ORACLE®

DBA Best Practices: A Primer on Managing Oracle Databases

Mughees A. Minhas
Sr. Director of Product Management
Database and Systems Management

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Top 3 DBA Activities

1. Performance Diagnosis
2. SQL Optimization
3. Space Management

Top 3 DBA Activities

Top 10 Best Practices for

1. Performance Diagnosis
2. SQL Optimization
3. Space Management

Source: IOUG DBA Survey

- ☑ **Performance Diagnostics**
- ☑ SQL Optimization
- ☑ Space Management
- ☑ Q & A

Performance Diagnostics

ORACLE

Performance Diagnostics Topics

- Key Concepts
- Automatic System Diagnostics
- Manual System Diagnostics
- Advanced Topics
  - Targeted analysis
  - Comparative analysis

Key Concepts

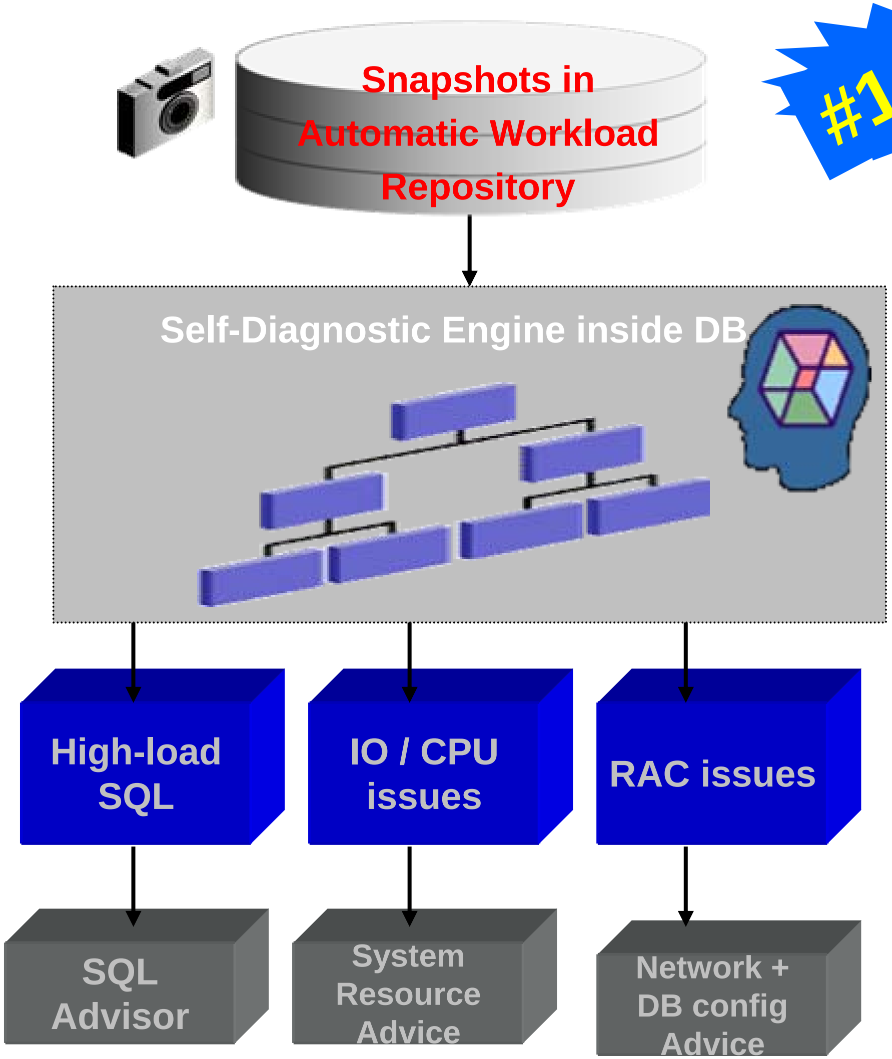
- **DB Time**
  - Total time in database calls by foreground sessions
  - Includes CPU time, IO time and non-idle wait time
  - DB Time <> response time
  - Total DB time = sum of DB time for all active sessions

