the current application sizing product and with another company to work on implementing the transformed HoriZone. (In fact, I'm always looking for good beta sites.) After they've used the product for a while, you can do a write-up about their experience. That's better than me yapping about it, anyway. So if anyone is interested, just email me at **craig@orapub.com**.

Parting Thoughts

You have a prodigious work output: running a company, developing new products, teaching, writing, and speaking. What's your secret? What keeps you motivated? Do you have any time-management tips for us?

I am an inventor, a framer, and very inquisitive. I also believe I have been placed on this earth for a very special purpose. I try to align my God-given talents with the purposes set before me. When that happens, it's wonderful. It's like life is being breathed into me. When it doesn't happen like this, I get lazy and want to sleep in the mornings. You know, stuff like that. That's not living, that's existing—and that's not the way I want to live.

My best time-management tip is to recognize what you do best at different times of the day. For example, for me, my intellectually sharpest and deepest analysis time is in the morning before 10:00. In the afternoon, I focus on relationships or softer things, where I can talk and communicate without the need to iterate five levels deep. It's amazing what I can do if I plan my day right.

Thank you for giving us so much of your time today. One last question: Does Craig Shallahamer, the entrepreneur, developer, teacher, writer, and speaker have a central message—and what might that be?

I'd have to say my central message and my advice is to move into the direction that brings life into what you do. Real life encourages and empowers people. Artificial life steals life, kills life, and destroys life. I hate that kind of thing, so my personal focus is to bring life. I'm not always successful in doing that, but that's my theme. ▲

Craig Shallahamer's favorite hangout is **www.orapub.com**. *You can get on his mailing list by going to* **resources.orapub.com/AccountSettings.asp?AddNewCustomer=Y**. *You can read about his new book at* **www.apress.com/book/bookDisplay.html?bID=10271**. *Send an email to* **craig@orapub.com** *if you would like to take advantage of the free offers mentioned in this interview.*

Expert Oracle Database Architecture: 9*i* and 10*g* Programming Techniques and Solutions

A Book Review* by Ken Jacobs

Details

Authors: Thomas Kyte

ISBN: 1590595300

Format: Softcover

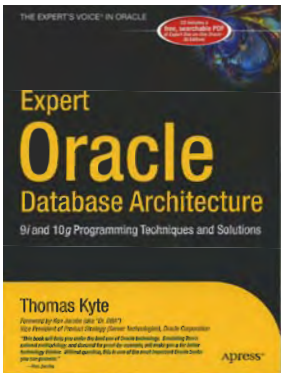
Pages: 768

Year of Publication: 2005

Edition: 1

Price: $49.99

Publisher: Apress



Review*

THINK. In 1914, Thomas J. Watson, Sr. joined the company that was to become IBM, and he brought with him this simple one-word motto. It was an exhortation to all IBM employees, no matter their role, to take care in decision-making and do their jobs with intelligence. "THINK" soon became an icon, appearing on publications, calendars, and plaques in the offices of many IT and business managers within and outside IBM, and even in *The New Yorker* magazine cartoons. "THINK" was a good idea in 1914, and it is a good idea now.

"Think different." More recently, Apple Computer used this slogan in a long-running advertising campaign to revitalize the company's brand, and even more important, to revolutionize how people think of technology in their daily lives. Instead of saying "think differently," suggesting *how* to think, Apple's slogan used the word "different" as the object of the verb "think," suggesting *what* to think (as in, "think big"). The advertising campaign emphasized creativity and creative people, with the implication that Apple's computers uniquely enable innovative solutions and artistic achievements.

When I joined Oracle Corporation (then Relational Software Incorporated) back in 1981, database systems incorporating the relational model were a new, emerging technology. Developers, programmers, and a growing group of database administrators were learning the discipline of database design using the methodology of normalization. The then unfamiliar nonprocedural SQL language impressed people with its power to manipulate data in ways that previously took painstaking procedural programming. There was a lot to think about then—and there still is. These new technologies challenged people not only to learn new ideas and approaches, but also to think in new ways. Those who did, and those who do, were and are the most successful in creating innovative, effective solutions to business problems using database technology to its best advantage.

Consider the SQL database language that was first introduced commercially by Oracle. SQL permits application designers to manipulate sets of rows with a nonprocedural (or "declarative") language, rather than writing iterative loops in conventional languages that process records one at a time. When I was first introduced to SQL, I found it required me to "think at 45 degrees" to figure out how to use set processing operations like joins and subqueries to achieve the result I wanted. Not only was the idea of set processing new to most people, but so also was the idea of a nonprocedural language, where you specified the result you wanted, not how to derive it. This new technology really did require me to "think differently" and also gave me an opportunity to "think different."

> ***"In most cases, it is application design, rather than Oracle parameter settings or other configuration choices, that most directly determines overall performance."***

Set processing is far more efficient than one-at-a-time processing, so applications that fully exploit SQL in this way perform much better than those that do not. Yet, it is surprising how often applications deliver suboptimal performance. In fact, in most cases, it is application design, rather than Oracle

* Excerpted from the foreword to *Oracle Wait Interface: A Practical Guide to Performance Diagnostics & Tuning* by Richmond Shee et al., with permission from McGraw-Hill.

parameter settings or other configuration choices, that most directly determines overall performance. Thus, application developers must learn not only details about database features and programming interfaces, but also new ways to think about and use these features and interfaces in their applications.

Much "conventional wisdom" exists in the Oracle community about how to tune the system for best performance or the best way to use various Oracle features. Such "wisdom" sometimes becomes "folklore" or even "mythology," with developers and database administrators adopting these ideas uncritically or extending these ideas without reasoning about them.

One example is the idea that "if one is good, more—lots more—is better." This idea is popular, but only rarely true. Take Oracle's array interface, for example, which allows the developer to insert or retrieve multiple rows in a single system call. Clearly, reducing the number of network messages between the application and the database is a good thing. But, if you think about it, there is a point of diminishing returns. While fetching 100 rows at once is far better than one at a time, fetching 1,000 rows at once instead of 100 is generally not really any more efficient overall, especially when you consider memory requirements.

"Much 'conventional wisdom' exists in the Oracle community with developers and database administrators adopting these ideas uncritically or extending these ideas without reasoning about them. One example is the idea that 'if one is good, more—lots more—is better.'"

Another example of uncritical thinking is to focus on the wrong aspects of system design or configuration, rather than those most likely to improve performance (or, for that matter, reliability, availability, or security). Consider the "conventional wisdom" of tuning the system to maximize the buffer hit ratio. For some applications, it's true that maximizing the chance that required data is in memory will maximize performance. However, for most applications it's better to focus attention on performance bottlenecks (what we call "wait states") than it is to focus on specific system-level metrics. Eliminate those aspects of the application design that are causing delays, and you'll get the best performance.

I've found that breaking down a problem into smaller parts and solving each part separately is a great way to think about application design. In this way, you can often find elegant and creative uses of SQL to address application requirements. Often, it is possible to do things in a single SQL statement that at first seem to require complex procedural programming. When you can leverage the power of SQL to process sets of rows at a time, perhaps in parallel, not only are you more productive as an application developer, but the application runs faster as well!

Sometimes, best practices that were based, even in part, on some degree of truth become no longer applicable as the facts change. Consider the old adage, "Put indexes and data in separate tablespaces for best performance." I've often seen database administrators express strong opinions over the merits of this idea, without taking into account changes in disk speeds and capacities over time, or the specifics of given workloads. In evaluating this particular "rule," you should think about the *fact* that the Oracle database caches frequently and recently used database blocks (often blocks belonging to an index) in memory, and the *fact* that it uses index and data blocks sequentially, not simultaneously, for any given request. The implication is that I/O operations for both index and data really should be spread across all simultaneous users, and across as many disk drives as you have. You might choose to separate index and data blocks for administrative reasons or for personal preference, but not for performance. (Tom Kyte provides valuable insights on this topic on the Ask Tom web site, **asktom.oracle.com**, where you can search for articles on "index data tablespace.") The lesson here is to base your decisions on facts, and a complete set of current facts at that.

"Sometimes, best practices that were based, even in part, on some degree of truth become no longer applicable as the facts change. Consider the old adage, 'Put indexes and data in separate tablespaces for best performance.'"

No matter how fast our computers or how sophisticated the database becomes, and regardless of the power of our programming tools, there simply is no substitute for human intelligence coupled with a "thinking discipline." So, while it's important to learn the intricacies of the technologies we use in our applications, it's even more important to know how to think about using them appropriately.

"Tom encourages you to treat Oracle as much more than a black box. Instead of you just putting data into and taking data out of Oracle, Tom will help you understand how Oracle works and how to exploit its power."

Tom Kyte is one of the most intelligent people I know, and one of the most knowledgeable about the Oracle database, SQL, performance tuning, and application design. I'm pretty sure Tom is an aficionado of the "THINK" and "Think different" slogans. Tom quite obviously also believes in that

anonymous wise saying, "Give a man a fish and you feed him for a day. Teach a man to fish and you feed him for a lifetime." Tom enjoys sharing his knowledge about Oracle, to the great benefit of our community, but rather than simply dispensing answers to questions, he helps others learn to think and reason.

