TechJournal

New York Oracle Users Group

Second Quarter 2012

Second Quarter General Meeting

Tuesday, June 5, 2012
St. John's University – Manhattan Campus
101 Murray Street

Sponsored by:
F5 and Quest Software

Free for Paid 2012 Members
Don't Miss It!

In This Issue –

www.nyoug.org 212.978.8890

Two Million Database Professionals Count on One Solution.

Simply the best for Oracle database professionals - Toad 11. Supported by over a decade of proven excellence, only Toad combines the deepest functionality available, extensive automation, and a workflow that enables database professionals of all skill and experience levels to work efficiently and accurately. Countless organizations empower their database professionals with Toad. The best just got better.

Watch the video at **www.quest.com/Toad11SimplyBetter**.

QUEST SOFTWARE®
Simplicity At Work™

NYOUG Officers / Chairpersons

Table of Contents

New York Oracle Users Group, Inc.
#0208
67 Wall Street, 22nd floor
New York, NY 10005-3198
(212) 978-8890

General Meeting – June 5, 2012 Agenda

sponsored by F5 and Quest Software

AGENDA

Time	Activity	Track/Room	Presenter
8:30-9:00	**REGISTRATION AND BREAKFAST**		
9:00-9:30	**Opening Remarks** **General Information**	(single session) Auditorium	Michael Olin NYOUG President
SESSION 1 9:30-10:30	**KEYNOTE: Exadata: A Critical Review**	(single session) Auditorium	Mike Ault Texas Memory Systems
10:30-10:45	**BREAK**		
SESSION 2 10:45 -11:45	**New Enterprise Manager 12c** **Database Management Features**	DBA Auditorium	Mughees Minhas Oracle Corporation
	Dynamic SQL in the 11g World	Developer Room 118	Michael Rosenblum Dulcian, Inc.
SESSION 3 11:45 -12:30	**Ask the Experts Panel**	(single session) Auditorium	Michael Olin Moderator
12:30 -1:30	**LUNCH - ROOM 123**		
SESSION 4 1:30-2:30	**Validating Your I/O System:** **Coming Out of the Black Box**	DBA Auditorium	Mike Ault Texas Memory Systems
	Do-It-Yourself Data Migration	Developer Room 118	Dr. Paul Dorsey Dulcian, Inc.
2:30-2:45	**BREAK**		
SESSION 5 2:45-3:45	**Under the Hood of Oracle ASM: Fallibility Analysis**	DBA Auditorium	Alex Gorbachev Pythian
	Building Web Data Dashboard Without Coding	Developer Room 118	Dana Singleterry Oracle Corporation
3:45-4:00	**BREAK**		
SESSION 6 4:00-5:00	**The Essentials of Data Discovery:** **Do You Know Where Your Data Is?**	DBA Auditorium	Dhan Patel IBM Software

ABSTRACTS

9:30-10:30 AM KEYNOTE: Exadata: A Critical Review

This presentation provides a critical review of the Exadata platform covering the strengths and weaknesses of the Exadata system and its software. The special Exadata features are discussed including when they can and can't be used. The Exadata is also compared to a COTS based system that is less expensive and more powerful.

Mike Ault has been working with Oracle since 1990. Mike has written or co-written over 24 books on Oracle technology as well as dozens of articles and presentations. Mike is a frequent presenter at international, national and regional conferences.

10:45-11:45 AM DBA TRACK: New Enterprise Manager 12c Database Management Features

This session will focus on advanced database performance analysis techniques and describe recently introduced features for Exadata management and monitoring. Oracle Enterprise Manager 12c has many new database management features for the advanced DBA. Mughees will discuss new key features that help maximize database performance, enable smart capacity planning, improve data security, and provide the basis for smart and effective database administration.

Mughees Minhas is responsible for all aspects of product management relating to database, systems and quality management solutions at Oracle Corporation. He has more than 17 years of experience working with Oracle products with special interests in the areas of application testing and quality, performance diagnostics and tuning, and SQL optimization. Mughees is a frequent speaker at various Oracle forums and has authored several papers on the internal workings and usage best practices of a wide variety of Oracle technologies. Mughees holds an undergraduate degree in Electrical Engineering from Washington University and an MBA from Columbia Business School. He is currently senior director of product management in Oracle's Server Technologies division.

10:45-11:45 AM DEVELOPER TRACK: Dynamic SQL in the 11g World

Dynamic SQL is one of the most critical tools in the developer's toolbox. It empowers well-written systems to rapidly realign with the real world. In addition to the core elements required to properly utilize Dynamic SQL, this presentation focuses on effective and appropriate usage of all 11g additions to the PL/SQL language (in conjunction with other advanced features such as object types, collections, large objects etc). Real-life examples are used as illustrations for all key points.

Michael Rosenblum is a Development DBA at Dulcian, Inc. He is responsible for system tuning and application architecture. He supports Dulcian developers by writing complex PL/SQL routines and researching new features. Mr. Rosenblum is the co-author of *PL/SQL for Dummies* (Wiley Press, 2006). Michael is a frequent presenter at various regional and national Oracle user group conferences. He won the ODTUG 2009 Best Speaker Award. In his native Ukraine, he received the scholarship of the President of Ukraine, a Masters Degree in Information Systems, and a Diploma with Honors from the Kiev National University of Economics.

1:30-2:30 PM DBA TRACK: Validating Your I/O System: Coming Out of the Black Box

This presentation covers the techniques used to validate your I/O subsystem for IOPS, bandwidth and latency to provide a fingerprint that can then be used to check new systems for improvement or change.

Mike Ault has been working with Oracle since 1990. Mike has written or co-written over 24 books on Oracle technology as well as dozens of articles and presentations. Mike is a frequent presenter at international, national and regional conferences.

1:30-2:30 PM DEVELOPER TRACK: Do-It-Yourself Data Migration

Data profiling tools are expensive and provide little value beyond what you could be written in a day. ETL tools are complex, VERY expensive and built to support extremely complex migration environments. Data migration is the hidden nightmare of a project. Incorrectly managed, data migration can consume more resources than development. This presentation provides a roadmap for data migration, from data profiling to final loading and incremental data updates. Several projects that had to move hundreds GBs of information will be discussed. A simple mapping repository that was used to generate 100% of all mapping code will also be demonstrated.

Dr. Paul Dorsey is president of Dulcian, Inc. an Oracle consulting firm specializing in business rules and web-based application development and chief architect of Dulcian's BRIM® tool. Paul co-authored 7 Oracle Press books on JDeveloper, UML Modeling, and Oracle database tools as well as *PL/SQL For Dummies*. He is an Oracle Ace, past IOUG and ODTUG volunteer of the year and an Oracle 9i Certified Master. Dr. Dorsey's submission of a Survey Generator built to collect data for The Preeclampsia Foundation was the winner of the 2007 Oracle Fusion Middleware Developer Challenge and Oracle selected him as the 2007 PL/SQL Developer of the Year.

2:45-3:45 PM DBA TRACK: Under the Hood of Oracle ASM: Fallibility Analysis

Oracle Automatic Storage Management (ASM) has introduced a new concept of mirroring that is implemented differently than in any known RAID levels. So what happens when not one but two or more disks fail? Is such situation a hypothetical and highly unlikely? This session will help attendees to evaluate the data loss risks and adopt the best ASM configuration according to their risk profile. For a better understanding of ASM reliability features, Alex will peek under the hood of ASM and provide live demos simulating ASM disk failures and ASM handling of such failures.

Alex Gorbachev is an Oracle ACE Director and member of OakTable Network. Alex is sought after speaker at the user conferences around the world. Currently Alex is the CTO at Pythian and the search for the perfect fit between technology, engineering talents and business is what keeps him up at night.

2:45-3:45 PM DEVELOPER TRACK: Building Web Data Dashboard Without Coding

Learn how to build an enticing data dashboard rich in features utilizing Oracle ADF Data Visualization Components. Spice up your applications with graphs, maps, gauges and more to help your customer understand the data in their database. See how you can get from data in the database to a full blown Web-based dashboard in less than 45 minutes of development in this live development demo-driven session. See how Oracle ADF Faces helps you create astonishing UIs with visual and declarative development.

Dana Singleterry is a Principal Product Manager in the Oracle Development Tools group, specializing in JDeveloper, ADF, ADF Faces, Maven Integration, Lifecycle Support and Web Services. He has been working at Oracle since 2004, initially in Java EE Consulting, as a Product Manager in the Application Server group and currently as a Product Manager in the Development Tools group. Dana has published in trade journals, is the author of many Oracle How-To's and actively contributes to the development of JDeveloper and ADF. Dana is a frequent presenter at user groups and industry conferences both nationally and internationally, including various Oracle user groups and Oracle Open World / Oracle Develop.

4:00-5:00 PM DBA TRACK: The Essentials of Data Discovery: Do You Know Where Your Data Is?

Participants will learn why data discovery has become a recommended best practice by analysts and industry leaders to help ensure data quality and governance. They will also learn best practices to execute data discovery projects and how to evaluate solutions, whether vendor-built or internally-built, for discovery. Data Discovery is of fundamental importance to a wide range of projects, including business intelligence, data integration through master data management, data governance, and data archiving. This session explains what data discovery is, and why it has become such an important technology for organizations implementing enterprise data governance and data management strategies. Dhan will review the requirements for data discovery software and how data discovery technology should fit within the suite of tools needed to govern and manage data across your enterprise, regardless of where it is located.

Dhan Patel is a Sr. TPM within IBM's Software Group, specializing in data management for Oracle applications. He has spoken at numerous Oracle, OAUG, Quest International and IOUG events including Oracle OpenWorld, Collaborate and others. He has over a decade of experience in working with Oracle users to help optimize the management of their data.

Your Ad Here!

Vendors, place your advertisement in the NYOUG Tech Journal. Let our members know you want to do business with them.

Ad Options Available: Full Page – Black/White or Color
Half-Page – B/W only

Sponsorships: General Meeting – Primary and Secondary
Special Interest Group
Journal Ad only
Most sponsorship packages include color and/or black/white ads.

Message from the President’s Desk

Michael Olin

Summer, 2012

Leading from Behind?

A day or so before Oracle’s latest announcement, heralding the arrival of the Oracle Cloud, I was talking with an IT executive with whom I had worked side-by-side building Oracle Forms and Reports applications in the final years of the old millennium. We were discussing how so much of what we take for granted about IT today was one once a heretical idea with Oracle CEO Larry Ellison as its lone champion. With cloud computing, Oracle’s announcement this week seemed to be long overdue. Although Ellison started his presentation by explaining that Oracle’s decision to embrace the cloud was made some seven years ago, at this point, Oracle certainly doesn’t look like it was first out of the gate.

Expectations of Thought Leadership

As users of Oracle’s products, we have come to expect the company to be “first” with just about everything. Clearly, Oracle was the first commercial implementation of a SQL-based RDBMS. Other significant firsts (and some near-firsts) included multi-platform support, read-consistency, client-server, distributed queries, row-level locking and support for both XML and Java in the database. For decades now, Oracle has been quick to develop and deploy products that embrace the most talked about concepts and theories not only in the database space, but now, computing in general. Of course, there are also notable cases where Oracle was seemingly left far behind. When GUI-based application development tools first started appearing, Oracle developers were still tweaking their .inp files in text editors and generating character-mode applications. As more and more applications for Oracle database back-ends were developed using tools like PowerBuilder, Oracle rushed out a barely functional GUI-based Oracle Forms version 3 and didn’t really get things right until version 4.5.Oracle alum Marc Benioff pioneered SaaS (software as a service), pointing the way to a myriad of hosted applications and the ongoing move to cloud computing with salesforce.com long before Oracle started moving in the same direction. However, when it comes to the “big picture”, Oracle seems to be consistently in front of the pack.

Radical Ideas…

In the mid 1990’s, Oracle proposed the “Network Computer” (NC). These diskless workstations were meant to provide a low cost alternative to PCs when the cost of a modest windows box was more than $1,000. The NC would have a lightweight browser-based operating system and would retrieve whatever applications it needed from the network. Sun, IBM, and others briefly manufactured these devices, but they never really caught on, as PC prices continued to drop rapidly and internet connection speeds were still too slow to support the necessary application downloads. When Oracle announced Oracle8i in 1999, one of the features that Larry Ellison was most excited about was Oracle’s iFS, the Internet File System. He explained that it made no sense for organizations to be storing documents and media files on hundreds or thousands of Windows NT file servers. With iFS, all of those files could be stored directly in the database, with the Oracle RDBMS, centralizing all of an organization’s data in one place and providing a single mechanism for data access, backup, and recovery. While neither of these ideas really took off at the time, in hindsight, they could easily be viewed as harbingers of innovations we take for granted today.

…Whose Time Has Come

Today, diskless devices abound. From Netbooks to Smart phones to the iPad and other tablets, we assume that a wireless connection is all that we need to access either the corporate network or the public internet. If we need to install an application locally on our devices, we download it. But, just as often, the application itself runs somewhere else and the electronics in our hand simply provide for connectivity and a user interface. Local document storage is becoming a thing of the past. Now we store and share our work using services like Google Docs or Dropbox. We store our media in the cloud, choosing from vendors like Google, Amazon, or Apple. Many of these innovations can be traced back to a crazy idea that Oracle promoted years ago, yet Google and Apple are viewed as the real innovators. I suppose that's fitting, since Oracle began by building software to implement the ideas of E.F. Codd and others at IBM.

What comes next?

I plan to watch the entire hour-long Oracle Cloud announcement video again. I wouldn't be surprised to find a passing reference to another radical idea that just may change the face of computing. It will take some time before business fully embraces the cloud and until then, Oracle will continue to have a home in the corporate data center. While Oracle is solidly behind their Exadata offerings today, I expect change will be coming sooner than we expect. First, we may see spinning disks replaced by solid state devices. Then, we'll continue our migration to the cloud. It has been almost 15 years since Oracle first suggested that we store all of our data in one place and access it over the network with a relatively simple device. I don't think that at the time anyone thought that we'd be parking everything at Larry's house, but it sure looks like that's where we may be headed.

Upcoming Meeting Dates

ODTUG Kaleidoscope 2012

DATES: June 24-28, 2012
LOCATION: San Antonio, TX

Click here to register for Kaleidoscope 2012

--

SAVE THE DATE: NYOUG Fall 2012 General Meeting
DATE: Wednesday September 12,2012
LOCATION: St. John's University – 101 Murray St. New York, NY 10007

SAVE THE DATE: Special NYOUG Winter General Meeting
DATE: Wednesday December 12, 2012
LOCATION: New Yorker Hotel – 481 Eighth Ave (34th St.) New York, NY 10001

Oracle11g Upgrade Motivators: Adaptive Cursor Sharing

Dave Anderson and John Watson
SkillBuilders.com/Oracle

Introduction

We (the SkillBuilders Oracle team) think that there are four new performance-related “killer” features that most technologists will come to see as major drivers for upgrading to Oracle 11g:

1. SQL Plan Management
2. Results Cache
3. Parallel Processing Enhancements
4. Adaptive Cursor Sharing & Cardinality Feedback (based on Feedback Based Optimization)

In this article I will introduce Adaptive Cursor Sharing (ACS). I will provide a very brief review of SQL statement processing, then we’ll look at what problems Oracle is trying to solve with these features. I’ll talk a little bit about what were previous solutions, then we’ll take a look at the concept of feedback-based optimization, which provides or is the foundation of 11g Adaptive Cursor Sharing and cardinality feedback.

Review Optimization, Soft Parse and Hard Parse

To fully understand ACS, we must be familiar with how SQL statements are compiled. (You might call it parsed.) I’ll just give you a broader term – compiled; this includes soft parse and hard parse. Say, for example, you’ve coded “select column name, column name from table name where column = :x”. That statement has to be parsed and translated into an internal format.