➢ **Goal: To Reduce Total DB time**

- Active Session
  - Session currently spending time in a database call, i.e., accruing DB time
- **Average Active Sessions** $= \dfrac{\textbf{DB Time}}{\textbf{Wall-Clock (Elapsed) Time}}$

- Average Active Sessions is a new metric for measuring DB load

Automatic System Diagnostics using ADDM



Use ADDM (Automatic Database Diagnostic Monitor) for database-wide performance diagnostic

- Self-diagnostic engine in the database
- Helps resolve current and past problems
- In 11g, a RAC specialist as well!
- Provides impact and benefit analysis, non problem areas
- Runs proactively out of the box, reactively when required

Cluster Database: racdb > Advisor Central > Logged in As SYS

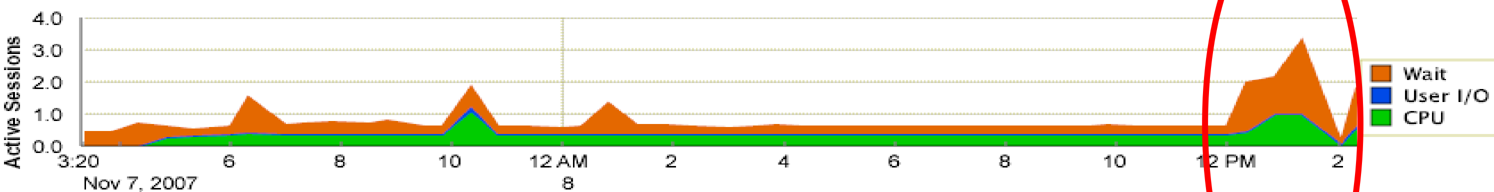
Automatic Database Diagnostic Monitor (ADDM)

Page Refreshed **Nov 8, 2007 2:44:24 PM PST** Refresh

Database Activity

Run ADDM | Finding History

The icon selected below the graph identifies the ADDM analysis period. Click on a different icon to select a different analysis period.



Zoom

TIP For an explanation of the icons and symbols used in this page, see the Icon Key

ADDM Performance Analysis

Task Name: ADDM:634508306_1071 (End Time:Nov 8, 2007 1:10:32 PM)

Filters | View Snapshots | View Report

Task Owner **SYS** | Average Active Sessions **3.9** | Period Start Time **Nov 8, 2007 1:00:32 PM PST** | Period Duration (minutes) **10** | Instance **racdb**

Impact (%)	Finding	Affected Instances	Occurrences (last 24 hrs)
90.6	Top SQL by DB Time		126 of 141
45.9	Unusual "Concurrency" Wait Event		124 of 141
32.1	Sequence Usage	2 of 2	5 of 141
31.1	Session Connect and Disconnect	2 of 2	124 of 141
28.7	CPU Usage	2 of 2	17 of 141
3.6	Unusual "Other" Wait Event		28 of 141