On his web site (**asktom.oracle.com**), in his public speaking engagements, and in this book, Tom implicitly challenges people to "think differently" too, as they design database applications with the Oracle database. He rejects conventional wisdom and speculation, instead insisting on relying on facts proven through examples. Tom takes a very pragmatic and simple approach to problem solving, and by following his advice and methodology, you can be more productive and develop better, faster applications.

Not only will Tom's book teach you about features of Oracle and how to use them, but it also reflects many of these simple thoughts:

- Don't believe in myths—reason for yourself.
- Don't follow "conventional wisdom"—often the things everybody knows are simply wrong!
- Don't trust rumors or opinions—test things for yourself and base decisions on proven examples.
- Break apart a problem into simpler questions, and assemble the answers to each step into an elegant, efficient solution.
- Don't do things in your programs when the database can do them better and faster.
- Understand the differences between the ideal and the real.
- Ask questions about and be skeptical of unjustified "company policies" for technical standards.
- Consider the big picture of what's best overall for the requirements at hand.
- Take the time to THINK.

Tom encourages you to treat Oracle as much more than a black box. Instead of you just putting data into and taking data out of Oracle, Tom will help you understand how Oracle works and how to exploit its power. By learning how to apply Oracle technology creatively and thoughtfully, you will be able to solve most application design problems quickly and elegantly.

As you read and enjoy this book, I know you'll learn a lot of new facts about Oracle database technology and important concepts about application design. As you do, I'm confident that you'll also start to "think differently" about the challenges you face.

IBM's Watson once said, "Thought has been the father of every advance since time began. 'I didn't think' has cost the world millions of dollars." This is a thought with which both Tom and I agree. Armed with the knowledge and techniques you'll learn in this book, I hope you'll be able to save the world (or at least your enterprise) millions of dollars, and enjoy the satisfaction of a job well done. ▲

About the Author(s)

Thomas Kyte has been using the Oracle database since 1988 and version 5. He is a vice president in the Oracle Public Sector group and the Tom behind **asktom.oracle.com**. He blogs at **tkyte.blogspot.com**.

Ken Jacobs is Vice President of Product Strategy in the Server Technologies division at Oracle. He holds several software patents, including Patent 5,870,758 for a "Method and Apparatus for Providing Isolation Levels in a Database System," which describes Oracle's concurrency control scheme in which readers never block writers and writers never block readers.

Why Oracle?

Ask the Oracles!

Mogens Nørgaard: Splendid question. I assume we're only discussing the Oracle database, not Oracle as a company.

Most systems in this world get by without an Oracle database. Just think of all the systems that run without a relational database, plus all the systems that run on top of a relational database other than Oracle's. There are tasks for which databases are not appropriate. Imagine trying to run Google on an Oracle database.

So why Oracle?

In Miracle A/S we have recently hired a few .NET programmers and architects, and they don't really know "why Oracle," or indeed why any database at all. They just want permanent storage, be that files or databases or a third option.

> *"They don't really know 'why Oracle,' or indeed why any database at all. They just want permanent storage, be that files or databases or a third option."*

That tells me it will get harder in the future to answer this question—some of my old-time, hardcore Oracle techies even think the Oracle database is irrelevant ten years from now.

My reasons for choosing Oracle over SQL Server?

- The basic, old, and tried architecture with its read consistency;
- fantastic backup/restore/recovery features;
- my beloved time-based instrumentation (a.k.a. the wait-interface);
- all the indexing options you'd ever want;
- the myriad features that ensure you can do pretty much anything; and, of course,
- RAC.

That's why. ▲

*Mogens Nørgaard is the CEO of Miracle A/S (***www.miracleas.dk***), a database knowledge center and consulting/training company based in Denmark, and is the co-founder and "father figure" of the Oak Table network. He is a renowned speaker at Oracle conferences all over the world and organizes some highly respected events through Miracle A/S, including the annual Master Class and the Miracle Database Forum. He is also the co-founder of the Danish Oracle User Group (OUGKD), and was voted "Educator of the Year" in* Oracle Magazine*'s Editor's Choice Awards, 2003. Mogens can be reached at* **mno@miracleas.dk**.

Jeremiah Wilton: Often the answer is "not Oracle." Oracle is expensive and complex, and not for every application needing a database. In addition to the relatively high license and support costs, Oracle requires the services of database administrators, a very expensive and opinionated class of technical professionals. The open-source community has produced some fine packages which, while currently weaker than Oracle in features and scalability, have great usefulness in a broad variety of applications. Oracle's decades of concerted development have evolved a maturity that makes it the best option, by no means for all, but for certain enterprise-class requirements that cannot be met by other products.

Oracle, if you want a consistent view of data without blocking writers. Other products have improved their locking and serialization models over the years, but Oracle is still the premier database providing multi-version read consistency (MVRC). Oracle's adherence to this model is long-standing, and it is a fundamental characteristic of the Oracle server. Oracle uses undo to provide users with a view of data consistent with respect to a fixed point in time, even if that data is being changed out from under the reader. This model allows writers to make changes seamlessly to data even if others are reading it.

> *"Often the answer is 'not Oracle.' Oracle is expensive and complex, and not for every application needing a database."*

Multi-version read consistency becomes important as concurrency and scale of an application increase. Because readers do not block writers, applications can perform operations in many threads simultaneously without the database serializing

work unnecessarily. In addition, MVRC allows, within limits, for reporting and decision support to take place in the same instance as OLTP operations.

Oracle, if you want the option of scaling across multiple inexpensive nodes. With Oracle 9*i*, Oracle became best of breed in shared-everything cluster databases. While most other clustering technologies require or encourage users to segregate their data and not actively share data across hosts, Oracle's RAC does away with such concerns to a degree far greater than any competing clustering model. Although scaling on RAC is not linear, and cross-node contention for heavily-changed buffers is still possible, RAC opens up scalability to inexpensive units of architecture with greater ease and simplicity than are available on any other platform.

> ***"MVRC allows, within limits, for reporting and decision support to take place on the same instance as OLTP operations."***

Oracle, if you want to tune without speculation. Hands down, Oracle has provided among the richest instrumentation features of any enterprise software package currently available. They led the way in time-based performance analysis with the wait-event interface in Oracle 7. With Active Session History (ASH) in Oracle 10*g*, Oracle provides incredibly rich performance data by inexpensively sampling and tracking wait events, sessions, and cursors over time. Competing database products not only fail to provide a similar feature, but have not even caught up to the basic wait-tracking provided by Oracle in June 1992. Performance tuning on many other database platforms is a guessing game.

While third-party wait-tracking products from Precise and Quest have been available for other database platforms for some years, they restrict users to their GUI interface, and refuse to publish their agent API. In Oracle's case, the data is accessible via tables and views, and can be queried just like any other data. The sampled data is thus available for scripts, monitoring, and other repeatable operations.

> ***"Performance tuning on many other database platforms is a guessing game."***

By no means are read-consistency, shared-everything clusters and wait sampling the only features that differentiate Oracle from competing packages. No doubt you have your own favorites. But the above features are so fundamental to providing availability, scalability, and stability to an application that their absence in competing products, especially those from private commercial sources, makes Oracle than much more competitive.

---

*Jeremiah Wilton (***jeremiah@ora-600.net***) was the first DBA at* Amazon.com*. He is now principal consultant at ORA-600 Consulting in Seattle. An occasional speaker at NoCOUG and other conferences, Jeremiah specializes in complex recoveries, high availability, and scalability. His white papers on these and other topics can be found at* **www.ora-600.net**.

Don Burleson: "Why Oracle?" This was the question that everyone asked when Oracle was displacing mainframe DB2 systems back in the early 1990s. Back then, Oracle had carved out a niche as being a versatile database, not constrained, as DB2 was to MVS and Informix is to UNIX. At the time, I was charged with making a "buy" decision for the new "mini computer" databases, and I've been intimately involved in this question for decades.

Oracle is not the most elegant database, and it's not always the fastest platform. Oracle rose to the pinnacle of the IT world on other virtues. Rather than wait for the end-user community to react to market changes, Oracle took a proactive approach and often introduced features far ahead of the market demand. I remember when Oracle 7 was undergoing beta testing and we wondered about the new "cost-based optimizer," "hash joins," and bitmap indexes. In Oracle 8, we replaced EBU with RMAN and learned about partitioning and star transformations. And so on—high anticipation with every release.

At Oracle OpenWorld 2006, Oracle announced some exciting new features of Oracle 11*g* and promised a mind-boggling 482 of them, some minor and some profound, including rolling upgrades, query results caching, and capture and replay of database workloads.

It's not a random event that Oracle dominates the world's database market and runs on over 60 platforms, everything from a mainframe to a Mac.

Oracle does have many competitors in the marketplace, ready to pounce on any perceived weakness. We see these arguments from Oracle competitors:

Oracle is expensive—hey, you get what you pay for. Plus, Oracle now has the free Oracle XE to compete with MySQL.

Oracle is too complex—Complexity is the natural by-product of being robust and flexible. However Oracle addresses this issue with automated memory and storage management. Beginning with 11*g*, Oracle's Automatic Storage Management (ASM) will enable a single storage pool to be shared by multiple databases for optimal load balancing. Also, in 11*g* all memory can be tuned automatically by setting one parameter. You literally tell Oracle how much memory it can use, and it determines how much to use for PGA, as well as SGA.