Soft parse includes syntax check and semantic check. We have to know the semantics of the statement, who’s executing it, do you have proper privileges, etcetera. Soft parse also includes searching the library cache to see if the query plan is already stored, if it already has been optimized and, if so, hard parse can be avoided.

Hard parse is optimizing for the lowest cost execution plan and it does involve often an expensive step. By default, it compares up to 2,000 permutations if there are that many for a complex query.

Hard parse also stores the execution plan in the library cache, part of the shared pool. This requires latches (i.e. memory locks). Latches are a necessary evil - they avoid chaos in a multi-user system, but consume CPU time, decrease scalability and consume memory.

So let’s settle on the fact that hard parse is a high overhead operation. You have probably heard the motto, “Parse once, compile many.” Now, soft parse can also be avoided at times as well. We’ll learn, however, today that that might not be the best thing for ACS, which relies on soft parse.

Common Optimization Problems

What are the common problems with respect to optimization and what is Oracle trying to solve? One big problem is often encountered by mixing bind variables and data skew.

Bind variables are variables coded in the predicates (i.e. WHERE clause) of our SQL statements. Depending on the selectivity of the value bound to the variable at run-time, *you may have a need for different execution plans.* So, in the past, the developer and the DBA were always faced with a tough choice: Use bind variables for cursor re-use and adhere to our motto of, “Parse once, execute many,” and get reduced parsing and reduced memory use, or use literals for better execution plans.

The 9i solution to this problem is bind peeking. There are some issues there; I'll get back to that...
We also have a common problem of cardinality mis-estimates, particularly in intermediate results in a complex plan. So a sub-step might join together two tables or might process a view and come up with an intermediate result set that has to be joined to yet another table or view. If the optimizer estimates that the intermediate result set will contain 10 rows, it may choose a nested loops join; but at run-time the actual number is 1 million rows the nested loops may be terribly inefficient; if CBO knew that the result was 1 million rows it would have selected, say, a hash join. Oracle 10g has a solution - SQL profiling. However, again, there are some issues there.

Previous Solutions: Peeking, Stored Outlines and SQL Profiles

Dynamic Sampling was introduced with Oracle 9i and hasn't changed significantly since. That allows the optimizer to take a quick look at the data when it first parses a statement, take a quick look at the data and check things like cardinality and skew in the data, but it's a very quick look (i.e. not always accurate) and doesn't persist results (dbms_stats persists statistics in the catalog).
The other major feature feature that came with 9i was bind variable peeking to help when bind variables are used on skewed data. When you code literals:

```
SELECT * FROM EMP  WHERE SAL > 0
```

is a lot different from

```
SELECT * FROM EMP WHERE SAL > 1000000
```

If the data is skewed, Oracle will roduce a very different execution plan.
But with bind variables Oracle has no idea what you want. For example, assume a bind variable is used:

```
SELECT * FROM EMP  WHERE SAL > :B1
```

With peeking, Oracle takes the parse-bind-execute-fetch process and for the very first execution, reverses it to bind-parse-execute-fetch. Oracle expands the bind variable to see what you want, then develops an execution plan and that plan is then used for the next hundred million executions. And you better hope that that first bind variable had a value that is typical for the rest of its lifetime at that instance or at least until the plan is aged out and the statement is re-parsed.
Peeking helps a lot. It did give Oracle a lot more information than it ever had before. Now, it could also cause a major problem, though, that if that first variable was not typical, then your plans might be completely inappropriate for a long time subsequently.
The next facility to get around that, that predates bind variable peeking, was the Oracle 8i "stored outline", which is just a set of hints. So, rather that hinting it manually, you instruct the optimizer to generate a set of hints that freeze a plan. So parse the statements, save as a set of hints and use that from now onwards. That got around the problem of bind variable peeking with "neurotic" results. You always got the same plan, not necessarily the best plan but at least it was the same plan.
Oracle then introduced SQL Profiles. Profiles giving the optimizer additional information at parse time because it can bring in to the parsing algorithm. So, unlike a stored outline, you are all doing dynamic parsing, no question about that, but with a profile, the dynamic parsing is not based purely on the DDL of the objects, the syntax, the statements, the statistics and so on and the dynamically peeked variables. It also brings in the *historical information* of what happened when the profile is built, and what actually were the results of running the statement through different plans improving things a great deal.

A Case Study

The environment was a 4-node RAC OLTP database supporting an e-commerce website. What our client found is they were getting wildly divergent performance. Somedays, even some hours within a day, the searches would run really well. The next day they'd run appallingly. And worse still, because it was a RAC, each instance in a cluster did its own parsing, and that made things even more complicated because each instance was building up its own execution plan. And if you're going to the website and your query happened to be routed through the connection port, one instance it would run well. If it was routed through the connection port another instance, it would run appallingly. Why? Bind variable peeking. What was happening is whenever we restarted an instance or flushed the shared pool, the first query to hit that instance, the bind variables were peeked and the plan was developed based on first values; sometimes it was just OK, sometimes a horrible choice for the queries that followed. The plan was used for hundreds of millions executions. There was just no stability.

So what do we did to fix it? Well, first off is we almost went through the solutions previously:

Dynamic Sampling, Stored Outlines and SQL Profiles.

We began with an easy change, though we suspected it would not completely solve the problem. We increased dynamic sampling. The default setting in 9i was only one, which is far too low (see the OPTIMIZER_DYNAMIC_SAMPLING initialization parameter). At 10g, it equals to two. I would normally recommend pushing it up to four or five, which we did, but as we suspected it not solve the problem.
Then we disabled the bind variable peeking. You can disable it with the undocumented initialization parameter _optimizer_peek_user_binds. (Of course it defaults to true.) If you set that to false, you disable the ability to peek the bind variables at all, and that means you get *stability*. In effect, you're forcing the optimizer to revert to the rule-based optimizer. Instead of doing bind-parse-execute-fetch on the first execution, it does parse-bind-execute-fetch. So it's going to be relying purely on the syntax, the DDL and - to a certain extent statistics - to create the execution plan, but it can't take account of any skews in the data. So having disabled the bind variable peeking, at least we got stability, but what we got was stability with rubbish (i.e. poor) execution plans, resulting in poor response times.
So then we have to do some serious tuning, and we try putting hints in, but tuning with hints - there's not much fun about that. So we reverted to stored outlines. We run the application in various ways, wait until we've got execution plans that didn't seem too bad for most queries and froze them with stored outlines. That gave us stability with plans that were not half bad, but they certainly were not great.
We tried profiles of course, but the only real solution is to upgrade to 11g because in 11g, we have adaptive cursor sharing and that really should be the solution to all these problems.

11g Feedback Based Optimization

So let's take a look at a new concept that 11g architecture embraces: Feedback Based Optimization, the underpinnings of Adaptive Cursor Sharing. I should note that this is not an Oracle only concept. DB2 has the learning optimizer. [Microsoft] SQL Server has some notion of it as well.
Oracle's implementation of Feedback Based Optimization manifests in two features, adaptive cursor sharing and cardinality feedback. While it's different in many ways from the other products on the market, the concept that the database vendors are getting into is let's take some of the run-time data and send it back to the optimizer. If warranted, let's recompile the query. Oracle puts the run-time data in the cursor cache, note it's the cursor cache, so at least for now it is not persistent, and then the SQL engine *feeds back* this knowledge and the Oracle optimizer can make use of it. Feedback Based Optimization is the foundation for the two features:

1. Adaptive cursor sharing to overcome limitations in bind variable peeking
2. Cardinality feedback. To fix cardinality misestimates.

It should be noted that this feature is primarily intended for frequently executed SQL. Since both features can cause SQL to be re-optimized, which we've already determined is expensive, the features are intended for SQL with high execution counts, so that the cost of re-optimization can be amortized over many subsequent executions. Short-duration SQL statements usually fit this model, such as SQL and an online transaction processing application. Long-duration SQL such as decision support queries against data warehouses are typically not executed frequently enough to benefit.

Adaptive Cursor Sharing Concepts

Adaptive cursor sharing provides enhanced optimization in some queries containing bind variables. It really is the next generation of Oracle solution when peeking falls short, leading to instability (unpredictable response times) and poor performance.

What are the goals of ACS as laid out in one of the original papers back in probably 2001, 2002 that was written on this concept?

First of all, there should be no user intervention. The developer or the DBA should not have to do anything in order to feed back information from the SQL engine to the optimizer. If we're talking about re-optimization, we all should be a little nervous about that. We want it to be as low overhead as possible. So it must be fast, use as little memory as possible and - as best as is possible - we must determine that re-parsing must be worth it. That is to say the new plan must perform better than the original plan.

How Adaptive Cursor Sharing Works

When SQL is introduced into the system, it *can be* marked as bind sensitive. When a bind sensitive query is introduced, Oracle stores additional information in the cursor. The additional information is things like selectivity estimates for each predicate that contains a bind variable. A limitation: as repeated executions occur and new values are introduced, you could imagine the enormous amount of possible combinations of values in complex SQL that has many bind variables. I believe that Oracle limits this tracking of selectivity on 12 variables.

Additional selectivity processing code is now included in the soft-parse phase of a query for bind sensitive SQL. So, with the low overhead in mind as one of our goals, only SQL that contains predicates where selectivity can be computed quickly are marked as bind sensitive, and that includes equality predicates - but only equality predicates on columns with histograms. (If there are no histograms, there's no reason to employ ACS, because we would assume that we were going to get the same amount of data back no matter what the value is in the bind variable. If histograms are introduced, we'd assume there's data skew, and we do need this feature. We do need different execution plans for the different values presented in the bind variables.

How does this mix with another new 11g feature, SQL Plan Baselining? If baselining is enabled and Oracle ACS introduces a new execution plan for a statement into the system, but since it hasn't been accepted, hasn't been evolved, it would not be used. So the plan is saved but not yet used. You can set up a nightly job to evolve plans and accept them, so tomorrow the plan becomes an acceptable execution plan. In this scenario we get the stability with different bind variables typically a day later.

However, variations in selectivity alone is not enough to know that re-optimization is worth it. Remember, one of the design goals is to be sure as much as possible that going through re-optimization and putting a new plan in the library cache is worth the time and resources. So Oracle wanted another test.

Bind Aware Queries

The other test is the concept of row source profiling. In row source profiling, Oracle samples query executions and saves these execution statistics - including the number of rows processed by the query. To reduce overhead, the sampling rate is decreased over repeated executions. It can eventually be turned off automatically. If row source profiling reveals large variations in the number of rows processed, some additional information is stored in the cursor, and the query is marked *bind aware*.

A bind aware query is re-optimized if - on a subsequent execution - at cursor matching time (during a soft parse) Oracle realizes that the new bind values are outside the selectivity range stored in the "bind profile".

Then, if the new execution plan is different from the original plan or all the other child plans, a new child cursor is created. You will end up with additional child cursors in this scenario.

ACS and Previous Features

What is the impact of ACS on older features? Stored outlines negate adaptive cursor sharing.
CURSOR_SHARING=SIMILAR is deprecated. It's planned for obsolescence.

Adaptive Cursor Sharing Summary

To wrap up ACS, one thing we should be aware of is to employ ACS a soft parse is required.
Soft parse has a bit of overhead, so the PL/SQL has built-in soft parse avoidance. For example, when you loop through a query result set, it's not re-soft parsing over and over and over again.
So soft parse avoidance is an issue with respect to ACS and, in fact, on Oracle support it's listed as a bug. I guess it could be considered that, but we also musk ask ourselves: Is that what ACS is really designed for? Perhaps not.
We also mentioned cursor_sharing=similar is deprecated and obsolete in R12. There is a BIND_AWARE hint, which I mentioned a little while ago, and that will force the re-optimization on the first execution with a different selectivity profile - because with the hint, we've already put it in BIND_AWARE mode. We've gone from BIND_SENSITIVE to BIND_AWARE and so we've essentially taken out one for the litmus tests, which was a variation or a large variation in the number of rows processed. There's also an opposite hint called NO_ BIND_AWARE if you want to disable this feature.

Cardinality Feedback

The next feature that I'd like to talk about is cardinality feedback. (Cardinality in this context is the number of rows passed back by a step in an execution plan.) This is the second feature that comes from the notion of feedback based optimization. Incorrect cardinality often causes poor response times. That is why Oracle has upgraded the engine and optimizer with respect to this.
Let's just take a moment, however, to review and demonstrate why CBO can *mis-estimate* cardinality. (Later, we will see that these are the causes of "suspicious" queries.)
Complex predicates on a table can cause CBO an issue with respect to estimating the number of rows that the step in the execution plan will return. However, when you're using complex and multiple predicates, consider the 11g extended statistics feature. This can help.
Of course, data skew is another reason for cardinality mis-estimates.
Scalar functions is yet another cause. For example, we often use upper or lower on character searches or TRUNC on dates. Note that 11g extended statistics can help with this issue too.
Old or missing objects statistics can cause mis-estimates.
Another reason for incorrect cardinality estimates are often produced for intermediate sets, as I talked about before.

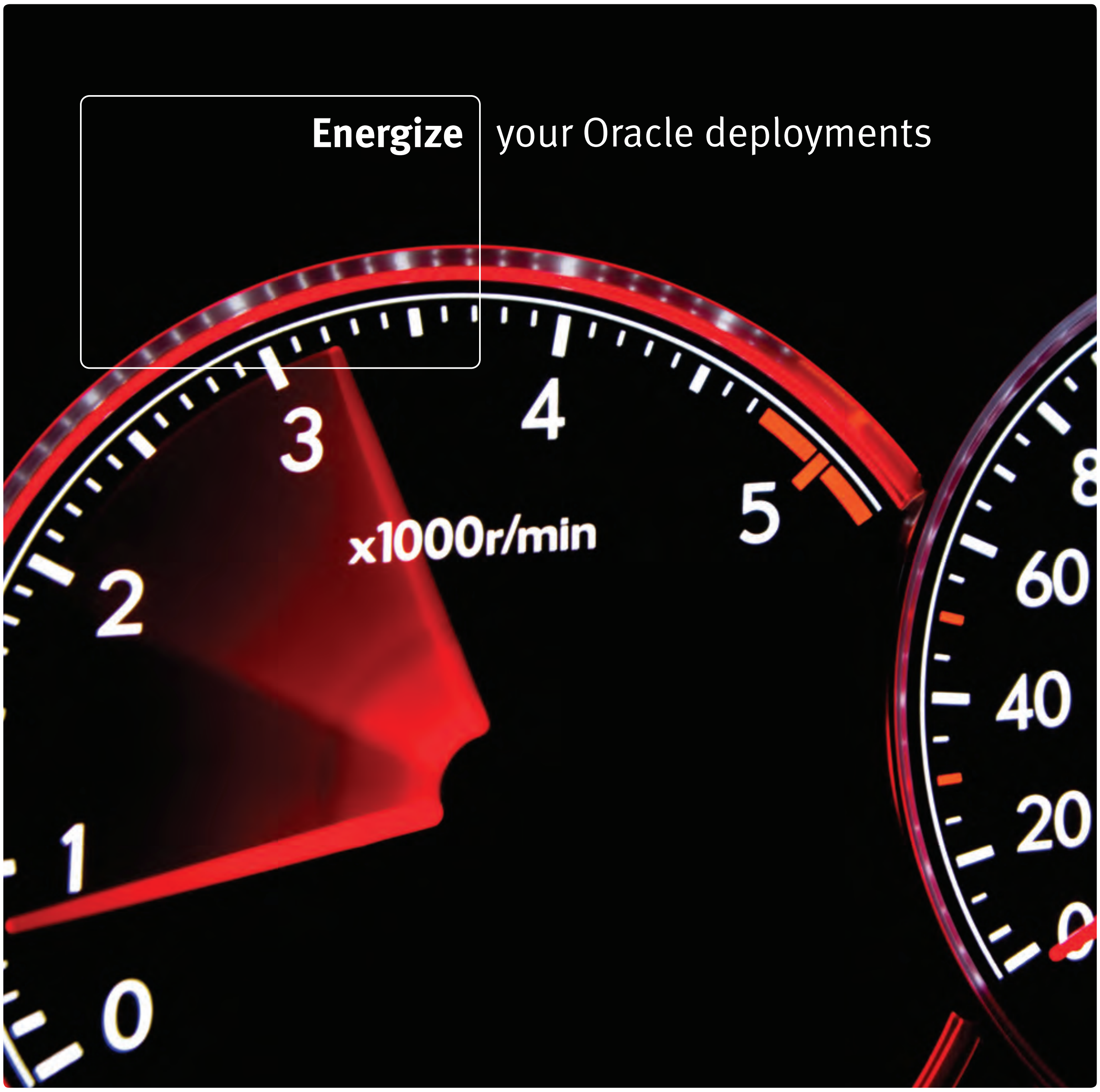
How Cardinality Feedback Works

How does cardinality feedback work? "Suspicious queries" (see above) are monitored. After the first execution, Oracle compares the actual cardinality to the estimated cardinality. If the actual cardinality is significantly different than what it estimated, Oracle injects a hint into the query to supply the correct cardinality and re-parses the query. It should be noted that at this time comparison only happens after the first execution to reduce overhead.