Informational Findings

Affected Instances

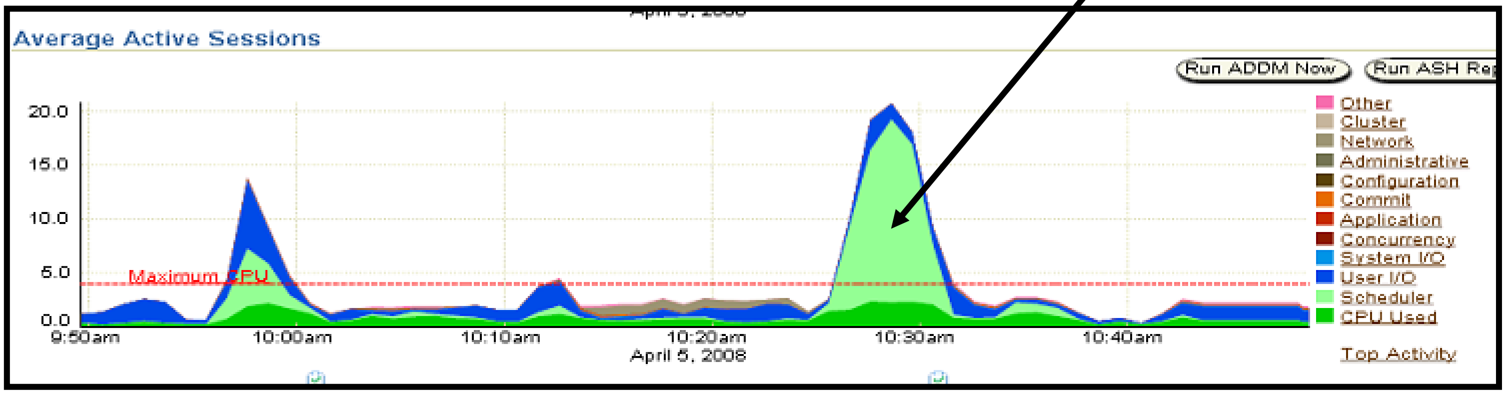
Name	Impact (%)	Status
racdb_racdb1	65	ANALYZED
racdb_racdb2	35	ANALYZED

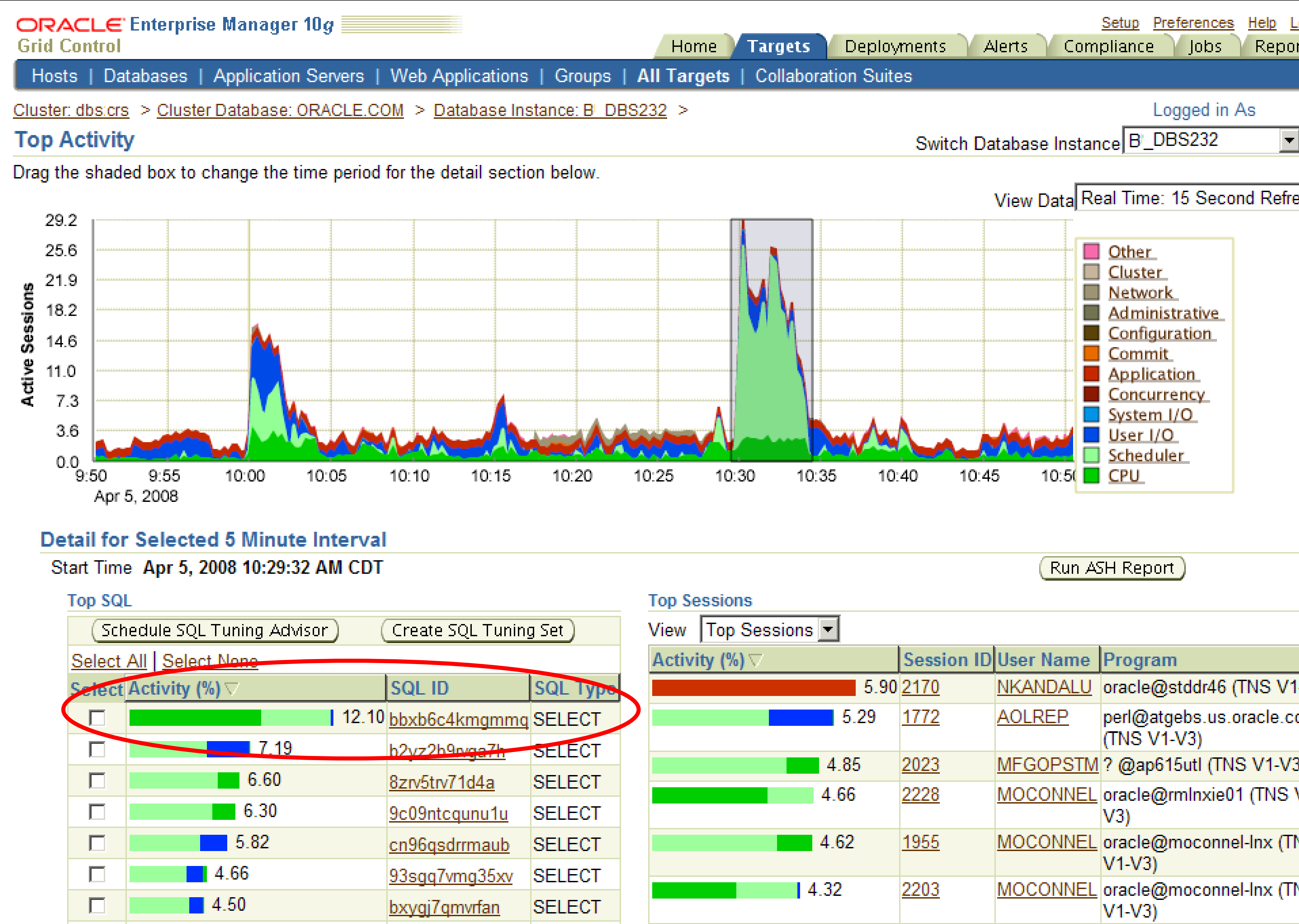
Database | Setup | Preferences | Help | Logout