> ***"At Oracle OpenWorld 2006, Oracle announced some exciting new features of Oracle 11g and promised a mind-boggling 482 of them."***

When managers have to make a "buy" decision for database software, they just go with the market leader. Isn't that enough rationale? ▲

Don Burleson has been a full-time DBA for 25 years and has authored or coauthored 34 books on Oracle database management, including five Oracle Press books. He serves as an advisor for many Fortune 500 Corporations, and his website is **www.dba-oracle.com**.

James Koopmann:

9. We all like to sit on the edge of our seats, waiting to see what the suffix to the next version number will be. It always seems to be tied to an industry buzzword, so I wonder what that next buzzword will be. World domination? Oracle 12*wd*?

8. I am waiting for my WebMaster trading card to really skyrocket in price. These came out ten years ago. The only problem is that I just might have to wait another ten years for them to really catch on.

7. Don't we all just want to wait and see how this automatic database everything is going to pan out? All we DBAs need to do is keep that feature hidden, or tell everyone how it doesn't really work for *our* particular environment. Then we are safe.

6. I'm waiting to see what Oracle names its operating system. Hmm, "O/S." I think they already have that one figured out. Oracle/System, don't you know? Just wait for the patent on O/S.

5. There is always something new to learn as new versions come out. Oh, and we get to relearn old features as new "fixes" are introduced. Heck, we can even justify going to a training class or two as Oracle renames and repackages an old feature.

4. The conferences are just too good to pass up. It sure is nice to see that marketing budgets have begun to rise again. The number of party sponsorships has come back up, but where are all the T-shirts? My collection is getting old and ragged.

3. Aren't we all just sitting around, waiting to implement RAC? And doesn't ORACLE really stand for "Our RAC Learning Environment?" If we only had the time to prove all those theories.

2. Oracle is still a black art to many of our managers, which allows us free will to spend countless hours researching and implementing stuff we already know how to do. If you have an Oracle-savvy manager, you might just have to relate any task to RAC, Grid, HA, or MAA. Then you are sure to get the blank stare you so desperately desire, and your job is safe once again.

1. And the number one reason why Oracle: Money, Money, Money. I am constantly told there is *gold at the end of the Oracle rainbow. So I stick around, just waiting to find that rainbow. Just so you know, I have been in Ireland and the rainbows aren't much different there!* ▲

---

*James F. Koopmann is founder of Pine Horse, Inc. (***www.pinehorse.com***). James is an accomplished author and has worked with a variety of database-centric software and tools vendors as strategist, architect, DBA, and performance expert. James can be contacted at* **jkoopmann@pinehorse.com**.

Data Palette from Strata Via

by Andrew Wright

Introduction by Iggy Fernandez

Do you know how to start or stop a database? I thought I did—until I was hired by my current employer. We have Solaris, AIX, HP/UX, and Linux. We have 8i, 9i, and 10g. We have VCS, HP Service Guard, Sun Clusters, and RAC. We have diverse applications that need to be stopped and started in conjunction with the database. We have backups, batch jobs, and SLAs to worry about. We have custom approval and notification requirements in each case. I'm sorry, there are so many variations of the startup and shutdown procedures that I can't remember all of them—I have to check the documentation each time and hope that it is accurate and detailed enough for the likes of me. Working here is an exercise in learning the extent of one's ignorance, and unhappy business users are not very forgiving when things go wrong. OEM is no help to me here. Is anybody listening? Help!

The Data Palette Way

Take a moment to reflect on your current database operations.

- How many platforms do you support? How many database platforms does your company have?
- Are the supported platforms all the same version?
- Are you taking advantage of the capabilities of new versions, or are you supporting new versions with old techniques?
- Who configured each database? Are all configurations the same?
- Are the day-to-day operations the same for each database across the board?
- How many members are on your DBA team? Do all of them handle all tasks in a consistent manner? Do they all prefer the command line, or do some of them have their own favorite set of tools?
- How many scripts are deployed in each database instance? Are they the same across each server? Is everyone on the team completely familiar with all these scripts? How do you keep these scripts updated on a regular basis? Are these scripts fully "aware" of your environmental policies, including maintenance windows?

"By getting everyone to follow the same task recipes, the underlying methods are standardized. This introduces consistency in quality of work across multiple DBAs, regardless of their level of experience."

Data Palette's Standard Operating Procedure (SOP) Module is designed to drive consistency and predictability across an organization. This is done using the three core Data Palette features:

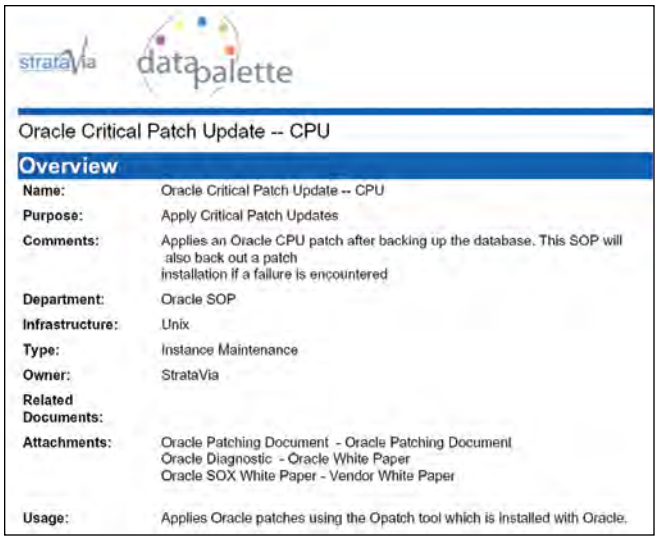
- Standardization and centralization of best practices using SOP Documentation from the Data Palette library, or by creating/importing your own
- Automation of these standardized best practices using metadata-driven SOP Workflows from the Data Palette library, or by creating/importing your own
- Ability to correlate events for prediction of problematic situations requiring automated responses, and the ability to trigger automation when such problem signatures are detected

Case Study

The 2007 daylight savings time change caused a lot of extra work for DBAs this year. These changes had to be researched, understood, tested, and then applied across the enterprise. We had Java and other patch routines for the DB2 and Oracle platforms, along with some T-SQL to run against SQL Server Notification Services. StrataVia published a 2007 DST Patch SOP, and Data Palette users imported this SOP using the SOP Import feature, reviewed the detailed documentation and the code that would be executed, and scheduled the patches across all of their databases, starting with development and test environments. Applying the patches in a consistent, lights-out manner, regardless of platform and without having to manually build custom scripts, took a lot of the pain out of this time-sensitive project.

Standardization and Centralization of SOPs

A centrally accessible set of "task recipes" is an important precursor to automation. This enables multiple DBA team members to be on the "same page" regarding the best way to carry out recurring tasks in their environment. By getting everyone to follow the same task recipes, the underlying methods are standardized. This introduces consistency in quality of work across multiple DBAs, regardless of level of experience. Unless recurring tasks are standardized up front, attempting to automate them is futile. Additionally, having a repository of such task recipes reduces the negative impact of



Oracle Critical Patch Update -- CPU

Overview

Name:	Oracle Critical Patch Update -- CPU
Purpose:	Apply Critical Patch Updates
Comments:	Applies an Oracle CPU patch after backing up the database. This SOP will also back out a patch installation if a failure is encountered
Department:	Oracle SOP
Infrastructure:	Unix
Type:	Instance Maintenance
Owner:	StrataVia
Related Documents:	
Attachments:	Oracle Patching Document - Oracle Patching Document Oracle Diagnostic - Oracle White Paper Oracle SOX White Paper - Vendor White Paper
Usage:	Applies Oracle patches using the Opatch tool which is installed with Oracle.

DBA turnover and accelerates ramp-up time for new DBAs coming in.

Data Palette comes preconfigured with a library of documented best practices for recurring and time-consuming tasks, such as database installation and provisioning, refreshes and cloning, and applying patches, database upgrades, database migrations, and automated maintenance plans across all major OS and DBMS platforms, including Oracle, SQL Server, and DB2 UDB. These tasks are documented in detail, down to the actual commands that are to be executed, in a sequential manner. DBAs can collaboratively accept these task recipes out of the box, or modify them to better fit their organization. New SOP documents can be created for highly custom and difficult tasks, and existing documentation in the form of Word documents and other formats can be imported into the Data Palette platform to be shared via the central documentation library.

The Data Palette SOP Export feature is used to share documentation in PDF format, which enables junior or new DBAs to quickly educate themselves on the operational procedures for your organization. This reduces much of the hand-holding required to cross-train a DBA on new platforms or environments. The SOP documentation can also be sent to external groups, such as change control boards, help desk operators, and IT auditors to familiarize them with the procedures applicable to your databases.

Automation of SOP Workflows

Every SOP document should have a corresponding automation routine. Data Palette comes preconfigured with a variety of automated tasks via its SOP Workflow Library. This open-source SOP library enables rapid implementation of best practices across your organization and includes automation workflows for all the tasks mentioned earlier and more.