In Closing

Adaptive Cursor Sharing and Cardinality Feedback are two critical performance related features in Oracle 11g. If you tune Oracle and SQL for a living, you will need to learn these features; otherwise you will not understand why your execution plans change dynamically. They are also great upgrade motivators – great reasons to upgrade now.

EMC²
where information lives®



Reduce risk and accelerate your application deployments by drawing on the power of our Oracle® expertise.

With EMC® Proven® Solutions, your information infrastructure accelerates towards greater productivity. Learn more at www.EMC.com/oraclesolutions.

EMC² PROVEN SOLUTIONS

Maximize the ROI of Your Oracle Investment Utilizing APEX

Kashyap Shukla, Mauli Systems, Inc.

Introduction

This paper explains how Oracle Application Express (APEX) can be used to significantly reduce the overall cost of application development, improve time to market, and maximize existing investment in Oracle technologies and skills within your organization. There are some success stories with APEX where organization saved significant cost by developing applications even by non-experienced developers with minimum learning curve. This paper will explain the overall technology architecture of APEX and provide a comparison with other development technologies.

Why Introduce Another Application Development Tool and Why APEX?

This is the first question management raises, when anyone suggests utilizing another application development tool and the organization has already standardized on a certain platform and technology. What are the benefits, implications and concerns? What is the ROI? How is it different than what we have standardized e.g. .Net?

Application Development Technology Trends

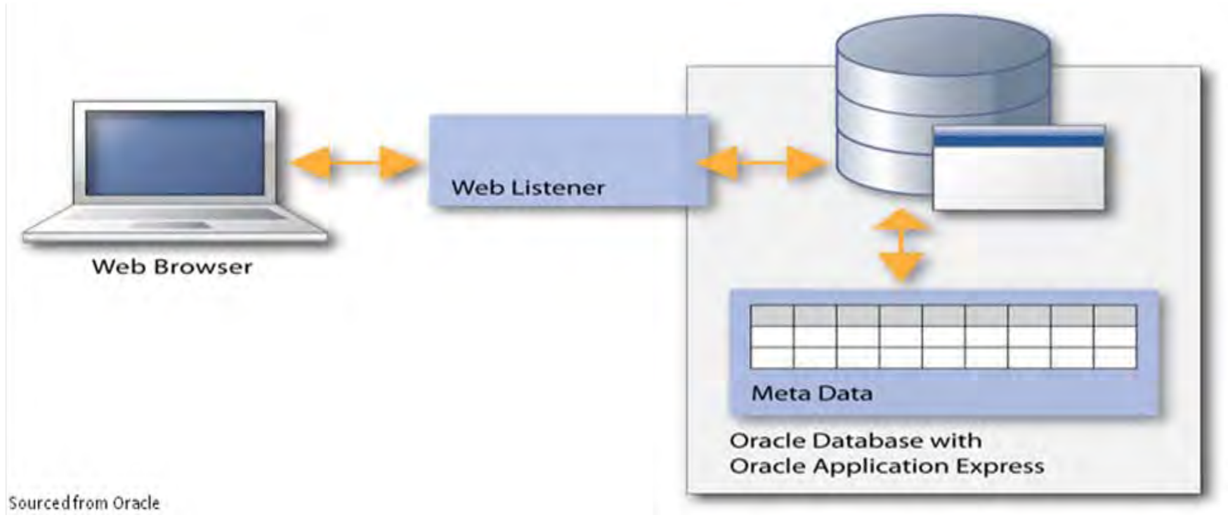
Application Development Technology is continuously evolving. Introduction of new tools are a natural part of the technology lifecycle.

- In mid to late 1980s client-server technology
    - Was introduced and tools such as Oracle Forms, Power Builder, Centura (Gupta SQL), VB, FoxPro, etc. were used to build enterprise applications. Many such applications are still employed.
- In the late 1990s, web technology
    - Was introduced and tools such as Cold Fusion, Dreamweaver, JSP, ASP, etc. were used to build applications.
- In mid-2000s enterprise application development platforms
    - Such as .Net and Java were introduced and many organizations standardized on one or the other as their standard application development platform. These are comprehensive technologies allowing you to build a wide range of applications such as windows, web, and mobile applications; however, these are not Rapid Application Development (RAD) technologies and heavy weight compared to APEX.

Considering above trends it is very likely that we might see another shift in the application development technology, in next ten years or so. Organizations have major concerns about the vendor support for older technologies, frequently upgrade to their latest supported version or migrate applications to the newer technology, which significantly increases overall cost of ownership. While over the past 25 years Oracle greatly enhanced the Database technology providing many new features inside the database including APEX and supported all these different application development technologies. Organizations have more challenges and higher costs in implementing, maintaining and migrating application development technology compared to the database technology. Since APEX resides inside the database it is probable to have the same longevity as the database. Organizations can significantly benefit by reducing cost on application development technology by utilizing lighter application architecture such as APEX and making applications lighter by moving business logic to the database as much as possible.

Contrast APEX vs. .Net-Oracle Solution

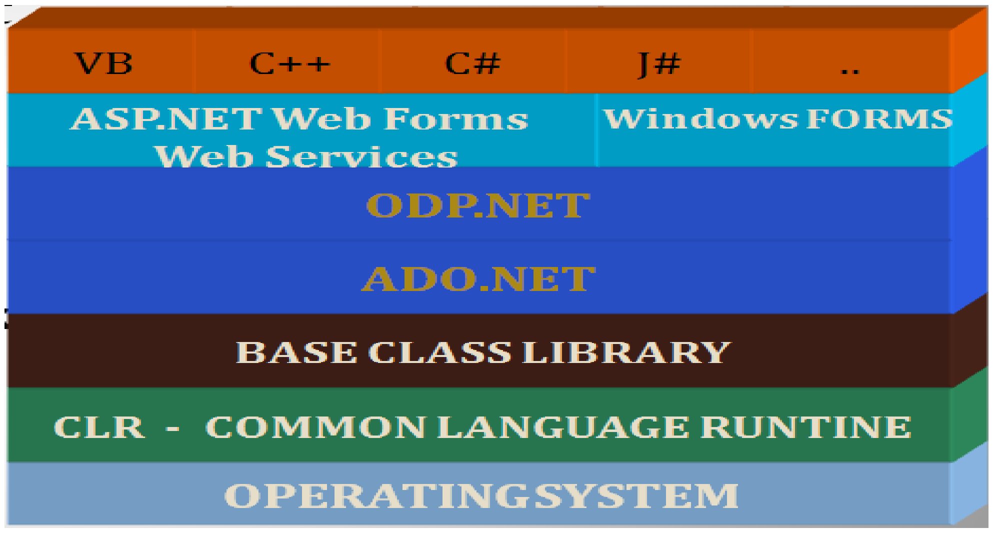
APEX Architecture



APEX is very light weight compared to other platforms.

- Simple 2-Tier Architecture.
- Integrated within the Oracle Database - Runs everywhere where Oracle Database Runs – Fully supported NO-COST Feature.
- Meta data driven - Pages dynamically rendered using database Metadata. No code generation or file based compilation.
- Independent of OS.
- Develop, Deploy and Run within Browser.
- Components Required: Oracle Database and Browser
- Skills required: SQL, PL/SQL, HTML, Java Script

Typical .Net Architecture with Oracle Back-end



.Net is heavy weight compared to APEX.

- 3-Tier/N-Tier Very Comprehensive Architecture.
- Allows you to build windows, web and mobile apps.
- OS Dependent

- Components Required: * Visual Studio, ADO.NET, ODP.NET, Nhibernate / Entity Framework (LINQ), CLR - Framework 2.0 – 4.5, Custom Framework, MVC/MVP Architecture, IIS, AZMAN (Optional), *TFS (Optional). * Separate license required per developer.
- Skills Required: SQL, PLSQL, HTML, Java Script, and additionally need OO (Object-Oriented), One or more languages (C#, VB, C++…), Frameworks and different architectures.
- More layers means more involved in development, deployment, and maintenance.
- There are many options to standardize and maintain.

Required Skill Set Comparison by Technology:

Client-Server Technology e.g. Oracle Forms	Oracle APEX	Microsoft .Net (Web)
SQL & PLSQL	SQL & PLSQL (60%)	SQL & PLSQL
Forms	APEX Framework	Visual Studio, ADO.Net, ODP.Net
HTML*	HTML	HTML
JAVA Script*	JAVA Script	JAVA Script
		OO
		C#, VB, J#, ..
		MVC/MVP Architecture
*applies to JVM environment		Nhibernate / Entity Framework (LINQ)

Client-server developer experienced with any tool e.g. Oracle Forms is skilled with SQL and PLSQL, which are the main languages also used for the back-end. So the same developer is skilled and experienced with both the back-end and front-end developments. Organizations that have invested in Oracle technology also have a corresponding investment in developing or acquiring resources skilled in Oracle database and application development. The skill-set required to develop applications in APEX largely builds on the existing Oracle languages and techniques and is tightly integrated with the Oracle database. There is minimal learning curve; therefore cross training is easier and cost-effective. Thus, introducing this new application development architecture allows an organization to leverage and keep the same resources who have built the business knowledge by utilizing APEX as another tool.
Enterprise Development technologies such as Java and .Net are very comprehensive technologies. These technologies can often require knowledge of multiple technologies and techniques in order to deliver a solution. Finding a single developer with a range of technology skills broad enough to deliver such a solution can be difficult or costly, many times requiring multiple resources to deliver the solution. APEX allows for a single developer to bring a comprehensive end to end solution to market quickly and effectively.
For example, if your web applications are developed with asp.net, but your backend is Oracle, even with a simple application you would need to consider the difficulty in sourcing the skills required to deliver the solution. When hiring a .Net developer you would need to see if they have experience with the methodology that is standardized at the company and with Oracle back-end experience. Most .Net resources have SQL Server background so you need to have Oracle Developer within the team, which may not be cost-effective.

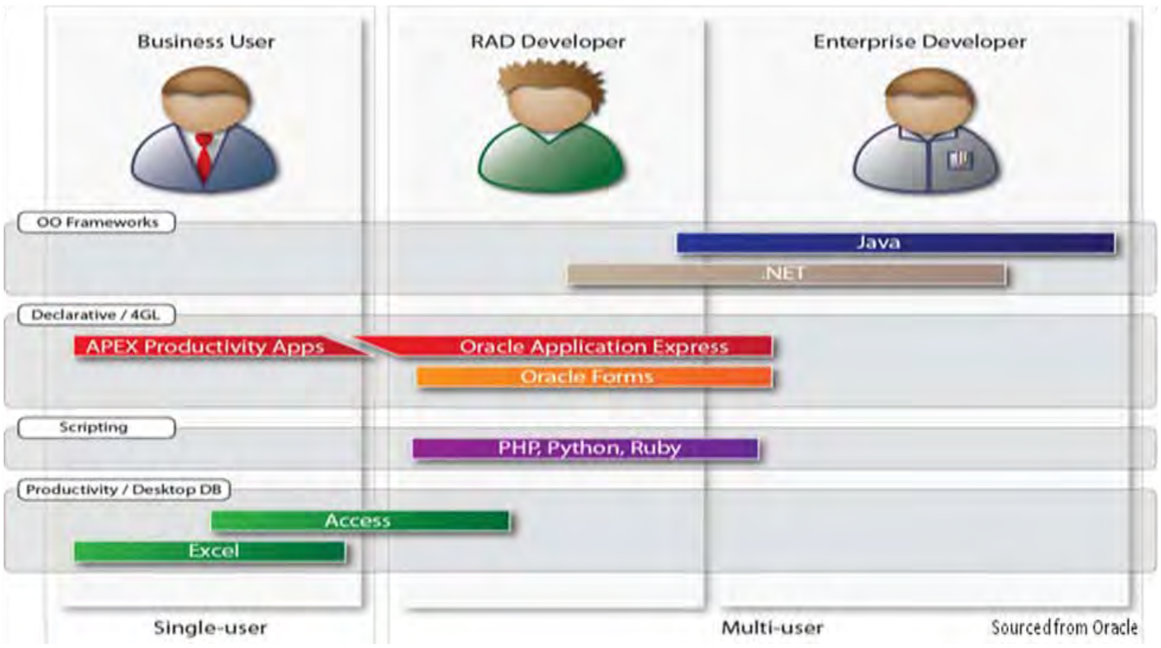
Where Does APEX Fit In?

APEX is targeted at the Rapid Application Development (RAD) developer in the same spectrum (Declarative / 4GL) as Oracle Forms Development Tool. APEX has advantage over scripting languages is the declarative framework and longevity as based on PL/SQL which has been around for 20 years. Also it has advantage over OO Frameworks due to its light weight and RAD.

Many may think that APEX is best suited for only Departmental / small scale applications. However, APEX should be considered in the same spectrum as Oracle Forms and many enterprise legacy systems are built using client-server technology based on Oracle Forms. APEX can be used by professional experienced developers to build enterprise applications and end-users or less experience developers can build smaller scale applications rapidly. APEX can also be used for converting client-server applications including Oracle Forms Applications. If the applications are too large, you could split and build few manageable size modules. It is also recommended to push business logic to back-end as much as possible and make the applications lighter.

Even for a moment if you think APEX is only best suited for smaller scale applications, then ask yourself the following questions which can help you make your decisions.

- What % of new development is for enterprise vs. departmental applications?
- Do you need to enforce developers to use enterprise application development technology to build all the applications?
- Which skill-set resources are available in-house or easily available?