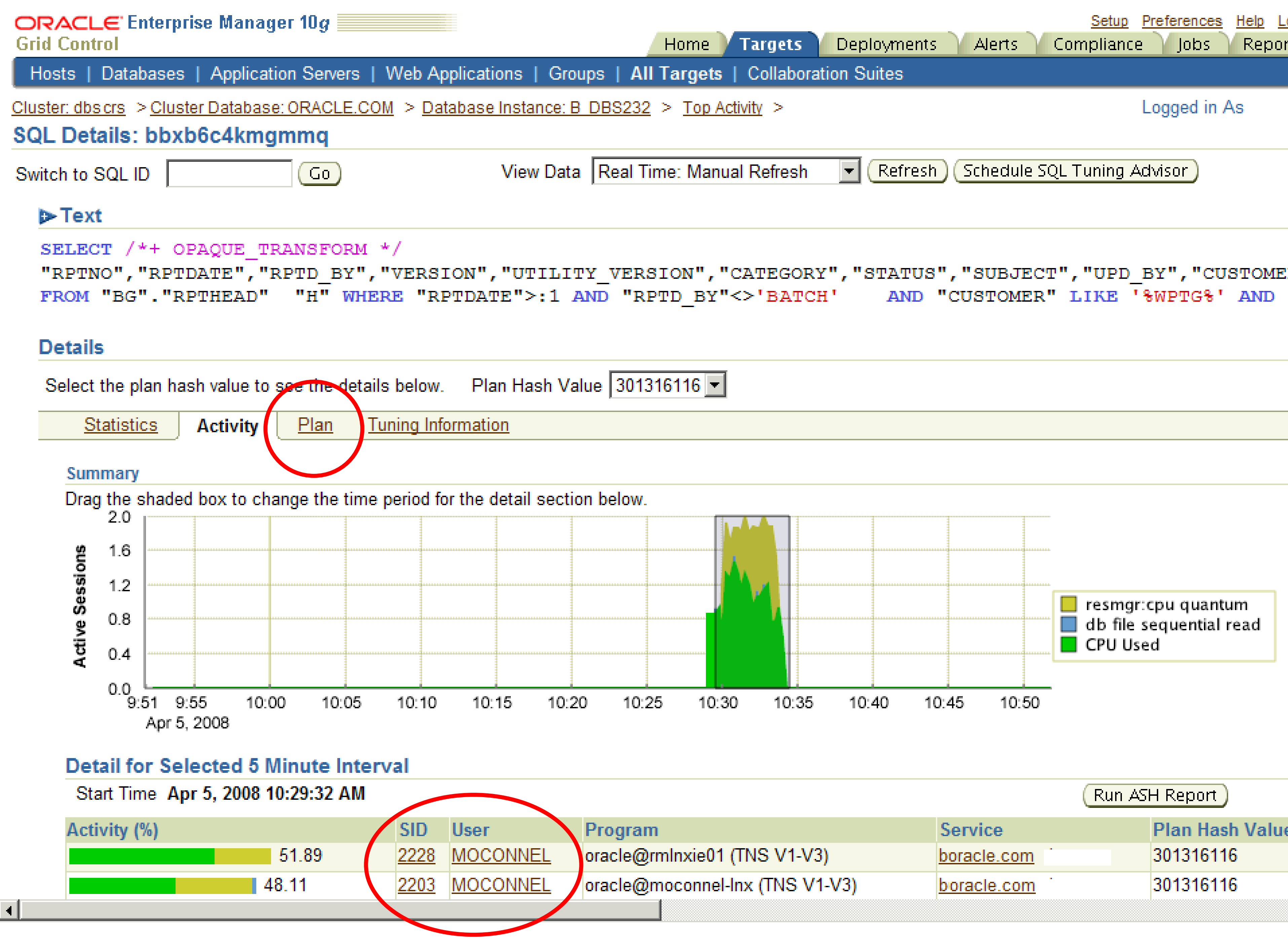
Manual Performance Diagnostics

- EM Performance Page facilitates manual performance analysis
- Method (Advanced):
  - Observe Average Active Sessions graph
  - “Click on the Big Stuff”
- Answers the “who” and “what” of the problem
  - **Who** is slowing down the system?
  - **What** is that person/process doing?

Click!