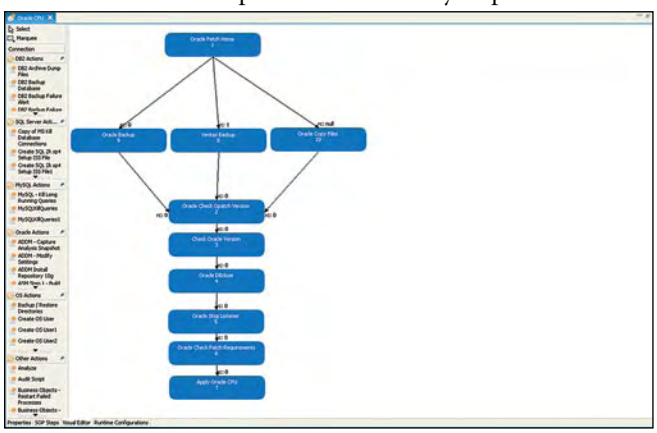
Data Palette SOPs leverage metadata from your environment to scale across hundreds of servers, multiple operating systems, and various configurations. For instance, let's assume that the maintenance window for all your data warehouse–related databases is Saturday night from 9:00 to 11:00, EST. This value is defined in a metadata variable and is referenced within an SOP Workflow to avoid any unintended execution of automation routines outside the maintenance window. However, let's say the maintenance window changes for a subset of the databases to Sunday morning from 5:00 to 7:00, EST. One needs only to update the value for the affected databases in a central property sheet, and the metadata reference within the SOP workflow is correspondingly updated, thereby avoiding any code-level changes. Similarly, if the ORACLE_BASE or ORACLE_HOME values or any configuration parameters change after an upgrade or a migration, the SOP automation workflows are able to get the most current value via metadata references embedded in the workflow, thereby avoiding the need for any manual code updates. In the latter example, the metadata variables get the most current values due to Data Palette's built-in environmental change detection and capture capabilities.

Data Palette SOPs use a workflow structure to chain common task steps together. This structure enables you to reuse your existing work as part of new tasks. For instance, you can drag and drop your existing Oracle Backup task into the Oracle CPU Patch Workflow to perform a backup prior to applying a patch. The metadata within the workflow allows the SOP to be flexible across database configurations. For example, as part of the Oracle CPU Patch SOP, the ShutDownMethod metadata variable is used to determine how to shut down the target database, depending on whether the instance is a single node, part of a RAC cluster, or a Data Guard configuration. If required, this SOP workflow can be modified to accommodate custom shutdown requirements, such as terminating certain application processes prior to shutting down the database or referencing a third-party tool to carry out any of the steps. This flexible structure and metadata allow you to run the same SOP across multiple database configurations.

The workflow structure enables your SOPs to be integrated with any existing third-party software, such as the service desk and ticketing system, enterprise monitoring tools, scheduling tools, and other provisioning and configuration management databases (CMDBs). These steps are often done manually by the DBA before or after the task, such as updating the patch level in your CMDB, or creating a ticket in the incident management system before proceeding with resolving an alert/page triggered by the monitoring tool.

The SOP Workflows that ship with Data Palette encapsulate the procedures documented in the corresponding SOP documentation library to apply consistent, predictable best practices across all of your databases, with the capability to view and customize the documentation and the workflows.

New SOPs are simple to automate if you prefer to reuse



your existing scripts within Data Palette. After you import an existing script, you can optimize it by replacing any hard-coded environmental references with dynamic metadata references, and then deploy your new SOP against a variety of database environments from the central console. Now you don't have to maintain a copy of each of your scripts on each of the database servers. The SOP module supports a variety of scripting languages, such as ksh, csh, Perl, VBScript, T-SQL, Python and Ruby. Data Palette ships with a Python interpreter, which is guaranteed to run in a consistent manner on all platforms, including Linux and Windows.

Automated Problem Fixes and Prediction Tools

Problem signatures are correlated events on your system that indicate the onset of a specific problem. This can include correlation of events both within and outside the database, such as the relationship between the number of connections to an Oracle instance and the percent of memory being used on that node. Problem signatures often tend to be more than simple point-in-time snapshots. For instance, they may need to refer to an increase in rate of growth of one or more resources such as disk space, I/O bandwidth on the SAN or NAS device, network bandwidth, or CPU usage. Short spikes may not be relevant, so Data Palette has built-in rules such as "high-memory utilization for the past five minutes."

Data Palette recognizes problem signatures, and SOP Workflows can be linked to these signatures to respond to the situation. Responding can mean many things, such as emailing or paging a DBA with a problem/root cause analysis report, creating a ticket in the incident management system, and/or executing an SOP to deal with the problem in a consistent and timely manner. A good example is adding a datafile when a tablespace starts to fill up and, if required, provisioning additional space on the SAN to allow the tablespace to expand automatically. A Data Palette SOP can be executed to accomplish this, while using your in-house best practices and datafile locations and naming conventions.

Data Palette also contains prediction tools to forecast problem events. For instance, the Data Palette Storage Space Projections module gathers historical data about the size of each tablespace and mount point, accurately predicts when a mount point will run out of space (i.e., one month from the current day), and will pinpoint the tablespace and database that will cause the space to run out. Data purges and other events are detected and incorporated into the analytic model to project this information accurately.

Summary

Many database environments are laden with ad hoc tools, hundreds of scripts, and impromptu task methods, causing DBA work to remain largely reactive. Data Palette's standardization, automation, and predictive capabilities can help control the environment in an evolutionary manner. ▲

Andrew Wright helped develop Data Palette and is now a senior sales engineer at StrataVia. For more information about StrataVia, visit **www.stratavia.com**.

The Impossible Dream

by Iggy Fernandez

This is my quest, to follow that star/No matter how hopeless,
no matter how far/To fight for the right without question or pause/
To be willing to march into hell for that heavenly cause
—Don Quixote in The Man of La Mancha

Iggy Fernandez

Summary

In previous essays, we described the immense difficulty of the query optimization problem and questioned the wisdom of relying on the query optimizer for predictable performance, suggesting that, contrary to decades of official advice, the only real solution was for the programmer to manually choose access paths and bake them into queries using the Hint mechanism. In this essay we describe two remarkable deficiencies of the Oracle query execution facility that are not related to the difficulty of the query optimization problem. We also study Oracle's proposal to finally cure these deficiencies in its next major release.

The Problem

Here is a comment from a participant in the "Ask Tom" forum. The first half of the author's sentence perfectly summarizes one of the biggest challenges (cardinality estimates) of the query optimization problem, and the second half of the sentence relates to the topic of this essay.

I became interested in the CBO's selectivity calculations trying to understand why it comes up with some of the ridiculously low cardinality estimates (like 1 when in reality there are 80,000+) which then lead to disastrous access plans that take hours, provided they finish at all, instead of minutes or seconds.[1]

While the optimizer can be excused for generating a bad execution plan, there is no excuse for the query execution facility staying the course when it becomes obvious that plan execution is taking far longer than had been estimated by the optimizer. The optimizer, of course, makes many assumptions when generating cardinality estimates for the steps in the query plan, and, as explained in previous essays, these estimates can be far from the mark. But actual cardinalities are obtained during plan execution. If these cardinalities are substantially different from the estimates generated by the optimizer, and if the query is taking far longer than had been estimated, it would be appropriate to stop executing the query midway and return it to the optimizer for re-optimization. The optimizer can then use the actual cardinalities obtained during partial plan execution to create a plan that is more likely to be executed in a reasonable time frame—a few milliseconds spent in reconsideration might pay handsome dividends, and the optimizer could decide whether to discard or retain the data sets generated during partial query execution.

Further, as is well known, your query plans are cached in the shared pool and are reused whenever the query is resubmitted by you or by another user. This is an eminently sensible thing to do when the time spent in query optimization is significant compared to the time spent executing the query. But it is not a very sensible thing to do when the time spent in query optimization is insignificant compared to the time spent executing the query, and when plan execution shows that the optimizer grossly underestimated the time it would take to execute the query using that plan—the optimizer should be advised to consider a change in strategy. [2]

We therefore see that any inefficiencies in a query plan are exacerbated by the obstinacy with which the query execution facility stays the course. [3]

The Solution

Oracle has apparently been thinking about this problem for a long time now, and the details of its solution and strategy are hidden in plain sight, on the website of the United States Patent and Trademark Office. Oracle filed 11 patent applications related to query optimization technology on September 7, 2004. A good deal of progress in implementing these inventions in the query optimizer has already been made in Oracle 10*g*, and the rest of the solution will presumably be unveiled in Oracle 11*g*.

20050097091	SQL tuning base
20050119999	*Automatic learning optimizer*
20050120000	Auto-tuning SQL statements
20050120001	SQL structure analyzer

[1] **asktom.oracle.com/pls/ask/f?p=4950:8:::::F4950_P8_DISPLAYID:4344365159075**.

[2] Complete execution details for every step in a query plan including execution time, number of data blocks that were examined, and the actual number of data rows that were generated can be examined using the V$SQL_PLAN_STATISTICS view.

[3] Oracle has two "query governor" features that you can use to play a game of "whack-a-mole." A limit may be set on the maximum amount of CPU time that a single query may use—different limits may be set for different users. Oracle automatically kills any query that exceeds its quota and issues an ORA-02393 error. This is, then, a systematic method of flagging queries for tuning—over time, all problematic queries will be identified and tuned. Refer to **download-west.oracle.com/docs/cd/B19306_01/server.102/b14200/statements_6010.htm#i2065930** for details. If you are licensed to use Oracle's Enterprise Edition, you can also use the Resource Manager to nip runaway queries in the bud—if Oracle estimates that a query will take longer than the allowable execution time, it issues an ORA-07455 error and does not execute the query. Refer to **download-west.oracle.com/docs/cd/B19306_01/server.102/b14231/dbrm.htm#i1010776** for details.