APEX FUD (Fear, Uncertainty, and Doubt)

Decision makers have many concerns and need to have many questions answered before investing into any new technology. Following is the list of typical concerns related to APEX:

- Is APEX Scalable and Secure? Yes
  - APEX resides within the Oracle DB it inherits all the strengths of the DB in relation to scalability, security, reliability, etc.
- Is not the architecture too simple to scale? No
  - Simple Architecture with minimum network overhead and scales as you scale the Database.
  - APEX also works on RAC environment.
- Is APEX only a personal productivity tool similar to Excel or Access? No
  - APEX is capable to handle complex applications with a lot of users and large volume data. Keep data related logic and business rule in the database (“Thick Database")

- Is APEX Extensible? Yes
  - APEX provides Web 2.0 features out-of-the-box, such as Interactive Reports, Flash Charts, etc., but can also be extended with JavaScript, JS Libraries and AJAX to incorporate more Web 2.0 capabilities.
  - Extend APEX framework by adding plug-ins
- Will I need additional costly tools? No
  - Develop, Deploy, Maintain using Internet Browser
- Do we need to pay for additional license? No
  - FREE With the Database License, including support
- Are skilled developers readily available? Yes
  - Growing APEX developers community
  - Require 60-70% of SQL – PLSQL skills
- Will there be a significant learning curve? No
  - There is no significant learning curve for Oracle developers
  - Easier Learning curve compared to .Net or Java
- The notion about something that's free is too good to be true….
  - Try it out….
- Will Oracle charge for APEX in the future?
  - Oracle provides APEX, SQL Developer, etc. at no-cost to encourage development in Oracle Technologies and increase utilization of the Oracle Database
- Is Oracle Committed to the platform?
  - Apex is in its 10th major release since 2004
- Is there any additional cost for maintenance on technology? No
  - It is metadata driven, code resides in DB.
  - No additional cost for Backup, Recovery, DR.
- Do we need additional tool for version control? No
  - Version Control for APEX is not that different from PL/SQL
- How well it integrates with other applications?
  - APEX applications are web based. Can be easily integrated with outside data sources using Web Services and DB links.
  - Integrates with AD and SSO solutions
- Do I need another set of UI Standards? May be
  - You can use any built-in Themes as your standards or one-time investment for customization.
- Is there market place adoption? Please check out at the following sites.
  - Catalog of Commercial Applications
  - Catalog of Internet Applications
  - Catalog of Consulting Companies
  - Oracle APEX Site
  - ODTUG - Oracle Development Tools User Group

APEX Application Development ROI

- Lowers overall cost of
  - Development
  - Deployment
  - Maintenance
  - Upgrades
- Quick Time To Market
  - Business benefits by deploying solutions quickly
  - Free up resources – available for next project
- Maximize existing investments in Oracle

    - APEX is FREE – Including Support
- Leverages existing skilled Oracle developers to contribute more broadly

For the past 25 years, Organizations have invested in Oracle Technology, not just in IT infrastructure, but they have also invested in technical resources with Oracle skills and business knowledge and are still supporting legacy client-server based systems.
ROI scenario is the cost of delivering a solution to meet a business need and whether the cost of delivery is worth it, based on the potential return or work-arounds.
For example: A solution that costs 50k that can be manually performed by a temp resource costing 20k would be considered as poor ROI. The same solution delivered at 10k cost by using RAD tool such as APEX would be considered a very positive ROI.
By utilizing APEX, it is now possible to bring ideas to the market that would have never made it due to cost and resource constraints. Small scale vs. enterprise applications – enterprise applications usually require certain rigors and standards that make "point solution" or small scale solutions "too bulky" (think of the .net framework for example). There needs to be a way for organizations to deploy "light weight" solutions. Organizations which have already invested in Oracle technology should utilize APEX for providing such solutions, which is available FREE including support with Oracle Database license and leverage existing investment and maximize ROI.

Summary

There is lot more to be said about APEX; however these topics cannot be covered in single paper. Message is that you utilize and leverage APEX "the Free Hidden Gem" available with your Oracle Database to significantly reduce overall cost of ownership, maximize ROI and you will also see how it greatly enhances the efficiency and productivity of your Organization.

About the Author

Kashyap Shukla (kashyap_shukla@mauli.com) is the Principal Consultant at Mauli Systems, Inc., providing independent Oracle and Microsoft consulting services in the North-East. He is an Oracle Certified Professional with 30 years of IT experience as a data architect, systems architect, developer, and DBA. He has a successful track record with building system architecture and implementing large scale, complex solutions for numerous global Fortune 500 companies spanning Data Warehousing, BI, custom, and 3rd party System Integration, Software Development, System Conversion, and Upgrade projects. In addition, he is author of commercial software products including the registered Copyright Software "MSI Data Certification System".

SIGS, SIGS and more SIGS!

The following Special Interest Groups (SIG) hold meetings throughout the year for the benefit of NYOUG members:

DBA SIG – Database Administration
Data Warehouse SIG – Business Intelligence
Web SIG – Web / XML / Java / Weblogic / APEX / Fusion
Long Island SIG – Nassau/Suffolk area - All topics

TOP PERFORMANCE

Change Data Capture and Data Integration for Oracle®, MySQL®, IBM® DB2, SQL Server®, Informix®, Sybase® and others.

www.hitsw.com

HiT® SOFTWARE

A BackOffice Associates, LLC Company

DBMoto®

DBMoto® is the best independent data replication and data integration solution available today for real-time Change Data Capture across multiple databases. No consulting required — data replication functions, mapping and verification are all available through prebuilt graphical interfaces!

- Change Data Capture for fast, non-intrusive data updates
- Open APIs to integrate DBMoto into your architecture
- Supports Oracle databases and Oracle® RAC
- Excellent for migrating data between different versions of Oracle
- Multi-server synchronization
- DBMoto Verifier™ for data comparisons before and after data replication
- Easy-to-use — wizards, intuitive graphical interfaces, customizable functions
- Small footprint — extreme performance!

"As a non-DBA expert, I am thrilled to have a software package that was easy to understand and deploy, and that manages my data synchronization for me."
— *TriActive*

"We had a very specific replication requirement to transform the data before it was entered into the MySQL databases. Traditionally this process required manual input to set up and initiate. With DBMoto it's as simple as pressing a button."
— *Gullivers Travel Associates*

"We evaluated virtually every other data replication product on the market and DBMoto came up as the clear winner on price, functionality and usability."
— *EFCO Corporation*

Visit **www.hitsw.com/NYOUG** for more information, including:

- **Free comparison sheet "*Why DBMoto for Oracle?*"**
- **Read case study "*TriActive Supports SaaS Business Model using DBMoto for Data Synchronization*"**
- **Download FREE trial of DBMoto (with full support)**

DATABASES SUPPORTED:
Oracle (v.9 and above) · MySQL · Microsoft SQL Server · IBM DB2 (all versions) · Netezza · Informix · Sybase ASE · SQL Anywhere · Ingres · PostgreSQL

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

ORACLE Gold Partner

MySQL in the Oracle Ecosystem

Philip Antoniades, Director, MySQL Sales Consulting, Oracle

Introduction

MySQL has been around for over 15 years as the world's most popular Open Source database, powering the majority of the Alexa top 20 web sites at any given time. Simple deployment and management, modest hardware requirements, and ubiquity with Open Source tools and applications make MySQL the best choice for a variety of applications. The use cases for MySQL are radically different than those for Oracle 11g, which means that in a diverse database environment there will frequently be a need for both products in different areas.

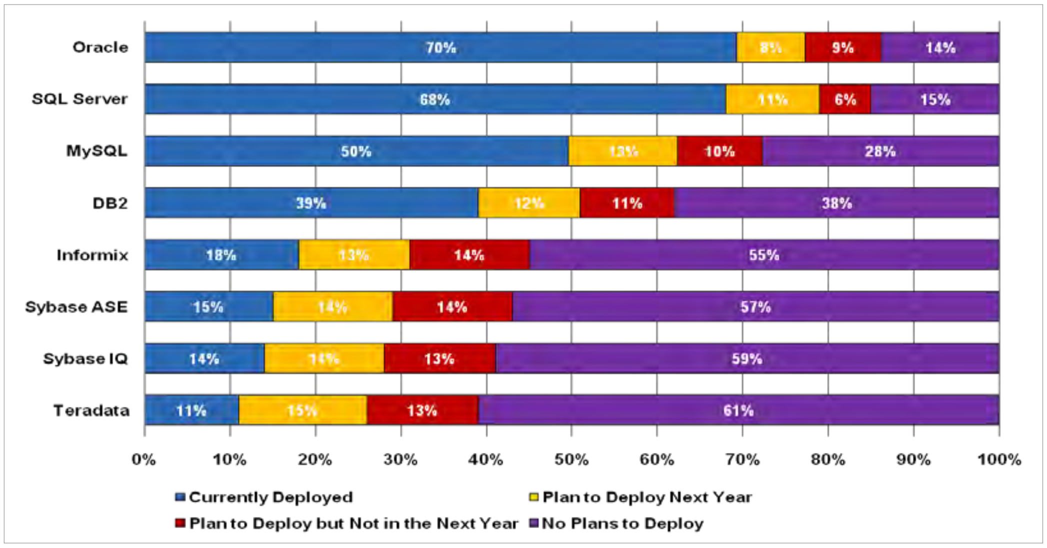
This paper is intended as an primer for Oracle 11g DBAs, Managers, and Developers interested in using the MySQL database in conjunction with their existing Oracle[1] framework. I will review the basic product roadmap and existing and planned integration with other Oracle tools and products, as well as common use cases for heterogeneous Oracle/MySQL environments.

MySQL in the Oracle Corporation

Almost two years ago, Edward Screven laid out the basic direction Oracle would take with MySQL[2], and that strategy is essentially unchanged. The core strategy has been to remain the "#1 Open Source Database for Web Applications", and properties like Facebook, Twitter, Google, Yahoo! and others are strong indicators that it still is. Beyond that the Oracle strategy to "make MySQL a better MySQL" has included numerous improvements to the Windows version and, key to our current purposes, enhancing the experience for "Oracle + MySQL Customers".

The reasoning is simple: Oracle customers are already MySQL users, if not customers. In 2009 a Gartner survey found that MySQL was the 3rd most deployed database in US corporations. By our estimates, 70% of Oracle customers already had MySQL deployed inside their organizations.

The original talk cited three products: Oracle Enterprise manager (OEM), Oracle Secure Backup, and Oracle Audit Vault. As we'll see, those are all moving toward full integration, but in the meantime Oracle has also found other integration points for MySQL



MySQL Inside Oracle Products

The fastest area of adoption has been with the suite of products in Oracle Fusion Middleware (FMW). Currently Oracle certifies MySQL for use with the following products (via JDBC):

- **Database Adapter for Oracle SOA Suite**
- **Oracle Business Process Management**
- **Oracle Enterprise Performance Management**

1 Unless otherwise specified, the word "Oracle" is used here referring to the flagship Oracle Database product, currently version 11g. While an all-MySQL shop is technically "an Oracle shop", that is not the common understanding.

2 http://www.youtube.com/watch?v=C0OkjtlbqVs

- WebLogic Server
- Oracle Virtual Directory
- Oracle Data Integrator
- Oracle Identity Analytics
- Open SSO STS
- Open SSO Fedlet

In version 11.1.1.6 of FMW you will additionally be able to use MySQL as a metadata repository in the following products:

- Database Adapter for Oracle SOA Suite
- Oracle Business Process Management
- Oracle Enterprise Performance Management

Bringing MySQL farther into the Oracle family of products.

Perhaps the most exciting certification for 11g users, however, is a product Oracle had yet to acquire in 2010 – Oracle GoldenGate. From the Oracle Spotlight document, "Organizations can leverage Oracle GoldenGate to move transactional data in real time between MySQL and other systems without impacting the performance of source or target systems."[3]Under Oracle, GoldenGate went from partially supporting MySQL to being fully certified to use MySQL as either a source or a target of replication in the latest version. Meaning wherever your data lives, you can have part or all of it available in either or both products.

MySQL with Oracle Tools

For the Oracle DBA, there are a number of tools that have made it much easier to actively manage a large Oracle environment over the years. Oracle is committed to making those tools available for heterogeneous Oracle/MySQL systems as well, so that DBAs, Developers, and Architects can always choose the right tool for the right job.

Oracle Secure Backup

MySQL Enterprise Users have used a MySQL Enterprise Backup for years to take non-locking full and incremental backups of MySQL systems. Our engineers have extended that so that Oracle Secure Backup can now be configured to back up MySQL databases to image or tape with the encryption, scheduling, and vault rotation they are used to having[4].

Oracle Database Firewall

ODF is a very sophisticated product, combining traditional perimeter firewall abilities (blacklist/whitelist, etc), encryption, and most importantly sophisticated SQL grammar parsing and statement substitution to prevent a variety of database attacks including SQL injection. Recently Oracle announced[5] that it had added full MySQL support to the product for monitoring and reporting.

Oracle Enterprise Manager (OEM)

OEM is a big part of the day for many Oracle DBAs as the premier monitoring and management tool. MySQL DBAs have typically used MySQL Enterprise Monitor to monitor groups of servers and MySQL Workbench to administer inividual ones.

There are currently third-party tools available to monitor MySQL through Grid Control, but engineers at Oracle are working on a plugin to bring full support to MySQL. For homogeneous MySQL environments there will continue to be the MySQL Enterprise Monitor[6]

3 http://www.oracle.com/us/products/middleware/data-integration/mysql-goldengate-spotlight-520648.pdf

4 http://docs.oracle.com/cd/E17952_01/mysql-enterprise-backup-3.7-en/meb-backup-tape.html

5 http://www.oracle.com/us/corporate/press/1451750

6 http://www.mysql.com/products/enterprise/monitor.html

Oracle Audit Vault
Similarly, Oracle Audit Vault is a work in progress. MySQL Enterprise 5.5 introduced and Audit Stream, and work is being done to allow OAV to consume that stream for full auditing of MySQL databases.

MySQL/Oracle Use Cases
Oracle has long been the database of choice for large ERP and other Enterprise software systems, because of the massive scale and redundancy of RAC and Exadata systems and the support and tools mentioned above. MySQL has been the choice for many web shops and startups, as well as embedded applications. Increasingly, those two environments are found under the same roof. With replication products like Oracle GoldenGate and an increasing number of ETL tools, data can move between different databases, making it no longer a choice of only one database for all tasks. Here are some ways our customers have mixed both Oracle database products.

Data Warehousing
Data Warehousing is a broad term for a number of analytic databases of derived or aggregated data sources, and encompasses a number of different use cases in and of itself from multi-terrabyte (or larger) systems to small "datamarts" spun up to solve a particular problem.
Many customers use MySQL for the latter type of system – with a tool like GoldenGate DBAs can divert a subset of real-time data into a MySQL Datamart that can be analyzed with most of the popular OLAP tools.

Look and Book Systems
For many customers the "system of record" is a large single system built to maximize availability and redundancy. When a large vendor like a concert or airline ticketing company moves on to the web, it is imperative that inventory be browsable without overloading the core system.
The largest sites have looked to a "look and book" system, where a group of MySQL servers are populated in real time with read-only data of ticket availability. This server pool can be grown or shrunk to provide fast response times to those looking for tickets, and only pass on actual sales (the "book") to the system of record. This architecture is very flexible for a variety of situations where the number of views of data is exponentially larger than the number of modifications.

Open Source or Packaged Applications
MySQL is the "M" in the ubiquitous "LAMP"[7] software stack. As a result, many popular Open Source tools (such as the Drupal or Wordpress web applications) were written with either principal or sole support for that database. For an Oracle shop wanting to host any of those applications, bringing up a MySQL database can be the path of least resistance.

Conclusion
The key phrase above for DBAs and Developers is "the right tool for the right job". Oracle is working to make it easier for people to focus on the requirements and lifecycle of an application without having to worry about data flow or care and feeding of the product(s) involved.

7 http://en.wikipedia.org/wiki/LAMP_%28software_bundle%29

SQLT – A Free SQL Performance Troubleshooting Utility Provided by the Oracle Center of Expertise (CoE)

Kevin Gilpin, Rolta TUSC

What is SQLT?

SQLT is a free utility created by the Oracle Center of Expertise (CoE) group. It is used to troubleshoot SQL performance. The utility calls the SQL Tuning Advisor (STA) and Trace Analyzer (TRCA) and its own scripts and PL/SQL packages to evaluate SQL statements and create a report containing findings and recommendations as well as a very comprehensive set of configuration and metadata related to a statement or set of statements.