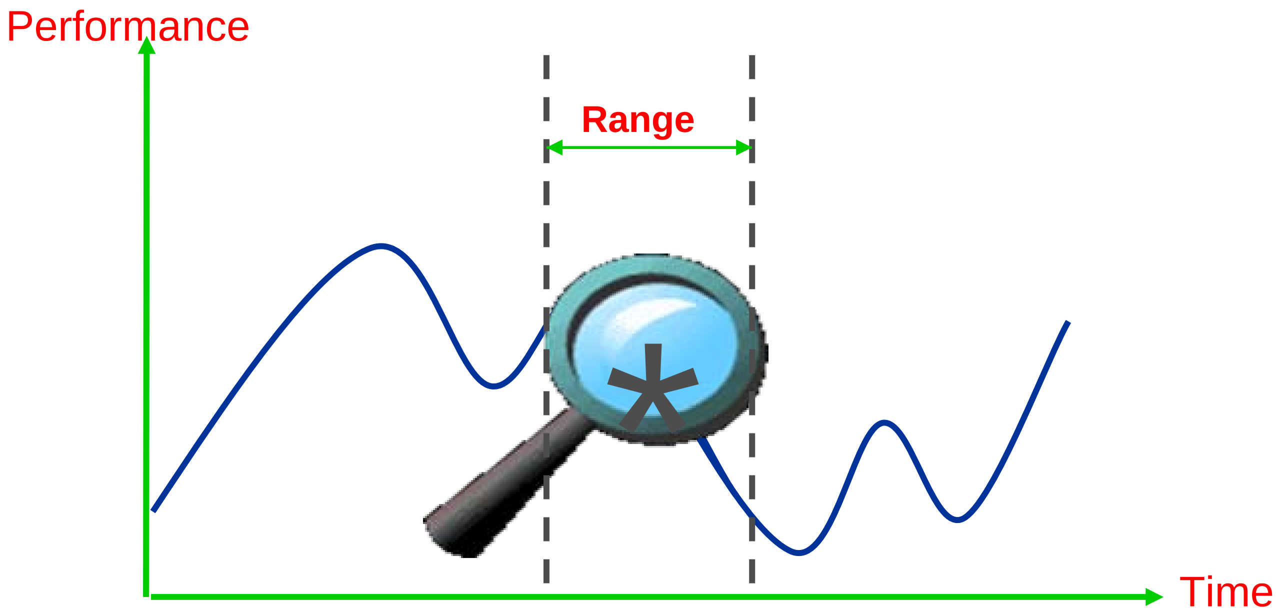



ORACLE Enterprise Manager 10g
Grid Control
Setup Preferences Help
Home Targets Deployments Alerts Compliance Jobs
Hosts | Databases | Application Servers | Web Applications | Groups | All Targets | Collaboration Suites
Cluster: dbs.crs > Cluster Database: ORACLE.COM > Database Instance: B_DBS232 >
Logged in As
Top Activity
Switch Database Instance B_DBS232
Drag the shaded box to change the time period for the detail section below.
View Data Real Time: 15 Second Refre
Active Sessions
29.2
25.6
21.9
18.2
14.6
11.0
7.3
3.6
0.0
9:50 9:55 10:00 10:05 10:10 10:15 10:20 10:25 10:30 10:35 10:40 10:45
Apr 5, 2008
Other
Cluster
Network
Administrative
Configuration
Commit
Application
Concurrency
System I/O
User I/O
Scheduler
CPU
Detail for Selected 5 Minute Interval
Start Time Apr 5, 2008 10:29:32 AM CDT
Run ASH Report
Top SQL
Schedule SQL Tuning Advisor
Create SQL Tuning Set
Select All | Select None
Select Activity (%) SQL ID SQL Type
12.10 bbxb6c4kmgmmq SELECT
7.19 b2yz2h9nga7h SELECT
6.60 8zrv5trv71d4a SELECT
6.30 9c09ntcqunu1u SELECT
5.82 cn96qsdrrmaub SELECT
4.66 93sgq7vmg35xv SELECT
4.50 bxygj7qmvrfan SELECT
Top Sessions
View Top Sessions
Activity (%) Session ID User Name Program
5.90 2170 NKANDALU oracle@stddr46 (TNS V1
5.29 1772 AOLREP perl@atgebs.us.oracle.c (TNS V1-V3)
4.85 2023 MFGOPSTM ? @ap615utl (TNS V1-V3