20050125393	SQL tuning sets
20050125398	Global hints
20050125427	Automatic SQL tuning advisor
20050125452	SQL profile
20050138015	High-load SQL-driven statistics collection
20050177557	*Automatic prevention of runaway query execution*
20050187917	Method for index tuning of a SQL statement, and index merging for a multi-statement SQL workload

Most of the patent applications relate to the "Automatic Tuning Optimizer" (ATO) introduced in Oracle 10*g* for "offline tuning" of individual queries and entire workloads.[4] Of most interest to us today is patent application 20050119999 for an "automatic learning optimizer" and patent application 20050177557 for a new type of query governor. The first patent application describes how offline tuning can be completely automated—presumably this technique will be implemented in Oracle 11*g*. [5] Here is an extract from the patent application. Of most interest is the *iterative* nature of the proposed process—problematic queries will automatically undergo as many rounds of improvement as necessary!

The auto-learning process starts by identifying a small subset of SQL statements which are potential candidates for auto-learning. For example, this subset may correspond to all SQL statements which are known to have a suboptimal execution plan and a high impact on the overall performance of the system. . . . Once the set of high-load SQL statements has been identified, the auto-learning optimizer can uncover its mistakes by analyzing each SQL statement in the set. Based on this analysis, if optimizer related problems are then found, corrective actions are produced and stored in a computer readable medium, such as a disk. Because the corrective actions are permanently stored, the auto-learning optimizer can perform an iterative learning process which accumulates, over time, more and more knowledge on problematic queries, and also prevents the corrected problems from recurring. . . . The auto-learning process may be on-line, with the auto-learning process running almost continuously as a background task. In this mode, high-load SQL statements stored in the cursor cache are targeted. Hence, the on-line mode can address critical SQL tuning issues while having a very low overhead on the system performance. . . . In another embodiment, the auto-learning optimizer performs learning off-line. In this mode, the auto-learning process is executed during the maintenance window as an automated manageability task. This off-line mode can have more time and system resources to perform the auto-tuning functions.

The second patent application describes how a query can be stopped *and restarted* if the online tuning process (described in the first application) finds an alternative query plan that is estimated to take less time than is estimated to be required to complete the query using the current query plan—presumably, this technique will also be implemented in Oracle 11*g*. You can inspect all the patent applications on the patent office's website, **www.appft1.uspto.gov/netahtml/PTO/srchnum.html**.

Concluding Remarks

Query optimization is not the slam-dunk exercise it is popularly imagined to be. The only alternative is manual query tuning, but few have the time or ability to do it well. If you have the Tuning Pack, help is on the way in the form of a learning optimizer, but you may have to wait quite a while—have you completed that upgrade to Oracle 10*g* yet? ▲

The author can be reached at **iggy_fernandez@hotmail.com**.

[4] ATO is exposed using the DBMS_SQLTUNE package. Refer to **download-west.oracle.com/docs/cd/B19306_01/server.102/b14211/sql_tune.htm** for details. You must have licenses for Diagnostics Pack and Tuning Pack in addition to a license for Enterprise Edition in order to use ATO.

[5] Nothing should stop the adventurous from attempting to do this themselves in Oracle 10*g* using the DBMS_SQLTUNE package.

SQL Challenge

I was asked to run the following SQL script on a production database. The data table in question contained approximately 100 million rows and each statement would update approximately five million rows.

```
UPDATE  /*+ PARALLEL (promotion,  8) */
        promotion p
   SET  promotion_id = 2412
 WHERE  promotion_id = 2182;

UPDATE  /*+ PARALLEL (promotion,  8) */
        promotion p
   SET  promotion_id = 2414
 WHERE  promotion_id = 2184;

UPDATE  /*+ PARALLEL (promotion,  8) */
        promotion p
   SET  promotion_id = 2416
 WHERE  promotion_id = 2188;
```

You are invited to criticize the script on as many grounds as possible. The prize for the most eloquent criticism is an iPod Shuffle and a CD of Man of La Mancha including the Impossible Dream song.

The contest is open to all NoCOUG members. Send your answers to **journal@nocoug.org** by May 31. The decision of the judges is final. ▲

The Best Is Yet To Come!

by Don Burleson

Don Burleson

Every new release of Oracle is highly anticipated, especially when the release introduces major new functionality. I remember when Oracle 7 was undergoing beta testing and we wondered about the new "cost based optimizer," "hash joins," and bitmap indexes. In Oracle 8, we replaced EBU with RMAN and learned about partitioning and star transformations. And so on—high anticipation with every release.

At Oracle OpenWorld 2006, Oracle announced some exciting new features of Oracle 11*g* and promised a mind-boggling 482 of them. Many of these enhancements are relatively minor (for example, the new "SIMPLE INTEGER" column datatype), while others are quite profound.

The trick to understanding the new features of 11*g* is to categorize the features as they apply to your own environment. For the purpose of this short article, I'll group new features into the following categories: RAC/Grid, DBA, developer, and PL/SQL.

*Note that Oracle 11*g *is still in beta testing, and all beta participants have non-disclosure agreements with Oracle, so we must reply on what's been revealed about Oracle 11*g *by Oracle Corporation itself.*

RAC/Grid Enhancements

> *"Ellison first promised this feature in 2002."*

RAC rolling upgrades—Oracle 11*g* promises to have a rolling upgrade feature whereby the RAC database can be upgraded without any downtime. Ellison first promised this feature in 2002, and it is a very challenging and complex new feature of 11*g*.

ADDM for RAC—The Automatic Database Diagnostic Monitor (ADDM) in 11*g* can give advice on the whole RAC (database level), not just at the instance level. Directives have been added to ADDM so it can ignore issues you are not concerned about. For example, if you know you need more memory and are sick of being told so, you can ask ADDM not to report those messages anymore.

Faster grid provisioning—Oracle 11*g* OEM has "provisioning" installs in which a binary footprint is tar'ed to the server blade and configured without a cumbersome installation process.

RAC instance load balancing—Starting in Oracle 10*g*R2, Oracle JDBC and ODP.NET provide connection pool load balancing facilities through integration with the new "load balancing advisory" tool. This replaces the more cumbersome listener-based load balancing technique. Starting with 11*g*, we see a RAC load balancing advisor utility.

DBA enhancements

Query results caching—This feature automatically caches the results of a SQL query (as opposed to the traditional caching of data blocks). Oracle describes this feature as "buffer cache taken to the next level." The DBA turns on this new feature with the command "alter table cache results," and the resulting per-process cache is shared across multiple sessions and at the client level. We also see the related RESULT_CACHE hint, which suggests that the result data will be cached in the data buffers, not in the intermediate data blocks that were accessed to obtain the query results. Oracle claims that you can cache both SQL and PL/SQL results for super-fast subsequent retrieval.

> *"Oracle describes this as buffer cache taken to the next level."*

Capture and replay database workloads—You can capture a production workload (with dbms_stats.gather_system_statistics) and apply it in a test and development instance to simulate a production workload during testing.

Automatic memory tuning—Automatic PGA tuning was introduced in Oracle 9*i*. Automatic SGA tuning was introduced in Oracle 10*g*. In 11*g*, all memory can be tuned automatically by setting one parameter. You literally tell Oracle how much memory it can use, and it determines how much to use for PGA, SGA, and OS processes. Maximum and minimum thresholds can be set.

Adaptive metric baselines—Notification thresholds in 10*g* were based on a fixed point. In 11*g*, notification thresholds can be associated with a baseline, so the notification thresholds vary throughout the day in line with the baseline. The AWR baselines from 10*g* have been extended in 11*g* to allow automatic creation of baselines for use in other features. A rolling week baseline is created by default.

Automatic Diagnostic Repository (ADR)—When critical database errors are detected (based on the DBA's definitions),

they automatically create an "incident" object, and the DBA is notified with details on their "incident." The new Incident Packaging Service (IPS) encapsulates all information about the incident, requests further tests and information if necessary, and allows you to send the whole package to Oracle Technical Support. (Source: Dr. Tim Hall.)

Automated storage load balancing—Oracle's Automatic Storage Management (ASM) now enables a single storage pool to be shared by multiple databases for optimal load balancing. Shared disk storage resources can alternatively be assigned to individual databases and easily moved from one database to another as processing requirements change.

Interval partitioning for tables—This is a new 11*g* partitioning scheme that automatically creates time-based partitions as new data is added.

Oracle SecureFiles—These replace LOBs that are faster than Unix files to read and write. OLAP analytic workspaces will potentially benefit greatly, as the LOBs used to hold AWs have historically been slower to write to than the old Express .db files. SecureFiles are said to be a huge improvement to BLOB data types, and they are faster with compression and encryption.

Oracle 11*g* Audit Vault—Oracle Audit Vault is a new feature that will provide a solution to customers' most difficult security problems today, protecting against insider threat and meeting regulatory compliance requirements.