SQL is provided as a zip file freely downloadable from Oracle Support Note 215187.1. It can be used with Oracle 9.2 and higher. The utility makes heavy use of Oracle Tuning Pack features, but if the Tuning Pack is not licensed, these features are not used by SQLT and the utility can still be used in a limited fashion.

Use of SQLT provides a new framework to utilize the rapidly increasing power of Oracle Database's SQL performance management features of SQL Plan Management, SQL Plan Baselines, SQL Tuning Advisor and SQL profiles.

How SQLT Works

SQLT is invoked from SQL*Plus by calling any of a number of provided SQL scripts that execute SQLT in any of several different modes. The various modes can report on performance information about a statement as it is currently running, run the statement and capture performance information, derive performance information about a statement from the AWR, create a performance report based on a trace file, implement a SQL profile with a new execution plan and even implement changes to cost-based optimizer histograms.

The product of executing SQLT is an HTML report of all performance, configuration and metadata related to a particular SQL statement or set of statements called a SQL Tuning Set (STS). Some variations of SQLT produce a separate report along with a script that can implement a new SQL profile containing an improved SQL execution plan or changes to stored histograms.

How to Install It

Download the zip file pertaining to your Oracle RDBMS version from Oracle Support Note 215187.1. Unzip this file (in a directory unofficially referred to as <SQLT_HOME>) and follow the instructions in the sqlt_instructions.html file. In general, the installation consists of navigating to the <SQLT_HOME>/install directory, invoke SQL*Plus and connect to the relevant database as "sys as sysdba", and execute the sqcreate.sql script. There are a handful of prompts to answer:

1. Connect identifier for TNS connections (optional).
2. SQLTXPLAIN database user password.
3. Default and temp tablespaces for SQLTXPLAIN user.
4. Main application user of the SQLT utility. This user will be the one to execute the SQLT scripts. Note that additional users can be added to the configuration later.
5. Management Pack license, if any:
    - "T" if you have license for Diagnostic and Tuning
    - "D" if you have license only for Oracle Diagnostic
    - "N" if you do not have these two licenses

After the last prompt, the remainder of the installation should take a few minutes or less, although on databases with a large number of objects and/or a large number of segment extents allocated, the installation could take a lot longer. The installation creates its own data sub-dictionary in a new database schema that it creates called SQLTXPLAIN. As of SQLT v11.4.4.2 (February, 2012), the SQLTXPLAIN schema consists of the following:

```
OBJECT_TYPE           COUNT(*)
------------------- ----------
PROCEDURE                    1
TYPE                         4
SEQUENCE                    12
PACKAGE                     17
PACKAGE BODY                17
LOB                         91
VIEW                       101
TABLE                      212
INDEX                      227
```

There are also six directory objects created that all reference the database user_dump_dest directory.
The installation will prompt for the main application database user that executes the relevant SQL that is to be evaluated. The SQLT utility should be executed as this application user. After the SQLT installation, additional database users can be added to the installation simply by granting them the SQLT_USER_ROLE role.

How to Use It

The SQLT utility can be used in several ways. These methods are referred to with the following shortcut names: XTRACT, XECUTE, XTRXEC, XPLAIN, COMPARE, TRCANLZR, TRCAXTR, TRCASET, TRCASPLIT, XTRSET, PROFILE, XGRAM, XPLORE. All of these methods are described in detail in the <SQLT_HOME>/sqlt_instructions.html file. Read this file completely before using SQLT. Note that there are also two excellent Powerpoint files included in the <SQLT_HOME/doc/ directory.

The **XTRACT** method is probably the most commonly used method. It performs its analysis on a currently running or recently run SQL statement. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTXTRACT <SQLID>
```

where <SQLID> is the SQL ID of a currently running or recently run SQL statement. To find this SQLID, use Enterprise Manager’s Top Activity page or query the v$sql dynamic performance view.
The sqltxtract.sql script gathers data about the SQL statement or SQLID from the library cache and will execute for generally a minute or two, but in databases that have a large number of database objects or segment extents, the execution could take much longer. The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.

The **XECUTE** method takes as its input a text file containing a SQL statement or STS. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQTXECUTE <FULL_PATHNAME_OF_SQL_SCRIPT>
```

The sqltxecute.sql script executes the SQL statement or statements in the STS. Therefore, the length of time that the sqltxecute.sql script takes is bound by what those SQL statements take to execute. The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.

The **XTREC** method is a hybrid of the XTRACT and XTREC methods. It performs its analysis on a currently running or recently run SQL statement and then produces a script that is used to execute the given statement. The literal values substituted for bind variables in the statement are those derived from bind peeking at the time of executing the statement with the most expensive plans derived by the optimizer for the statement. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTXTREC <SQLID>
```

where <SQLID> is the SQL ID of a currently running or recently run SQL statement. To find this SQLID, use Enterprise Manager's Top Activity page or query the v$sql dynamic performance view.

The **XPLAIN** method takes as its input a text file containing a SQL statement or STS. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQTXECUTE <FULL_PATHNAME_OF_SQL_SCRIPT>
```

The sqltxplain.sql script executes explain plan on the the SQL statement or statements in the STS. The sqltxplain.sql script does not execute the statement. It is recommended that the XTRACT, XECUTE or XTREC methods be used before considering using the XPLAIN method because they are more accurate. They are more accurate because they can utilize bind peeking (if it is configured), which explain plan cannot use. The XPLAIN script will be least accurate if the SQL statement has bind variables. If it does not, then it should be fairly accurate. The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.
The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.

The **COMPARE** method uses the data in the SQLTXPLAIN database schema from two different databases in which a given SQL statement or STS has inexplicably produced different execution plans and the goal is to find out why. The sqltcompare.sql script takes two SQLID values as input and assumes that the SQLTXPLAIN schema data has been exported from one of the databases and imported into the other. Alternatively, a third database could be used into which the SQLTXPLAIN data from both databases has been imported. The output file is similar to the other SQLT methods and should contain the reason(s) for why the execution plans used by each database are different. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTCOMPARE <SQLID1> <SQLID2>
```

The **TRCANLZR** method takes as its input the full pathname a SQL trace file that has been created by a 10046 event trace of a SQL statement or STS. The output is a zip file containing an HTML report describing pertinent configuration and performance data related to the SQL. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTRCANLZR <FULL_PATHNAME_OF_TRACE_FILE>
```

The **TRCAXTR** method extends the functionality of the TRCANLZR method by first executing the TRCANLZR method on the given SQL trace file and then calling the SQLTXTRACT method on the top SQL that is found in the file. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTRCAXTR <FULL_PATHNAME_OF_TRACE_FILE>
```

The **TRCASET** method is automatically executed by the TRCAXTR method. In standalone execution mode, this method allows the user to select particular SQLID's upon which prior analyses have been performed. The user picks one prior analysis to have the XTRACT method performed on. This method can be executed in standalone mode by navigating to the <SQLT_HOME>/run/ directory invoking SQL*Plus, and connecting to the main application user that executes the SQL in question, and executing this command:

```
SQL> SQLTRCASET <APPLICATION_USER>
```

The **TRCASPLIT** method takes a SQL trace file that was produced with events 10046 and 10053 and splits the trace file into two separate trace files – one for the 10046 data and one for the 10053 data. The two separate trace files produced could then be separately used with the TRCANLZR method, if desired. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTRCASPLIT <FULL_PATHNAME_OF_TRACE_FILE>
```

The **XTRSET** method is an extension of the XTRACT method in which a set of SQLID's are passed to the script and the script derives the data about these SQLID's from the library cache and/or the AWR and combines the output of all of them into one report. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTXTRSET  <SQLID1> <SQLID2> … <SQLIDN>
```

The **PROFILE** requires that you first execute the XTRACT or XECUTE method on a particular SQLID. The script accepts a SQLID as input. When the sqltprofile script is executed, it will prompt for a plan hash value from a list of plan hash values captured from previous executions of XTRACT or XECUTE on the indicated SQLID. The user can choose a particular one of these plan hash values and the sqltprofile.sql script will output a script to install that plan hash value as the one to be used for that SQLID by use of a SQL profile. To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