Function-based data columns (FBC)—Oracle 11*g* has "virtual table columns," which are columns that are functions of other column values. For example, here we create column C3, which contains no data, just a derivation of the values of other columns. This is related to virtual indexes, indexes that are based on functions. Here is an example:

```
create table t1 (
c1 number,
c2 number,
c3 as (c1+c2) virtual
);
```

Developer Enhancements

Fully automatic SQL tuning—The 10*g* automatic tuning advisor makes tuning suggestions in the form of SQL profiles that will improve performance. You can tell 11*g* to automatically apply SQL profiles for statements where the suggested profile performs three times better than the existing query plan. The performance comparisons are done by a new administrative task during a user-specified maintenance window.

Improved SQL Access Advisor—The 11*g* SQL Access Advisor gives partitioning advice, including advice on the new interval partitioning. Interval partitioning is an automated version of range partitioning, where new equal-size partitions are automatically created when needed. Both range and interval partitions can exist for a single table, and range-partitioned tables can be converted to interval-partitioned tables.

Oracle 11*g* XML enhancements—Oracle 11*g* will also support Content Repository API for Java technology (JSR 170). Oracle 11*g* has XML "duality," meaning that you can also embed XML directives inside PL/SQL, and embed PL/SQL inside XML code. Oracle 11*g* XML will also support schema-based document type definitions (DTDs), to describe internal structure of the XML document.

PL/SQL New Features

PL/SQL intra-unit inlining—In C programming, you can write a macro that gets dynamically included "inline" whenever it's called, a great way to provide reusable code snippets. In Oracle 11*g*, PL/SQL stored procedures are eligible for inlining if Oracle thinks it will improve performance. No change to your code is required. Now you have no reason to not make everything a subroutine. (Source: Lewis Cunningham.)

> ***"Now you have no reason for not making everything a subroutine!"***

Scalable PL/SQL—The next scalable execution feature is automatic creation of "native" PL/SQL (and Java code), with just one parameter for each type with an "on/off" value. This apparently provides a 100% performance boost for pure PL/SQL code, and a 10–30% performance boost for code containing SQL. (Source: Mark Rittman.)

Improved PL/SQL stored procedure invalidation mechanism—A new 11g feature will be fine-grained dependency tracking, reducing the number of objects that become invalid as a result of DDL.

Easy PL/SQL compiling—Native compilation no longer requires a C compiler to compile your PL/SQL. Your code goes directly to a shared library. (Source: Lewis Cunningham.)

Stored procedure named notation—Named notation is now supported when calling a stored procedure from SQL.

Proxy connect for SQL*Plus—The "connect" command has been enhanced to allow for a "proxy," to aid applications that always connect with the same user ID. Here is an example.

```
connect sapr3[scott]/tiger
```

Concluding Remarks

The above short list of new features of Oracle 11*g* was gleaned from announcements by Oracle Corporation at Oracle OpenWorld 2006. There will be continuous changes as Oracle releases more information. I track upcoming 11*g* changes, so visit my site, **www.dba-oracle.com**, for updates. ▲

Don Burleson has been a full-time DBA for 25 years and has authored or coauthored 34 books on Oracle database management, including five Oracle Press books.

My Favorite Hints

by Chris Lawson

Chris Lawson

When the dog bites/When the bee stings/When I'm feeling sad/
I simply remember my favorite things/And then I don't feel so bad.
—*Maria in* The Sound of Music

I *consider facility with SQL Hints to be of crucial importance to the serious performance specialist.* When I help a client screen prospective performance DBAs, I always ask questions about common SQL Hints, such as USE_HASH, or ORDERED. Here's why: As the size of database tables grow, it becomes more and more likely that good performance will require a SQL Hint. It's not that the optimizer works badly for big databases; rather, the consequences just become much more severe for being a little bit wrong.

> ***"I consider facility with SQL Hints to be of crucial importance to the serious performance specialist."***

Not knowing SQL Hints will seriously handicap anyone trying to improve performance—especially in large databases. For instance, when joining huge tables in a data warehouse, a small mistake in join order can add hours to the runtime of a batch job. As the number of tables in the join increases, the importance of the join order (as well as the join method) is magnified.

There's another reason why Hints are important. For critical batch jobs, it's important that we achieve consistency in runtimes. Our customers simply can't tolerate huge variations in runtimes. Adding a SQL Hint is one way to achieve reasonably consistent runtimes. So, for example, if you know that a full table scan is the right choice most of the time, it's reasonable to lock in that execution plan.

My Favorite Hints

An SQL Hint is often appropriate when you know something special about the data distribution or bind variables that gives you "inside information." In these cases, you really "know better" than the optimizer. With a simple Hint, you can instruct the optimizer to choose your plan—even if the optimizer doesn't really like your choice.

There are a few Hints that I use frequently: LEADING, ORDERED, NO_MERGE, USE_HASH, FULL, and PARALLEL. Note that most of these are related to joins—this is where the optimizer often needs help. Of course, some hints don't really correct the optimizer, but are used for special purposes—the PARALLEL Hint is one example.

SQL Hints have the advantage that you are not really changing the *functionality* of the program. In a large production system, any change that smacks of functionality modification prompts time-consuming QA checks. Most QA groups understand that a SQL Hint doesn't change functionality. (Of course, you can still wreck performance if you're not careful.)

> ***"Not knowing SQL Hints will seriously handicap anyone trying to improve performance."***

Before we look at each Hint, let's review a few critical points that are often stumbling blocks to newer DBAs. Firstly, whenever the Hint uses a table name as a parameter, remember to actually specify the table *alias* if one has been used, not the actual table name. I have encountered numerous cases where the SQL Hint actually did nothing, because the DBA had mistakenly specified the table name instead of the table alias.

Second, be sure that the Hint comes immediately after the "SELECT" keyword. For example, if you need to extract distinct values, put the keyword "DISTINCT" *after* the Hint.

LEADING and ORDERED

> ***"Of all the SQL optimization issues, join order probably ranks near the top in importance."***

Of all the SQL optimization issues, join order probably ranks near the top in importance. There are several good ways to control join order. If you simply need to begin the join with a particular table, the Hint "LEADING(Table alias)" is a good choice. Whenever I need to explicitly order all tables in the join, I use ORDERED, and then rearrange the order of the tables in the "FROM" clause.

NO_MERGE

This Hint is not as well known as some others, but it can be very useful. Here's why: The Oracle optimizer will often try to merge, or "blend in," views or inline views into the main body of the SQL. This is actually not a bad idea, and generally works fine, but not always. In particular, you may know something about the data distribution or bind variables that the optimizer doesn't know.

For instance, suppose that our SQL includes a query from an inline view that simply and drastically reduces the result set to just a few rows. Clearly, then, you will want to perform the inline view first—as a standalone computation. Then, you can proceed to the remaining tables. An easy way to accomplish this is to use the Hint "NO_MERGE(View alias)."

USE_HASH

In addition to modifying the join order, fixing the join *method* is another critical optimization technique. This frequently involves an analysis of nested loop joins versus hash joins. Recall that if you have a large result set, you generally want to perform a hash join, not a nested loop join. This makes sense because a hash join, even though it normally requires a full table (or index) scan, avoids excessive index scanning and table access associated with nested loops.

In data warehouses, it's common to specify both the "USE_HASH" Hint and the "ORDERED" Hint. This way we ensure the proper join method, as well as the best join order.

FULL

"Although some DBAs are horrified at the prospect of a full table scan, in practice, it's often the right thing to do."

Although some DBAs are horrified at the prospect of a full table scan, in practice it's often the right thing to do. In particular, using an index is usually a bad idea if it means a huge number of index and table accesses. In data warehouses, for instance, it's very common to scan an entire partition, often using parallelism to speed things up. In these cases, we might see the Hint "FULL(Table alias)."

If you do opt for a full table scan, remember to set the DB_FILE_MULTIBLOCK_READ_COUNT to exploit the server's multiblock read efficiency. Most Unix servers can read a total of 1 MB at a time in multiblock reads. Therefore, if you use an 8 KB block size, a value of 128 for DB_FILE_MULTIBLOCK_READ_COUNT is the right choice. (You can set this at the session level, if necessary.) Remember also that Oracle will not continue a multiblock scan across the extent boundary, so make sure your extents aren't sized ridiculously small.

PARALLEL

This Hint isn't really an optimization technique so much as a resource utilization tactic. Unfortunately, parallelism is frequently misused. Keep in mind that quickly throwing more resources at a problem before optimizing is very unwise. Invoking parallelism should never be done until the SQL runs as efficiently as possible—including the best join order, join method, indexes, etc. Once you're sure the SQL is reasonably efficient, by all means consider parallelism.

Always remember that there is a cost to invoking parallelism—you are really taking away resources from everyone else! Furthermore, remember the law of diminishing returns; you may find, for instance, that a parallel degree of 6 greatly speeds up table scanning, whereas a parallel degree of 8 only ekes out a bit more throughput.

I occasionally see programs that specify a degree of parallelism so unrealistic that it consumes all CPU or disk I/O capacity on the server. As a general rule of thumb, on servers having 20 CPUs, I have found a parallel degree of 6 often works nicely. Extreme degrees of parallelism probably don't provide enough extra throughput to justify the extra resource cost.

"Always remember that there is a cost to invoking parallelism—you are really taking away resources from everyone else!"

Surprisingly, in some cases, parallelism can be used in cases that don't even perform full table or index scans. In these cases, parallelism is used as a quick and easy way to invoke *multi-threading*. This is not too difficult to set up, but there are some restrictions. See **oraclemagician.com/mag/magic11.pdf**.

Concluding Remarks

Remember—SQL Hints are powerful and often necessary, but using them incorrectly can be devastating. On one application I tuned, almost all the SQL Hints were inappropriate, and my first step in tuning was simply to remove the Hints. When applying a SQL Hint, remember that you are assuring the optimizer that you know better than it does. The optimizer is willing to trust you, but you take on full responsibility for the consequences—good or bad. So, understand what a Hint does before you slap it on. *No Hint is better than the wrong Hint.* ▲

"SQL Hints are powerful and often necessary, but using them incorrectly can be devastating. No Hint is better than the wrong Hint."

Chris Lawson is an Oracle DBA consultant in the San Francisco Bay Area, where he specializes in performance tuning of data warehouse and financial applications. He is a frequent speaker at NoCOUG, and has written for a number of publications, such as Oracle Internals, Exploring Oracle, SELECT, Oracle Informant, *and* Intelligent Enterprise. *Chris has held a variety of positions in the IT field—ranging from systems engineer to department manager—and is an instructor for the University of Phoenix. He can be contacted via* **www.oraclemagician.com**.

Bag o' Tricks

by Danny Chow

Danny Chow

Even with RMAN gaining popularity as a database backup solution, a lot of time-tested backup scripts are still in use. Be it a cold or hot backup, a trace file should always be created from the control file; however, finding the trace file in the udump directory is more or less a wild goose chase in some backup scripts that I've come across. Some scripts depend on the file creation timestamp, while some search for the string "CREATE CONTROLFILE" in the trace files. My backup script used to issue "alter database backup controlfile to trace" and "ls -1tr | tail -1" commands. Assuming that no other trace files are generated in the interval between these two commands, the trace file that we need will be correctly identified. This assumption served me well until I encountered a rare case where another session ended at the precise moment between these two commands. Knowing that the name of the trace file is associated with the session SPID, I developed a more rigorous method to find it.

```
set serveroutput on
set linesize 132
set verify off
set feedback off
spool backup_ctl_trc.sh

DECLARE

 v_trace_id       VARCHAR2(12);
 v_dump_dir       VARCHAR2(80);
 v_ctl_file_mask  VARCHAR2(80) default
'_ora_xxxxx.trc';
 v_db_name        VARCHAR2(80);
 v_backup_dir     VARCHAR2(80) default '&1';

BEGIN

 SELECT value
 INTO   v_dump_dir
 FROM   v$parameter
 WHERE  name = 'user_dump_dest';

 SELECT value
 INTO   v_db_name
 FROM   v$parameter
 WHERE  name = 'db_name';

 v_ctl_file_mask := v_db_name || v_ctl_file_mask;

 SELECT B.spid
 INTO   v_trace_id
 FROM   v$session A, v$process B
 WHERE  A.audsid = userenv('SESSIONID')
 AND    A.paddr = B.addr;

 EXECUTE IMMEDIATE 'alter database backup controlfile
to trace';

 dbms_output.put_line('cp ' || v_dump_dir || '/' ||
   replace(v_ctl_file_mask,'xxxxx',v_trace_id) ||
        ' ' || v_backup_dir || '/ctl_file.trc');
END;
/
spool off
HOST . ./backup_ctl_trc.sh
HOST rm ./backup_ctl_trc.sh
exit;
```

Beginning with Oracle 8*i*, Oracle has provided the means to eliminate the uncertainty, but the time-tested backup scripts have been running so well, most DBA have no pressing need to update them. If your database uses Oracle 8i or higher versions, a prefix of your choosing can be defined. By using "alter session" and "backup controlfile" command, the trace file name will use the prefix you specify, as shown in the following example.

```
SQL> alter session set tracefile_identifier=
CTL_TRACE;

Session altered.

SQL> alter database backup controlfile to trace;

Database altered.

SQL> exit

UNIX>ls /opt/app/oracle/admin/TESTDB/udump/*CTL_TRACE*
/opt/app/oracle/admin/TESTDB/udump/testdb_ora_
17473_CTL_TRACE.trc
```

If your database uses Oracle 9*i* or higher versions, you can create a trace file with a name and location of your choosing, as shown in the following example.

```
SQL> alter database backup controlfile to
trace as '/orabackup/testdb/ctl_trace.txt';

Database altered.

SQL> exit

UNIX>ls  /orabackup/testdb/ctl_trace.txt
/orabackup/testdb/ctl_trace.txt
```

In summary, the "tracefile_identifier" method will work on 8*i*, 9*i*, and 10*g* databases, while the "backup controlfile to trace as" method will work with 9*i* and 10*g* databases. ▲

Danny Chow is an independent Oracle consultant with many years of DBA and development experience, and holds certifications in Oracle 7, 8, 8i, and 9i. His email address is **dannychow@earthlink.net**.

Spring Conference Abstracts

	Auditorium	Executive Conference Room	X7 Conference Room
10:45 a.m.–11:45 a.m.	**Optimizing SQL in Your PL/SQL Programs** **Steven Feuerstein, Quest Software** Learn how to optimize both the performance and maintainability of your SQL statements in PL/SQL programs. We will cover the most important PL/SQL-specific features for SQL optimization, including FORALL, BULK COLLECT, and table functions.	**Wait Time–Based Oracle Performance Management** **Matt Larson, Confio Software** There are many ways to use Oracle wait events for performance tuning of an Oracle database, but often there is confusion on exactly what the data means. This presentation will review several real-life case studies of using wait event data coupled with Wait-Time based performance analysis to solve the most difficult performance-related issues.	**The Hunt for Killer Skew** **Iggy Fernandez, Verizon Business** Cary Millsap has sounded the alarm about killer "skew" that prevent us from discerning the performance problems through the fog of Statspack and AWR data. In this presentation we will refine a method (based on LOGOFF triggers), due to Richmond Shee, that helps us find killer skew, and we will illustrate the method with a real-life example.
12:45 p.m.–1:45 p.m.	**Writing Maintainable Code** **Steven Feuerstein, Quest Software** Building modular, reusable code doesn't happen by chance. In this presentation you will learn how to take advantage of packages, local subprogram units, cursor variables, table functions, autonomous transactions, and more to craft small, reusable units of code that are easy to maintain and test.	**EDITOR'S PICK** **AWR: Going Beyond the Scripts Supplied by Oracle** **Jerry Brenner, Guidewire Software** There's a wealth of information contained in the AWR tables. At our company, we generate a number of custom reports as part of every performance test run against one of our applications. This presentation will go over some of these reports and how we use them to tune our applications and improve our performance testing.	**Ultimate Guide for Oracle Storage Administration** **George Trujillo, Jr., Trubix Inc.** As Oracle storage solutions continue to evolve, it is important for all IT individuals working with storage to understand the pros and cons of different storage options. Attendees will learn the fundamentals of Oracle Automatic Storage Management and Oracle Clusterware from a storage perspective, not just an Oracle perspective.
2:15 p.m.–3:15 p.m.	**Oracle 2020: A Look at How Upcoming Trends and Changes Will Affect Your Life** **Donald Burleson, Burleson Consulting** The Oracle professional of the 21st century will be relieved of the tedium of monitoring and tuning, and will be free to concentrate on other important database administration activities. This fun and interesting presentation will give attendees a look into how their jobs are going to change, sooner than they think.	**Diagnosing Bottlenecks in Oracle Streams** **Chris Lawson and Brian Keating** Troubleshooting Streams replication issues can be very complicated, because there are lots of places to look when things don't go right. In this presentation we tackle the bewildering task of monitoring and correcting Streams problems. They suggest various ways to monitor the Capture, Propagate, and Apply processes, and what to do when things go wrong.	**Managing the Data Exchange Relationship** **Michael Scofield, ESRI, Inc.** Anytime data is exchanged between distinct and dissimilar organizations, a relationship exists, with a variety of duties, expectations, and technical issues, and this relationship must be managed. When one party seeks to acquire data from another, a host of issues must be addressed unambiguously. Architectural differences, architectural stability, scope, data quality, replication techniques, permissible usage, and liabilities all must be addressed. Of particular concern are incremental extractions of data after the first big bulk transfer. It is crucial to understand the difference between updates and corrections. It is also important, if the incremental transfers occur over a longer period of time, to detect changes in scope, architecture, meaning, and quality of the incremental updates.
3:45 p.m.–4:45 p.m.	**Inside the Oracle 10*g* Cost-Based SQL Optimizer** **Donald Burleson, Burleson Consulting** With each new release of Oracle, the cost-based SQL optimizer (CBO) becomes more intelligent, and Oracle10*g* is no exception. This presentation shows the major enhancements to the cost-based SQL optimizer in Oracle10*g*. We will also cover the use of the new Oracle 10*g* SQL profiles and discuss cursor sharing and automatic query rewrite.	**RAC for Beginners: The Basics** **Dan Norris, IT Convergence** Oracle Real Application Clusters (RAC) has been steadily gaining momentum in the market, with new and old customers considering RAC implementations. This session will focus on the technical architecture of the RAC feature, with an emphasis on areas that are frequently misunderstood.	