```
SQL> SQLTPROFILE  <SQLID>
```

The **XGRAM** method involves the usage of several scripts in the <SQLT_HOME>/utl/xtram/ directory. These scripts are used to make modifications to stored optimizer histograms related to particular SQL. Before using these, read all of the scripts in the <SQLT_HOME>/utl/xgram/ directory to determine which is appropriate to execute in a given case.

The conditions to consider using the **XPLORE** method are rare. The scripts to use this method are in the <SQLT_HOME/utl/xplore/ directory. Read the readme.txt file in that directory before doing anything else with this method. This method is appropriate to use in rare cases in which the optimizer is producing invalid results following an upgrade. The scripts used by this method will "explore" variations in the optimizer_features_enabled and fix parameters that may have led to the problematic behavior.

Usage Recommendations

- Remove old, orphaned SQL profiles.
- Delete old zip files produced by SQLT executions.
- Space consumption by the SQLTXPLAIN database schema.
- To add additional database users to the configuration so that you can run SQLT as those users, grant the role SQLT_USER_ROLE to those users.
- Upgrading SQLT – instructions in the sqlt_instructions.html file (in zip distribution bundle).

- If you wish to "start over" and just purge the entire SQLT schema and re-install it, connect as "sys as sysdba" and execute the <SQLT_HOME>/install/sqdrop.sql.
- Be patient when installing and running the utility on databases that have a *lot* of objects (Oracle EBS, BAAN, etc.). The utility executes queries on dictionary tables like all_objects, etc.
- You will get the best results from SQLTXTRACT and SQLTXECUTE if you execute these scripts as soon after the SQL of interest has started, if not right at the moment it has started execution.
- Optimizer statistics on the SQLTXPLAIN schema are locked after the SQLT installation. To gather stats on them, you must first unlock them with dbms_stats.unlock_schema_stats(ownname=>'SQLTXPLAIN');
- If you have a database with a *lot* of extents (a lot of rows in sys.fet$), the installation may take a long time. Be patient with the install because subsequent executions of the SQLT scripts should then not suffer some of the performance issues caused by bugs in some database versions, like bugs 2948717, 5029334, 5259025. Some MOS notes like 422730.1 suggest gathering stats on sys.x$ktfbue. Otherwise, upgrading/patching may be required to resolve these.
- Due to bugs like this, it might be helpful, for example, to create and schedule a batch script that drops and re-installs SQLT periodically so that you have the freshest data in tables like SQLTXPLAIN.TRCA$_EXTENTS.
- It is highly prudent to run frequently used and high importance SQL through SQLT on a regular basis and review the findings and recommendations.
- Consider setting the *optimizer_capture_sql_plan_baselines* parameter to TRUE (default=FALSE).
- Review of some SQLT reports (files provided separately).

References

My Oracle Support Notes

- 215187.1 – SQLT main note.
- 224270.1 – TRCANLZR – Tool for interpreting raw SQL traces.
- 1229904.1 – Real time SQL monitoring in 11g.
- 1366133.1 – SQL tuning health check script
- 1401111.1 – Announcement of SQLT webcast on May 15, 2012.
- 199083.1 – SQL query performance overview.
- 376442.1 – How to collect 10046 trace diagnostics for performance issues.
- 398838.1 – FAQ: SQL query performance.
- 276103.1 – Performance tuning using Advisors and manageability features

Related References

- 748642.1 – How to create AWR snapshots and baselines.
- 190124.1 – The CoE Performance Method
- 390374.1 – Oracle Performance Diagnostic Guide
- 461053.1 – OS Watcher Black Box Analyzer
- 122669.1 – How to perform a health check on an Oracle database
- Chapter 15 (SQL Plan Management) of Performance Tuning Guide
- Chapter 17 (Automatic SQL Tuning) of Performance Tuning Guide
- sqlt_instructions.html file in the SQLT zip bundle.
- Files in the <SQLT_HOME>/doc/ directory, including two excellent Powerpoint files.

PL/SQL Packages

- DBMS_SQLTUNE
- DBMS_AUTO_SQLTUNE
- DBA_SQL_PROFILES

- DBMS_SPM

Dictionary Views

- DBA_ADVISOR_TASKS
- DBA_ADVISOR_EXECUTIONS
- DBA_ADVISOR_FINDINGS
- DBA_ADVISOR_RECOMMENDATIONS
- DBA_ADVISOR_RATIONALE
- DBA_SQLTUNE_STATISTICS
- DBA_SQLTUNE_BINDS
- DBA_SQLTUNE_PLANS
- DBA_SQLSET
- DBA_SQLSET_BINDS
- DBA_SQLSET_STATEMENTS
- DBA_SQLSET_REFERENCES
- V$ADVISOR_PROGRESS
- V$SQL
- V$SQLAREA
- V$SQLSTATS
- V$SQL_BINDS

We can't begin to tell you what winning Oracle's highest honors means.

(Thankfully, our clients can.)

"...expertise, thoroughness and dedication to getting things right."
"...an excellent blend of systems design, support and in-depth knowledge..."
"[They are] our trusted database services partner."
"...flexible and very knowledgeable..."
"...a source of expert data-base services and support..."
"...a key partner in helping us architect and deliver a robust solution to our cli-ents."
"Cintra exceeded my expectations..."
"Cintra makes the whole process seamless..."
"[Cintra] helps me to sleep better at night."
"Cintra's service has been impeccable..."

Oracle's 2011 Specialized Global Partner of the Year for Database

2011 Oracle Titan Award for Database and Clustering

Experts in the rapid deployment of Oracle's Engineered Systems, including Oracle Database Appliance and Exadata

To learn more about our commitment to specialization and what it means to your business, visit cintra.com.

ORACLE Platinum Partner

Specialized
Oracle Database
Oracle Database Performance Tuning
Oracle GoldenGate
Oracle Real Application Clusters
Oracle Business Intelligence

CINTRA

US | 1-866-245-0881
UK | +44 (0) 845 121 3242

ADF Patterns for Forms Modernizations

Robert Nocera, NEOS and Vgo Software

Introduction

Through our modernization practice we have been exposed to a variety of different Oracle Forms applications, many of which have been around for decades. Many of them have been upgraded to each latest release of Oracle Formsand yet the applications are still starting to show signs of age. A number of these applications were written when the people working on them were still renting movies to watch on VCRs and though most people have by now upgraded to DVD and Blu-Ray players, companies have not been so eager to upgrade their existing Forms applications. The motto is usually "if it ain't broke, don't fix it," and indeed, in a lot of cases, the sentiment makes business sense.

Many of these applications perform core business functionality and echo a business process that was in place decades ago with tweaks and changes to that process that have caused changes to the application. The applicationscan contain a lot of business logic implementing rules that even the users are no longer familiar with. It is no wonder that a business would rather do as little as possible to disrupt that application and process than modernize the application.

However, this philosophy can only succeed for so long. Eventually the platform that the application runs on expires, the hardware is no longer available and the system must be upgraded. This upgrade can take many forms, sometimes just performing the bare minimum to get the application supported again. Increasingly, however, the need to upgrade these applications is seen as an opportunity to move the application to a platform that is considered more "modern".

With the advent of the web and the prevalence of web applications within businesses, users have become accustomed to certain UI standards. This familiarity with web applications makes it easier to gain user acceptance for changes in the UI design of the application that may be necessary because of the new framework, it also sometimes is the impetus to modernize an existing application because users are expecting more and more. Of course, the UI changes to the application are probably just the tip of the iceberg. The entire technological foundation of the application would be changing from Oracle Forms to a modern java-oriented ADF 11g framework.

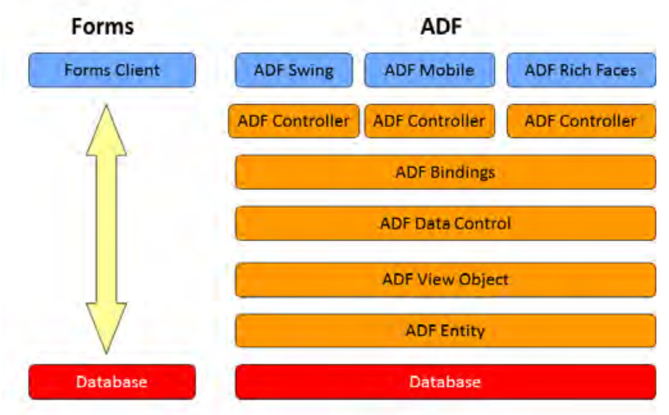


Figure 1. Forms vs. ADF High Level View

This paper will discuss some of those changes and introduce some patterns and best practices that can be used when modernizing an application from Oracle Forms to ADF 11g. Some of those patterns will be UI changes that attempt to limit the changes to the business process while making changes to the screens that make sense to the new framework. Others of those patterns and best practices will be hidden from the user completely and really just a better way to implement the same functionality in the new framework.

Basic Functionality

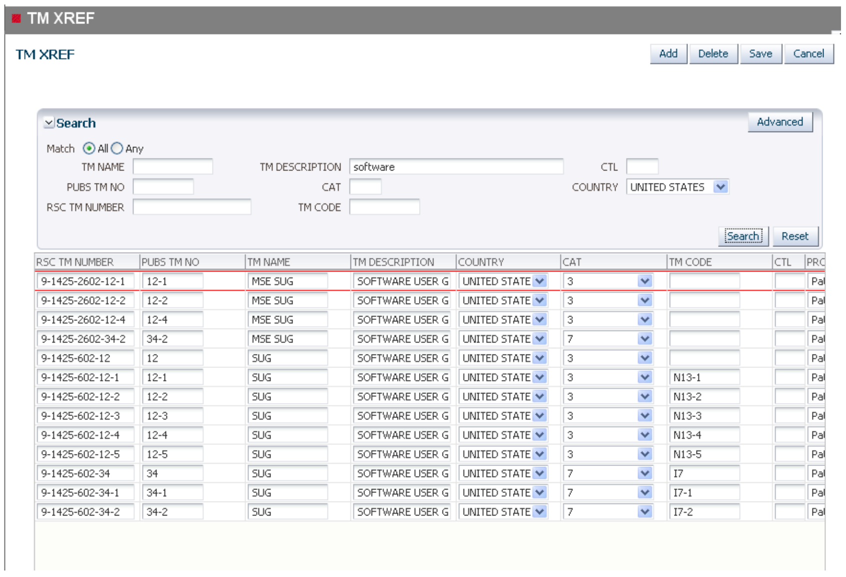
Oracle Forms have a lot of functionality built-in. The framework allows developers to quickly build applications. The design of the framework is also conducive to some patterns that are not as intuitive as they could be for the users of the applications built with it.

In a lot of cases, the same screen is used in an Oracle Forms application to perform a search and edit data. The types of layouts on these screens range from simple table-based layouts to more complex master-detail layouts. The user will normally select "Enter Query" from the menu, enter some searchcriteria, then select "Execute Query" from the menu to get the results of the query. Those results can then be edited from that same screen. A user can also add a new record and use the same screen to populate the data. This type of functionality can be divided into two categories; table-based, and form-based.

Table-based Search/Edit

A table-based search and edit pattern is a screen which contains a single table. The user can search on data by entering search criteria in the first row and searching on that. The results are displayed in the same table and can be edited by the user.

For this simple category of screen, typically used to maintain administration data such as lists of countries or product types, the UI pattern typically used in an ADF application is very similar to the pattern used in the Form application. The main difference is that there is a separate component used to perform the search.



Match [✓ All] Any

TM NAME		TM DESCRIPTION	software	CTL	
PUBS TM NO		CAT		COUNTRY	UNITED STATES
RSC TM NUMBER		TM CODE			

RSC TM NUMBER	PUBS TM NO	TM NAME	TM DESCRIPTION	COUNTRY	CAT	TM CODE	CTL	PRC
9-1425-2602-12-1	12-1	MSE SUG	SOFTWARE USER G	UNITED STATE	3			Pal
9-1425-2602-12-2	12-2	MSE SUG	SOFTWARE USER G	UNITED STATE	3			Pal
9-1425-2602-12-4	12-4	MSE SUG	SOFTWARE USER G	UNITED STATE	3			Pal
9-1425-2602-34-2	34-2	MSE SUG	SOFTWARE USER G	UNITED STATE	7			Pal
9-1425-602-12	12	SUG	SOFTWARE USER G	UNITED STATE	3			Pal
9-1425-602-12-1	12-1	SUG	SOFTWARE USER G	UNITED STATE	3	N13-1		Pal
9-1425-602-12-2	12-2	SUG	SOFTWARE USER G	UNITED STATE	3	N13-2		Pal
9-1425-602-12-3	12-3	SUG	SOFTWARE USER G	UNITED STATE	3	N13-3		Pal
9-1425-602-12-4	12-4	SUG	SOFTWARE USER G	UNITED STATE	3	N13-4		Pal
9-1425-602-12-5	12-5	SUG	SOFTWARE USER G	UNITED STATE	3	N13-5		Pal
9-1425-602-34	34	SUG	SOFTWARE USER G	UNITED STATE	7	I7		Pal
9-1425-602-34-1	34-1	SUG	SOFTWARE USER G	UNITED STATE	7	I7-1		Pal
9-1425-602-34-2	34-2	SUG	SOFTWARE USER G	UNITED STATE	7	I7-2		Pal

Figure 2. ADF Table-based Search and Edit

If you are familiar with the ADF 11g framework you know that the simplest way to provide the type of CRUD capabilities in these types of forms is to create an Entity based on the table being edited. A View Object would then be created for that Entity and exposed to the user in the page. To provide the search capability to the user, the developer would create a set of View Criteria. This View Criteria is then used to create an ADF Query Panel that the user of the application will use to search the data.

That the ADF framework provides such a component is a very good thing, it can reduce the complexity of what the developer needs to do greatly. Of course, great power comes with great restrictions. The down-side is that developers do not have complete control over how the query component is rendered. Developers do have, indirectly, the ability to influence the rendering of the query component. By creating a View Criteria, ordering the source attributes in the View Objects, and modifying some of the attributes on the component itself,the query panel's display will be modified.

In order to take advantage of the benefits of the ADF Query Panel, it must be included on the page and this in itself is a difference from a typical Form application that in some respects influences the Search and Edit patterns that we use.

The simple table-based pattern keeps the UI very similar to the Form screen except for the addition of the ADF Query Panel control and buttons for adding and deleting records. One common pattern that we use is to remove the header menu that provides a lot of the functionality in a Forms application and use buttons instead.
In the simple layout, the buttons used include Add, Delete, Save and Cancel. In this case, the page is popped up from a menu, in cases where it is not, a back or return button may be necessary to allow the user to navigate away from the page. In order to alleviate any unnecessary button clicks, the ADF Panel Query is provided fully disclosed. The user can then search and see the results on the same screen. Edits to the resulting data can be performed in the table itself and records can be deleted from the returned results. When the user wants to add a record, they simply click the add button which will insert an empty row into the table and then the user can enter data directly into that row.

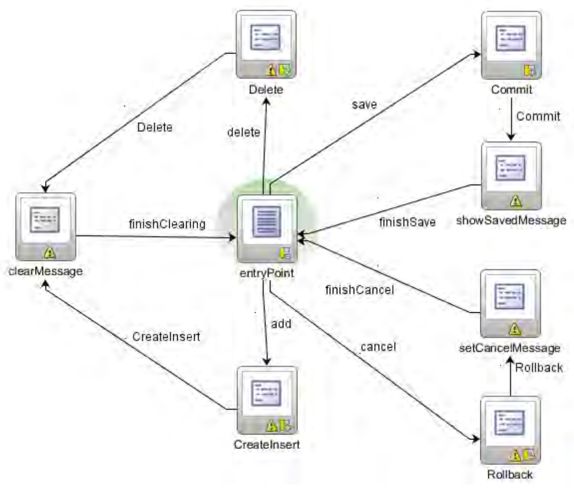


Figure 3. Task Flow for Table-based Search and Edit

The Task Flow for this type of search and edit functionality explicitly shows the messages being set and cleared to indicate to the user that an action has been successfully performed.
Users of the original application will normally react well to a change such as this. Though the UI has changed from the original form, it provides them some additional functionality that they did not have before via the ADF Query Panel. If the Oracle Metadata Repository was also used, the users could even save their favorite queries using this panel and have them available whenever they return to the screen.
The resulting screen is also fairly intuitive. It is fairly obvious to the user how to perform the functions that they need to perform on this screen and so even though it is different, it is easy for them to use.

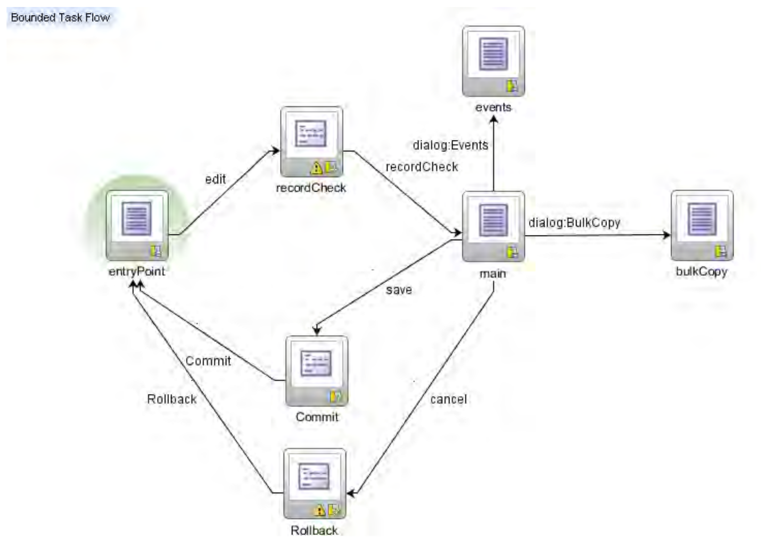


Figure 4. Example Task Flow for Separate Search and Edit

Another example of a search and edit form is the more complex search and edit screen. This is a screen where the record is displayed to the user, not in a table, but in a Form layout that allows the components to take up more real-estate on the screen. In a lot of these cases, Master-Detail blocks are displayed in the Form that allow the user to not only add, edit and delete master records but add, edit and delete detail records as well.

In these situations a pattern similar to the simple table-based search and edit is followed except that the search and editfunctionality is split up into two separate screens. It would be possible to provide an ADF Query Panel at the top of the page and show the results one at a time similar to the original form, but providing a separate search page with a table for results allows the users to search and select the exact row they want to edit without having to scroll one by one through the list.

The edit page will normally contain just Save and Cancel buttons but if users want to be able to scroll through and edit the entire result set without navigating back to the list page, it is easy enough to add Previous and Next buttons as well similar to the existing Forms screen.

Tabs vs. Scrollbar

One problem that Forms developers faced was the amount of screen real-estate allowed for their application. If a Form screen had many fields and many detail tables to show, there may not have been enough screen space to show it all. A solution that is very common among many applications of that era was to provide a set of tabs to separate the data. Often different detail tables would be on different tabs.

Today, in modern web applications, users are accustomed to scrolling up and down a page to see an entire set of information. Many of these older Forms screens with lots of tabs can be handled by removing the tabs and listing all of the information on one page. This allows the user to scroll up and down and see all they want to see.

This is not to say that there is no place for tabs at all. Tabs can be very useful in partitioning the information shown to a user. In fact in at least one example we added tabs to a page that did not have them originally in order to provide more screen space for the detailed data.

LOVs vs. Dropdowns

Forms applications use a lot of Lists of Values (LOVs). Newer applications have a mix of LOVs and dropdowns. A lot of times, when an application used a list of values, after the user selected a value from the list, a description field would be populated.

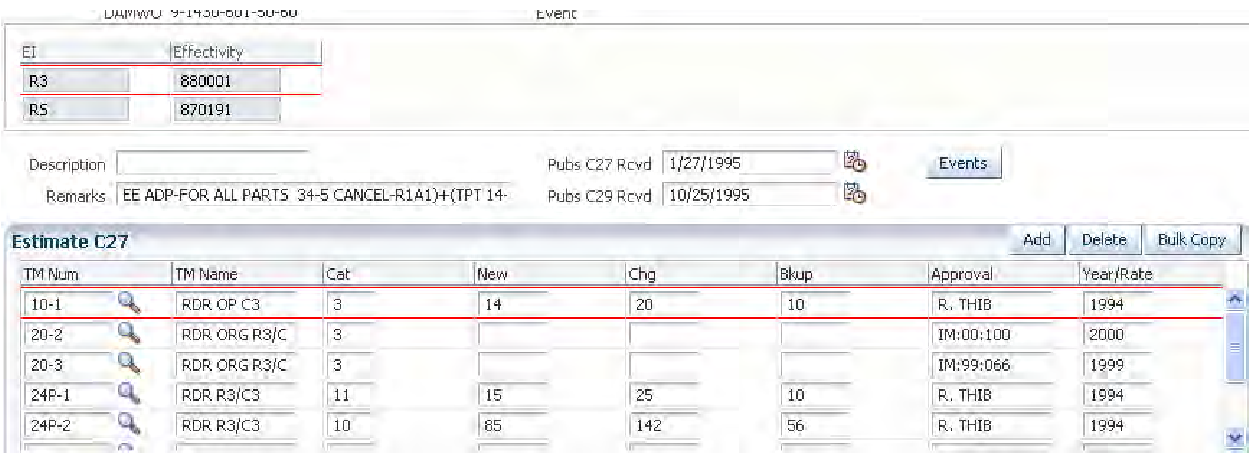
ADF 11g provides an easy way to produce this same behavior but you can also just as easily display the description instead of the key when a value is selected or show the key and the description when a value is selected. The ADF 11g LOV components also provide some more enhanced search functionality.
For short lists of values, consider replacing with single choice selects, for longer lists use popup LOVs to postpone the initial query that retrieves the list until it is actually needed.

Post-Query vs. View

Post Query Triggers in Oracle Forms applications can be used for many things and depending on what function the trigger was performing in the Form, implementing the same functionality in ADF 11g can be quite similar or very different. Oracle Forms allows you to make an awful lot of database calls in a very short period of time having hardly any impact at all on the performance of the application. This makes, for example, populating a description field via Post-Query Trigger an acceptable option. This option is not as acceptable for applications built in other frameworks, including ADF. Instead, in ADF, when you create a LOV and add a transient description field from another Entity, as any number of blog posts and development guides will tell you to do, ADF will create a join in the query for the View that contains the description field so that the field will be populated at the same time the rest of the data is retrieved from the database. This eliminates the need for a separate call and even a separate call per row of results returned in some cases.
This is a very common and simple example but at the same time it illuminates a point that should be recognized. The performance of your application is going to be greatly affected by how many database calls are made. Watch out for situations where you might be executing a query per row of a result set and try to reduce that access as much as possible.

Keystrokes and Menu Items vs. Buttons

In many older Forms applications the users are used to a lot of keystroke commands to execute certain functions. You can mimic some of this behavior with shortcuts on buttons but no matter what you do, some of the functionality is almost certainly going to be replaced by buttons.
Forms is also very good at keeping track of context, knowing where the cursor is in the UI and executing the right event depending on the field that has focus. For example, if there is a master record and several detail tables, clicking add will add a record to the right record depending upon what block had focus when the button was pressed.



EI	Effectivity
R3	880001
R5	870191

Description:
Remarks: EE ADP-FOR ALL PARTS 34-5 CANCEL-R1A1)+(TPT 14-
Pubs C27 Rcvd: 1/27/1995
Pubs C29 Rcvd: 10/25/1995

Estimate C27

TM Num	TM Name	Cat	New	Chg	Bkup	Approval	Year/Rate
10-1	RDR OP C3	3	14	20	10	R. THIB	1994
20-2	RDR ORG R3/C	3				IM:00:100	2000
20-3	RDR ORG R3/C	3				IM:99:066	1999
24P-1	RDR R3/C3	11	15	25	10	R. THIB	1994
24P-2	RDR R3/C3	10	85	142	56	R. THIB	1994

Figure 5. Use of Buttons on Detail Table

While this can be achieved in ADF 11g, it makes the application more complex, most of the time unnecessarily complex, and it also has the effect of not being intuitive for new users. Instead, put your detail tables inside a panel header group and add a toolbar with add/delete buttons to the header's toolbar facet. This puts more buttons on the page, but it makes their use very intuitive.

More Advanced Post-Query Triggers

More complex post-query triggers will require some forethought as to where that functionality belongs in an ADF application as there are many choices.

For complex functionality that is affecting multiple views it is best to create a method in the ApplicationModule's implementation class and expose it so that it can be run declaratively in the Task Flow. As mentioned before, other Post-Query trigger events may be better implemented by creating a more complex query in the View Object itself that takes into account the logic in the Post-Query trigger.

Bounded Task Flow per "Form"

Existing Forms lend themselves well to being mapped to Bounded Task Flows in ADF. Use the "Use Existing Transaction If Possible" selection so that if you nest your task flows the transaction will be handled by the parent.

Each Form will normally encapsulate a particular process in a system. If the original application was designed correctly the mapping will be very close to 1 Form to 1 Task Flow. If the original Forms application is too complex and contains too many screens it would make sense to break it into multiple Task Flows that would then be nested inside a parent Task Flow.

Summary

Any project that is undertaken to modernize a legacy Oracle Forms application and implement it in ADF should consider that some aspects of the usability of the application will change. Most of these changes will result in an application that has a modern look and feel and the user base should be accepting of those changes.

Though the underlying architecture of the application will change completely, and though ADF is a very different platform than Oracle Forms, it is still possible to map functionality of an Oracle Forms application into functionality in an ADF application. A large part of these mappings are straightforward, such as mapping Oracle Forms' Blocks to ADF's View Objects. There will, however, be functionality that must be implemented in an ADF application that will not map directly from a Forms application and that functionality will be where the team working on the project will spend most of its time.

Taking time upfront to map out how an application will move from Forms to ADF and informing the user base of any potential impacts to the work they perform will greatly reduce the time spent developing the application in ADF and getting it rolled out to the user base.

About the Author

Robert Nocera is the CTO and co-founder of NEOS and Vgo Software. He is tasked with driving technology direction for the company's modernization solutions but has also developed a focused understanding of Oracle Fusion Middleware and ADF v11 in particular. Robert has spoken at ODTUG (2008), UKOUG (2010) and a number of regional OUGs (NYOUG, NEOUG, and more). He also has spoken on ADF and ADF security at the 2009 OOW "unconference." Robert is deeply skilled in all Java related technologies/languages. He has over twenty years of IT experience and is the author of the www.java-hair.com blog.

About NEOS and Vgo Software

NEOS is a management consulting and technology services company, offering business system modernization and enterprise data services capabilities. NEOS partners with its clients to best position their business by leveraging our collective experience, capabilities and proprietary products, across all industries and business functions.

NEOS was founded in Connecticut in 2000 and has expanded its business throughout the US, into Europe, the Middle East and Japan. NEOS clients range from large, mid-cap companies to the Global 1000 with specific industry expertise in financial services and insurance. Vgo Software, founded in 2005, was nominated as one of Connecticut's "Fast Track" companies and has enjoyed positive growth since its beginnings.

Vgo Software provides modernization solutions and professional services that support legacy application strategies, assessments, conversions, and development. Vgo utilizes proven proprietary methodologies and software products

including Evo, (Oracle Forms® to Java); ART™, (Application Portfolio Assessment for Oracle Forms® and PowerBuilder Applications); and Evolutions™ (Application Conversion Methodology). Vgo is one of only two certified third-party solutions conducting Oracle Forms® to Java modernization.
Vgo Software was created based on intellectual property developed through many professional services engagements conducted by its parent company, **NEOS LLC**. Our products and offerings are based on real-world experience from hands-on projects delivered to clients.
Vgo Software's product lines will continue to expand in functionality as well as grow in scale. It is our intent to continue to provide superior, thoughtful and innovative automation solutions to customers world-wide.

Contact

Please contact **NEOS and Vgo Software at +1-860-519-5601** for more information or visit us at www.neosllc.com and www.vgosoftware.com.

Exadata Demystified

Arup Nanda, Starwood Hotels

If you are an Oracle DBA familiar with Oracle 11gR2 RAC, clusterware and ASM and about to become a Database Machine Administrator (DMA) – the folks who manage an Exadata system, the very first question you are bound to ask yourself – how much extra stuff you have to learn, isn't it? If you are wondering about how much of your own prior knowledge you can apply, what is different in Exadata, what makes it special, fast and efficient, and a whole lot of other questions, this article will explain all that. Here you will learn about the magic behind the premise of the Exadata database machine.

What is Exadata

Exadata is a consolidated appliance containing all those components that make up a database system – Storage, Flash Disks, Database Servers, Infiniband Switches, Ethernet Switches and KVM (some models). Yet, it is not an appliance. Why? Because of two very critical and important reasons: it has additional software to make it a better database machine and the components, which are engineered to work very well together, can be managed independently and not as a sealed blackbox. That's why Oracle calls it a Database Machine (DBM); not an appliance or a server. Although the components can be managed separately it is possible (and advisable) to manage the entire system as a whole. The administrators are not called DBAs or system admins; but by a special term – Database Database Machine Administrator (DMA).

Oracle Database System

Before we go down to the details, let's first start with a basic description of how an Oracle database works. The actual data is stored on the datafiles on the disk. However, when the user selects from the database, the data does not come from the disks directly. Instead, the data is sent to a memory area called database buffer cache. The smallest amount of data addressable by an Oracle database is a database block, which is generally 8 KB in size but could be as small as 2 KB or as large as 32 KB based on how the DBA has configured it. A single datablock may hold several rows of a table. When the user selects a table, the entire block, not a specific set of rows, are selected from the datafiles and moved to the buffer cache. Referring to Figure 1, suppose the user wants to select the rows of all the customers who are angry. The following query would make it happen:

```
SELECT NAME FROM CUSTOMERS WHERE STATUS = 'ANGRY';
```

However, the data stored in the datafiles do not have any knowledge of the status column inside the files. Instead database server process picks up a block from the datafile and places it in the buffer cache. From the buffer, it would then extract the rows that satisfy the condition and return them to the user.

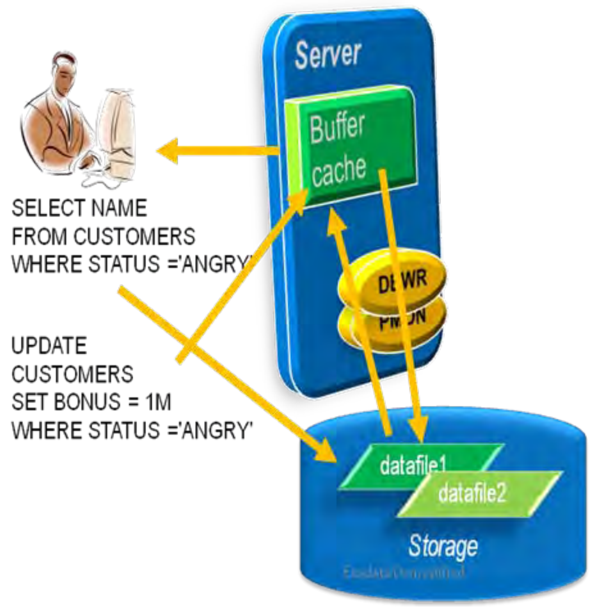

Server
Buffer cache
DBWR
PMON
SELECT NAME FROM CUSTOMERS WHERE STATUS ='ANGRY'
UPDATE CUSTOMERS SET BONUS = 1M WHERE STATUS ='ANGRY'
datafile1
datafile2
Storage

Figure 1. Oracle Instance

Suppose at this stage, the user wants to placate the angry customers by giving them 1 million bonus points. This can be done by issuing the following query:

```
UPDATE CUSTOMERS SET BONUS = 1M WHERE STATUS = 'ANGRY';
```

This update will not make changes in the datafile; but in the buffer cache. Later a different process, called Database Buffer Writer (DBWR) copies the buffers from the cache to the datafiles to make them consistent. This process is just of the many required to manage a database instance. Other processes such as System Monitor (SMON), process monitor

(PMON), etc. make sure the Oracle database system works as expected. The combination of these processes and memory areas such as buffer cache is known as an *Oracle Instance*.
In a Real Application Cluster (RAC) database system, there are two (or more) machines working on the same physical database system. Each machine runs an Oracle Instance. Since each instance can

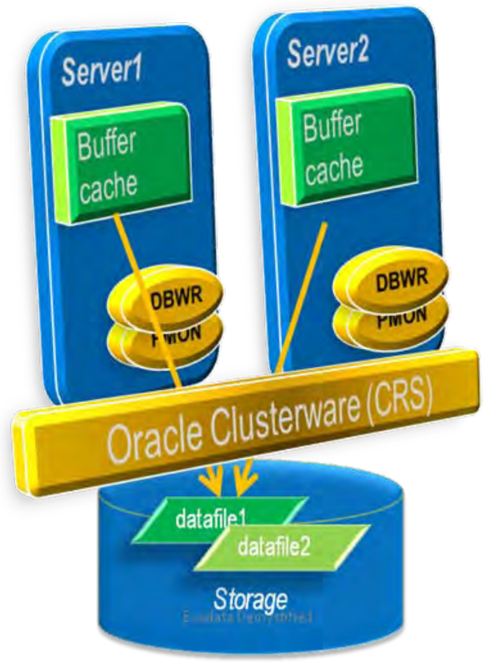


Figure 2. Oracle RAC

potentially write buffers to the disk, there is a risk of DBWR process of one machine overwriting the changes in the datafile made by the other leading to data corruption. To avoid this situation, a special software known as Oracle Clusterware coordinates the changes made by different machines in the same cluster. This is also known as Cluster Ready Services (CRS), as shown in Figure 2.

Limitations in the Current Model

Now that you understand how the traditional Oracle model works, let's see why there is an inherent bottleneck in the current design. The following components make up the processing parts of an Oracle database system

- CPU – the processing unit where the Oracle database software is installed and where the processes of the Oracle Instance run. The processes send instructions to the storage to get the datablocks.
- Memory – in the same system where the software runs and in which the database instance's memory areas (buffer cache, etc.) are located. When the datablocks are retrieved from the disk, they are stored here.
- Network – enables the communication between the CPU and the storage; could be Ethernet (for NAS) or fiber (for SANs). The datablocks come over this pathway.
- I/O Controller – the subsystems that enable flow of storage related information; examples include I/O Cards, Front End Adapter (FEA) cards, etc.
- Storage – the actual disks and the software system that manages them (e.g. SAN and NAS, etc.). This layer sends the datablocks as requested by the process running on the CPU.

The performance of the database system depends on the effectiveness of these components. The faster the component is, the better the performance. Over the years, we have seen the performance of these components going up by leaps and bounds, e.g. CPU powers have risen from KHz to GHz ranges; memory has become orders of magnitude faster and more expansive; I/O controllers appeared making the access faster and networks changed from bits per second to today's ubiquitous 10 Gbits/sec. However the story is different for storage – the disks haven't seen that kind of dramatic evolution. The law of physics comes in the way – the disks can rotate only so fast and not more.
While most of the components have become exponentially faster, they are still part of a chain and storage is the weakest link impacting the performance of the database system. Unless the storage becomes comparably faster, the overall performance will not improve with faster processors or more memory. System designers generally employ three techniques:

- Putting cache on the SAN or NAS systems avoiding spinning disks
- Moving to a flash based storage from harddisks
- Increasing the buffer cache by adding more memory so that the database instance can find everything there without going to the disk

While they work, they are not really solutions in case of a database system. The success of SAN caches is built upon predictive analytics, where the SAN software determines if a specific part of the disk is accessed more and moves it to the cache. This works well, if a small percentage of the disks is accessed most often. The emphasis is on disk; not data. The SAN does not know anything about data inside the disks; it predicts the next hot areas of the disk using some heuristics, which may now work well for most database systems. Most database systems are way bigger than SAN caches; so a little portion of the disk being on the cache doesn't help. Besides, the SAN needs to get the data from the disk to the cache and that takes time leading to the I/O at the disk level being still high. SAN caches are excellent for filesystems or very small databases; not for a regular sized database.
Solid state disks (also called flash disks) are definitely faster than hard drives; but they are still too expensive to be practical. The third argument, adding more memory to the buffer cache seems to have some merit, at least on the surface. However, memory is still very expensive. Besides, how much memory is enough to accommodate a major portion of the database? If you have a 1 GB database and 1 GB buffer cache, you might assume that the whole database will fit in the memory. Unfortunately, it's not true. Oracle database fills up to 7 times the DB size in the buffer cache, as I have explained in my blogpost: http://arup.blogspot.com/2011/04/can-i-fit-80mb-database-completely-in.html. Trying to get as much memory to fit in a typical database may not be practical at all. So this option is non-viable.
So, where does that leave us? We need to think outside the box. Let's revisit the problem. A typical query may:

- Select 10% of the entire storage
- Use only 1% of the data it gets

To gain performance, adding CPU and memory is not the answer, unless you have pockets as deep as Mariana Trench. The feasible solution is the database needing to get less from the storage, which will make the storage more efficient elevating it from the weakest link position. To enable this, we can't have the storage get all the blocks of the table from the spinning disks and send them to the CPU. We need to have some filtering at the storage level, i.e. the storage must be cognizant of the data it stores and it must understand what is being requested. This is what Exadata enables to bring that performance. Let's see how that works.

Magic # 1 Intellegent Communication

Since we identified that the storage must be smart enough to filter at the disk level and return only what is relevant, it has to get that instruction clearly from the CPU. In other words, the CPU and IO must speak the same language. In Exadata, a special type of protocol, called iDB (Intelligent Protocol) allows that communication, as shown in Fig 3. In line with the example shown earlier, the iDB protocol allows the CPU to send what the user wants (the "NAME" column) and the filter (the column "STATUS" = 'ANGRY') to the storage rather than asking to get all the data blocks of the CUSTOMERS table. iDB is built on the top of the Infiniband network protocol and media.

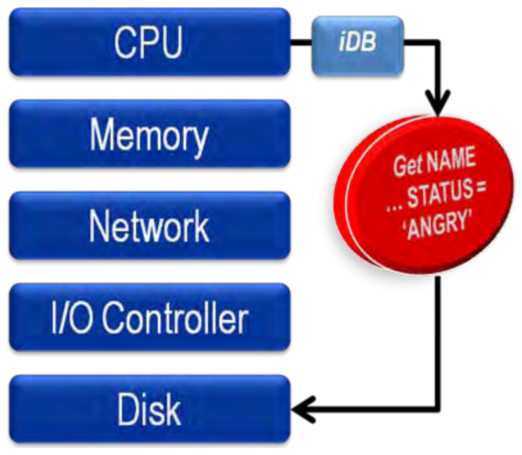


Figure 3. Intelligent iDB Communication

Magic # 2 Storage Cell Server

As you saw earlier, iDB is the medium that lets the storage know what the CPU wants; but how can the storage act on that information? Just a bunch of disks will not make it happen. This calls for some type of processing to be present at the storage layer. Fig 4 shows the conceptual layout of the storage. The physical disks are attached to a server known as Storage Cells. The Storage Cells, which are Sun blade servers, run a special software called Exadata Storage Server (ESS) that can process the iDB calls and send the data back as needed.

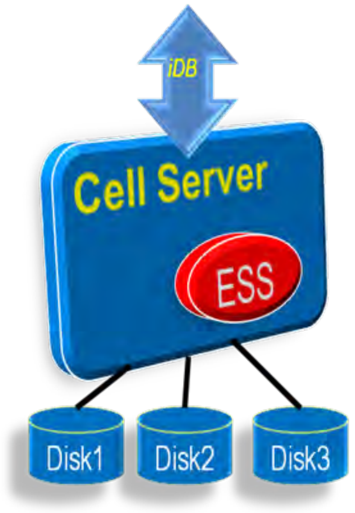


Figure 4. Storage Cell

Magic # 3 Storage Indexes

Now that you know the ESS software can understand what the user is asking for and then filter at the storage layer, the next logical question is how it can do so. This is where the third magic of Exadata comes in. Consider a table where a column named RATING is queried as follows:

```
SELECT NAME FROM CUSTOMERS WHERE RATING = 1;
```

The ESS server needs to know which parts of the disks to search for these values. It divides the disk into 1 MB parts and stores the minimum and maximum values of the column for that part. Refer to the Figure 5. Suppose the occupied part of the disk is 4 MB meaning there will be 4 parts of 1 MB each. The figure shows how the minimum and maximum values of each part are recorded. Using this data, the storage server can easily determine that inside area 1, the minimum value of RATING is 3; so a row with RATING = 1 will not be found there. Therefore the ESS software will skip searching that part of this disk for that row. Similarly it will skip part 2 and part 4 of the disk. Only part 3 will have some rows. Using these min and max values, the ESS software can eliminate searching for 75% of the disk, reducing I/O by 75%! This powerful feature is known as Storage Indexes.

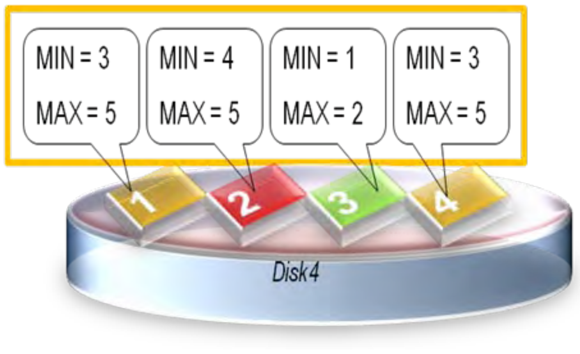


Figure 5. Storage Indexes

Storage Indexes are different from normal Oracle database indexes. Normal indexes reside in the database; storage indexes are in the memory on the storage cells. Normal indexes tell the database which specific blocks where a row can be found; storage indexes tell the storage server where the rows will not be found.
But the storage indexes do not work every time. The mechanism that allows the use of power of the Storage Cells is called Smart Scan. If you want to
find out whether Smart Scan and Storage Indexes are being used, you can use the following query:

```
select decode(name,
 'cell physical IO bytes saved by storage index',
    'SI Savings',
 'cell physical IO interconnect bytes returned by smart scan',
    'Smart Scan'
  ) as stat_name, value/1024/1024 as stat_value
from v$mystat s, v$statname n
where s.statistic# = n.statistic#
and n.name in (
  'cell physical IO bytes saved by storage index',
  'cell physical IO interconnect bytes returned by smart scan');
```

Here is a sample output:

```
STAT_NAME  STAT_VALUE
---------- ----------
SI Savings   0.000
Smart Scan   0.000
```

In this Smart Scan and Storage Indexes did not yield any savings. Why not? There are several reasons. First, a simple explanation: Smart Scans have been disabled by database level parameters

```
cell_offload_processing = true;
_kcfis_storageidx_disabled = true;
```

Otherwise, the pre-requisites for Smart Scan haven't been satisfied. Here are the pre-reqs:

- Direct Path Operations
  - Full Table or Full Index Scan
  - 0 Predicates
- Simple Comparison Operators

Other reasons are:

- Cell is not offload capable
- The diskgroup attribute cell.smart_scan_capable set to FALSE
- Smart Scan is not possible on clustered tables, IOTs, etc.

Magic # 4 Smart Flash

These are flash cards presented as disks; not as memory to the Storage Cells. The data, after being retrieved from the disk is stored in the Smart Flash. When the same data is requested, the ESS server satisfies the request from the Smart Flash instead of the disk. So, they are very similar to SAN cache; but Oracle, not the SAN, controls what goes in there and how long data stays in them. For instance, Oracle can determine the tables or indexes (not areas of the disk) that are accessed frequently and moves them to Smart Flash. These could come from various parts of the disks. As a DBA, you can also put specific objects in Smart Flash if you think they will benefit.
While Smart Flash makes reading faster, writing is still done to the spinning disks. So the database writes do not benefit from Smart Flash.

Magic # 5 Operation Offloading

The ESS software does much more than just filtering at the storage level. It can perform some of the operations directly at the storage level and return the results to the database processes, making the overall performance quite high. This is known as cell offloading. Some of the processes offloaded are:

- Bloom Filters
- Functions Offloading. You can know which functions can be offloaded by querying the view V$SQLFN_METADATA
- Decompression with Hybrid Columnar Compression. The compression operations handled by database nodes

- Expansion of Virtual Columns

There are some additional elements for the supercharged performance as well; but these are the major ones.

Components

Now that you know the magic behind the efficiency of Exadata, let's examine the components that make up the Exadata frame. Fig 6 shows the various components and how they are represented in the Exadata frame. The CPU and memory are located in Sun blade servers, where the database and clusterware softwares run. These are called Compute Nodes. The Operating System is either Oracle Enterprise Linux or Solaris. On the bottom of the stack, the I/O controllers and disks are located on Sun blades as well, known as Storage Cells or Cell servers. It runs Oracle Enterprise Linux software. The Exadata Storage Server (ESS) runs here. The network component is implemented on Infiniband and Ethernet switches and infrastructure.

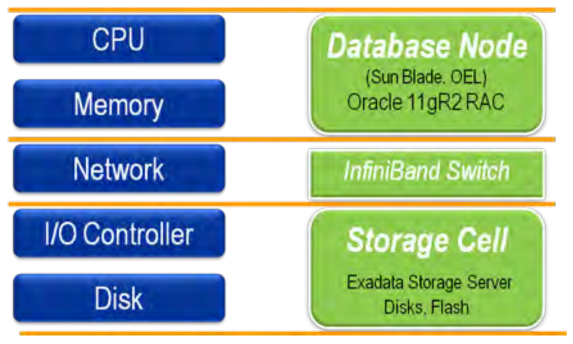


Figure 6. Components of Exadata

Three configuration types are available in Exadata: full rack, half rack, or quarter rack. The architecture is identical across all three types but the number of components differs. On a full rack, there are 14 Storage Cells and 8 Compute Nodes . The other configurations have proportionally number of components.

Network

Infiniband circuitry – the cells and nodes are connected through infiniband for speed and low latency. There are 3 infiniband switches for redundancy and throughput. Note: there are no fiber switches since there is no fiber component.

Ethernet switch – the outside world can communicate via infiniband, or by Ethernet. There is a set of Ethernet switches with ports open to the outside. The clients may connect to the nodes using Ethernet. DMAs and others connect to the nodes and cells using Ethernet as well. Backups are preferably via

Storage Cells

Each cell has 12 disks. Depending on the configuration, these disks are either 600GB high performance or 2TB high capacity (GB here means 1 billion bytes, not 1024MB). You have a choice in the disk type while making the purchase. Each cell also has 384GB of flash disks. These disks can be presented to the compute nodes as storage (to be used by the database) or used a secondary cache for the database cluster (called smart cache).

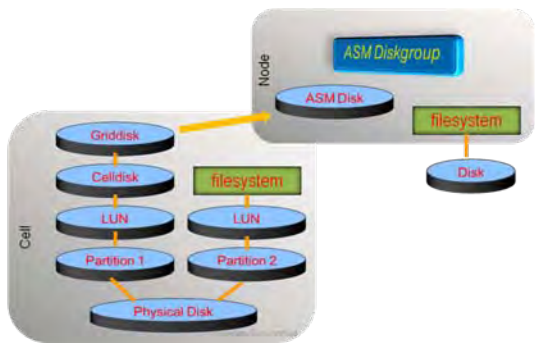


Figure 7. Disk Layout

Since the cells have the disks, how do the database compute nodes access them - or more specifically, how do the ASM instances running on the compute nodes access the disks? Well, the disks are presented to cells only, not to the compute nodes. The compute nodes see the disks through the cells. For the lack of a better analogy, this is akin to network-attached storage. (Please note, the cell disks are not presented as NAS; this is just an analogy.)

Two of the 12 disks are also used for the home directory and other Linux operating system files. These two disks are divided into different partitions as shown in Figure 7 below. he physical disks are divided into multiple partitions. Each partition is then presented as a LUN to the cell. Some LUNs are used to create a filesystem for the OS. The others are presented

as storage to the cell. These are called cell disks. The cell disks are further divided as grid disks, ostensibly referencing the grid infrastructure the disks are used inside. These grid disks are used to build ASM Diskgroups, so they are used as ASM disks. An ASM diskgroup is made up of several ASM disks from multiple storage cells.
If the diskgroup is built with normal or high redundancy (which is the usual case), the failure groups are placed in different cells, as shown in Figure 8. As a result, if one cell fails, the data is still available on other cells. Finally the database is built on these diskgroups.

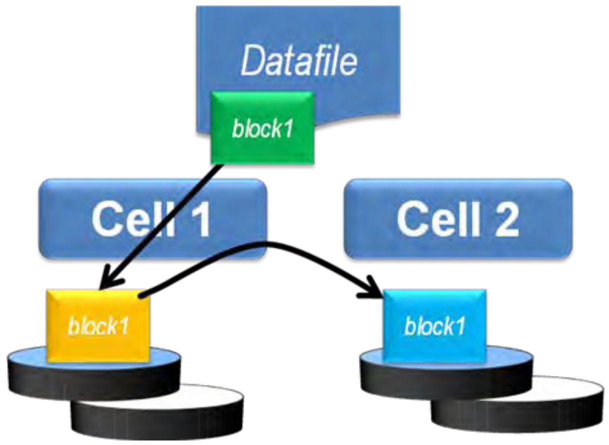


Figure 8. ASM Redundancy in Exadata Administration

Now that you understand the building blocks of Exadata, hopefully you have got an idea of what type of commands are used where. Here is a summary of the commands necessary on various building blocks.

- Compute Nodes
  - Linux Management – vmstat, mpstat, top, fdisk, etc.
  - ASM Commands – SQL*Plus, ASMCMD, ASMCA
  - Database Commands – startup, alter database, etc.
  - Clusterware Commands – CRSCTL, SRVCTL, etc.
- Storage Cells
  - Linux Management – vmstat, mpstat, top, fdisk, etc.
  - CellCLI – command line tool to manage the Cell

How difficult are these commands to master? Actually, they are not that difficult for an Oracle DBA. Here is a breakdown of the skills expected of a Database Machine Administrator (DMA).

Skill	**Needed**
System Administrator	15%
Storage Administrator	0%
Network Administrator	5%

Database Administrator	60%
Cell Administration	20%

As you can see, if you have been an Oracle 11.2 RAC Database Administrator, you already know 60% of the beast anyway. For the others, it's quite easy to master. For the linux components, visit my 5-part article series on Oracle Technology Network, " Linux Commands" http://bit.ly/k4mKQS. The Cell Administration is new for anyone, including seasoned Storage Administrators. For that, you can refer to my 4-part Exadata Command Reference article series http://bit.ly/lljFl0, also on OTN. All the articles are free. Using these articles you can master the rest 40% very easily to be successful Database Machine Administrator.

Frequently Asked Questions

Let's go through some quite frequently asked questions on Exadata.

Q: Do clients have to connect using Infiniband?
A: No; Ethernet is also available. However, Infiniband allows more throughput and lower latency.

Q: How do you back up Exadata?
A: Through the normal RMAN Backup, just like an ordinary Oracle RAC Database. No special tricks needed for Exadata.

Q: How do you enable Disaster Recovery?
A: Since the storage does not support any hardware based replication, the only solution is Data Guard. The standby could be a non-Exadata box as long as it runs Oracle Enterprise Linux and Oracle 11.2 software. However, you can't use any Exadata specific features such as Smart Scan.

Q: Can I install any other software?
A: Nothing is allowed on Cells. On Compute nodes software can be installed but the space is limited.
Generally you may want to install client software like Golden Gate, etc.

Q: How do I monitor it?
A: There are several ways to monitor it. Oracle Enterprise Manager Grid Control has a plugin specific ally for Exadata. Besides the CellCLI commands (available in Exadata Comamnd Reference article series shown above) is used to monitor Cells. Ordinary SQL commands are used to monitor the database.

Conclusion

The purpose of this article was to explain various components of Exadata and how they play together to bring the fast performance. In summary:

- Exadata has an Oracle Database running 11.2 RAC
- The storage cells have added intelligence about data placement
- The compute nodes run Oracle database and Grid Infrastructure software
- Nodes communicate with Cells using iDB which can send more information on the query
- Smart Scan, when possible, reduces I/O at cells for Direct Path operations
- Cell is managed by CellCLI commands
- DMA skills = 60% RAC DBA + 15% Linux + 20% CellCLI + 5% miscellaneous

Hopefully you will now appreciate the engineering behind the Exadata Database Machine making it faster than a regular Oracle database running on same or similar hardware. In this article I attempted to make sure you understand the workings under the hood so that it is no longer a "magic" but simply technology at work. Understanding this technology helps you shed the fear and uncertainty and enable you to assume the role of a Database Machine Administrator with aplomb. Good luck and happy DMA'ing.

About the Author

Arup Nanda (arup@proligence.com) has been an Oracle DBA for last 16 years and now a DMA, has seen all aspects of Oracle Database management from performance tuning to backup recovery. He works as principal global database architect at a major New York area corporation. He has written 4 books, 500+ article, presented 300+ sessions, blogs regularly at arup.blogspot.com, tweets at @arupnanda and spends time with his family in Connecticut when not doing any of these. He was the recipient of Oracle's DBA of the Year award in 2003.

MONITOR ORACLE
PERFORMANCE ON VMWARE

Get Visibility Into VMware

New! Confio IgniteVM™ is the only Oracle performance management solution that gives DBAs visibility from the database instance down to physical host and storage layers.

- See physical I/O, CPU, and memory resources available
- Be aware of dynamic virtual server changes
- Gain non-intrusive access to VMware statistics that directly impact Oracle

igniteVM CONFIO

IgniteVM is built from the same technology as the award-winning Ignite8 from Confio Software, the 2010 SQL Server Magazine Editor's Best Winner. Learn about IgniteVM and try it free. www.confio.com/SQLVM

Ignite by Confio Software, Boulder Colorado.

NYOUG 2012 Sponsors

The New York Oracle Users Group wishes to thank the following companies for their generous support.

F5 (www.f5.com)
Oracle (www.oracle.com)
Quest Software (www.quest.com)

Contact Caryl Lee Fisher for vendor information, sponsorship, and benefits

#1 Middleware

✓ **#1 in Application Servers**

✓ **#1 in Application Infrastructure Suites**

✓ **#1 in Enterprise Performance Management**

ORACLE®

oracle.com/goto/middleware
or call 1.800.ORACLE.1