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, Lisa Loper, at **lloper@dbspecialists.com**. ▲

Long-term event sponsorship:

- LOCKHEED MARTIN
- CHEVRON
- ORACLE CORP.
- PG&E

Thank you! Year 2007 Gold Vendors:

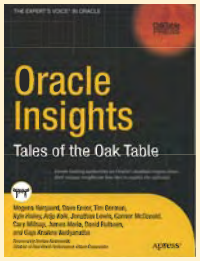
- BEZ Systems
- Confio Software
- Database Specialists, Inc.
- IT Convergence
- Network Appliance
- Princeton Softech
- Quest Software
- Roundstone Systems

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

TREASURER'S REPORT

Jen Hong, *Treasurer*

Beginning Balance		
January 1, 2007		**$ 41,944.67**
Revenue		
Membership Dues	18.913.00	
Meeting Fees	700.00	
Vendor Receipts	4,000.00	
Miscellaneous	100.00	
Interest	39.83	
Total Revenue		**$ 23,752.83**
Expenses		
Regional Meeting	6,351.64	
Journal	3,735.15	
Membership	289.73	
Administration	37.84	
Website	–	
Board Meeting	562.59	
Marketing	65.19	
Insurance	–	
Vendors	58.30	
P.O. Box	–	
Total Expenses		**$ 11,100.44**
Ending Balance		
March 31, 2007		**$ 54,597.06**



Oracle Performance Training Day with Kyle Hailey

The Training You *Need* at a Price You Can *Afford!*

Is there a *single* metric that summarizes database performance in a nutshell? If you knew what it was, you would watch it like a hawk, hour after hour and day after day, and you would use it to measure the effect of hardware upgrades, software upgrades, and other configuration changes. All your tuning efforts would be directed toward improving the number, and you would want to determine the effect of each user, each program, each application server, each table, and each SQL statement on the number.

If your database suddenly becomes sluggish, can you tap out a few commands and immediately pinpoint the culprit user, program, application server, table, or SQL statement? Can you do the same thing if the database was sluggish earlier in the day when you were not present?

> *If you want to learn how to quickly solve Oracle performance problems, come take advantage of a full day of training by Kyle Hailey on August 17 at the Carr America Conference Center in Pleasanton for only $250—register by July 1 to receive an early bird price of $195. The price includes continental breakfast, lunch, and a free copy of* Oracle Wait Interface: A Practical Guide to Performance Diagnostics & Tuning*, by Richmond Shee et al. It's the training you* need*, at a price you can* afford!

Kyle Hailey is a regular speaker at NoCOUG conferences and needs very little introduction to NoCOUG members. He is a founding member of the Oak Table and co-author of *Oracle Insights: Tales of the Oak Table*, together with other NoCOUG regulars like Jonathan Lewis, Cary Millsap, and Gaja Krishna Vaidyanatha. He had a long and distinguished career at Oracle and designed the performance tuning module in Oracle Enterprise Manager 10*g*. Take a look at the topics he will cover on August 17.

1. **Proving the database is running smoothly**—The power of proving that the database is running smoothly is often underestimated in performance problem situations. If the database is running smoothly, then the problem is elsewhere, perhaps in the application code. Kyle will explain how to determine if there is a bottleneck in the database, or if the database is running smoothly, in three easy steps.
2. **Performance data collection tools**—How to collect the necessary data to analyze the performance of an Oracle database. An overview of performance data collections tools, including both Statspack and AWR, and practical examples of their use.
3. **Oracle wait events**—Oracle wait events are central to any good performance tuning methodology but are often misunderstood and undocumented. Kyle will explain the causes, show how to analyze, and provide solutions for the top wait events that represent 99% of the wait time in most databases.
4. **Top five wait events**—A step-by-step analysis and solution for the top five wait events. From a Statspack report to the final solution, see each step of analyzing and solving these wait events using real-life examples.
5. **Easy wait events**—One-step solutions when the performance data identifies these wait events as the problem.
6. **Difficult wait events**—Methods for analyzing and solving wait events when Statspack or AWR data are not enough.
7. **Sampling techniques**—The only way to solve the majority of wait bottlenecks is with a technique called sampling. Sampling can be done on any version of Oracle from version 7 onward, but is automated in Oracle 10*g* with the Active Sessions History feature (ASH).
8. **Active Session History**—Active Session History, often abbreviated as ASH, revolutionizes performance tuning by providing an unprecedented depth of analysis. See how the power of ASH can improve your life.
9. **ASH report**—The performance tool that is probably the least used, but is the most powerful one available.
10. **Oracle OEM 10*g***—How the Oracle OEM 10*g* performance interface radically simplifies performance tuning and unifies all of the above techniques in one simple interface.

Register at www.nocoug.org. Seating is limited, so register early!

Teamwork at NoCoug

Back row, left to right: Jen Hong—Secretary and Treasurer; Lisa Loper—President; Roger Schrag—Director of Conference Programming; Eric Hutchinson—Webmaster; Randy Samberg—Track Leader; Hamid Minoui—Training Day Coordinator; Naren Nagtode—Director of Marketing; Joel Rosingana—Director of Membership; Darrin Swan—Vice President. Front row, left to right: Iggy Fernandez—Journal Editor; Nora Rosingana—NoCOUG Staff Member. Not pictured: Hanan Hit—Track Leader; Diane Lee—Vendor Coordinator and IOUG Representative.

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 all of us working together as a team is what makes for great conferences, training days, and other benefits.

But volunteering your time is far from being 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, and so on—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, book review, or cover photo to the *NoCOUG Journal*
- Join the *Journal*'s technical review team
- Give a presentation at a conference
- Help with registration and other chores on conference day
- Tell your colleagues about NoCOUG and the benefits of membership

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 email to **board@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 Membership Director
and Past President
Independent Consultant

"The friendships and professional relationships I have enjoyed during the last six years serving as a Board member ranks #1 on my list of reasons why I volunteer for NoCOUG. Every member of our organization from each Board member, to you our Oracle Users membership, makes NoCOUG a great organization to be a part of. The teamwork that goes into planning each event has become a well-oiled machine as we work diligently to deliver valuable Oracle content to our members. I have watched many of you stay loyal to NoCOUG during the challenging times and watched as you found new opportunities, myself included. Thank you for making NoCOUG an exciting family to be a part of. When it comes down to it, the heartbeat of NoCOUG is you."

–Darrin Swan
NoCOUG Vice President
Quest Software

NoCOUG Spring Conference Schedule

May 17, 2007, at Lockheed Martin, Sunnyvale, CA

Please visit **www.nocoug.org** for updates and directions, and to submit your RSVP.
Cost: $40 admission fee for nonmembers. Members free. Includes lunch voucher.

8:00 a.m.–9:00	**Registration and Continental Breakfast**—Refreshments served
9:00–9:30	**General Session and Welcome**—Lisa Loper, NoCOUG president
9:30–10:15	**Keynote:** *Fusion: The New Frontier—Oracle Administration in the Future* —Steve Lemme, Computer Associates
10:15–10:45	**Break**
10:45–11:45	**Parallel Sessions #1** **Auditorium:** *Optimizing SQL in Your PL/SQL Programs* —Steven Feuerstein, Quest Software **Executive Conference Room:** *Wait Time–Based Oracle Performance Management* —Matt Larson, Confio Software **X7 Conference Room:** *The Hunt for Killer Skew*—Iggy Fernandez, Verizon Business
11:45 a.m.–12:45 p.m.	**Lunch**
12:45–1:45	**Parallel Sessions #2** **Auditorium:** *Writing Maintainable Code*—Steven Feuerstein, Quest Software **Executive Conference Room:** *AWR: Going Beyond the Scripts Supplied by Oracle* —Jerry Brenner, Guidewire Software **X7 Conference Room:** *Ultimate Guide for Oracle Storage Administration*—George Trujillo, Jr., Trubix Inc.
1:45–2:15	**Break and Refreshments**
2:15–3:15	**Parallel Sessions #3** **Auditorium:** *Oracle 2020: A Look at How Upcoming Trends and Changes Will Affect Your Life* —Donald Burleson, Burleson Consulting **Executive Conference Room:** *Diagnosing Bottlenecks in Oracle Streams*—Chris Lawson and Brian Keating **X7 Conference Room:** *Managing the Data Exchange Relationship (Part I)*—Michael Scofield, ESRI, Inc.
3:15–3:45	**Raffle**
3:45–4:45	**Parallel Sessions #4** **Auditorium:** *Inside the Oracle 10g Cost-Based SQL Optimizer*—Donald Burleson, Burleson Consulting **Executive Conference Room:** *RAC for Beginners: The Basics*—Dan Norris, IT Convergence **X7 Conference Room:** *Managing the Data Exchange Relationship (Part II)*—Michael Scofield, ESRI, Inc.
5:00–??	**NoCOUG Networking and Happy Hour at Faz, 1108 North Mathilda Avenue, Sunnyvale**

Session descriptions appear on page 24.

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

The NoCOUG Journal design and production: giraffex inc./sf.

NoCOUG
P.O. Box 3282
Danville, CA 94526

RETURN SERVICE REQUESTED