4.66 2228 MOCONNEL oracle@rmlnxie01 (TNS V3)
4.62 1955 MOCONNEL oracle@moconnel-lnx (TN V1-V3)
4.32 2203 MOCONNEL oracle@moconnel-lnx (TN V1-V3)


ORACLE Enterprise Manager 10g
Grid Control
Setup Preferences Help
Home Targets Deployments Alerts Compliance Jobs
Hosts | Databases | Application Servers | Web Applications | Groups | All Targets | Collaboration Suites
Cluster: dbscrs > Cluster Database: ORACLE.COM > Database Instance: B_DBS232 > Top Activity >
Logged in As
SQL Details: bbxb6c4kmgmmq
Switch to SQL ID
Go
View Data
Real Time: Manual Refresh
Refresh
Schedule SQL Tuning Advisor
Text
SELECT /*+ OPAQUE_TRANSFORM */
"RPTNO","RPTDATE","RPTD_BY","VERSION","UTILITY_VERSION","CATEGORY","STATUS","SUBJECT","UPD_BY","CUSTOME
FROM "BG"."RPTHEAD" "H" WHERE "RPTDATE">:1 AND "RPTD_BY"<>'BATCH' AND "CUSTOMER" LIKE '%WPTG%' AND
Details
Select the plan hash value to see the details below.
Plan Hash Value 301316116
Statistics
Activity
Plan
Tuning Information
Summary
Drag the shaded box to change the time period for the detail section below.
Active Sessions
2.0
1.6
1.2
0.8
0.4
0.0
9:51
9:55
10:00
10:05
10:10
10:15
10:20
10:25
10:30
10:35
10:40
10:45
10:50
Apr 5, 2008
resmgr:cpu quantum
db file sequential read
CPU Used
Detail for Selected 5 Minute Interval
Start Time Apr 5, 2008 10:29:32 AM
Run ASH Report
Activity (%) SID User Program Service Plan Hash Value
51.89 2228 MOCONNEL oracle@rmlnxie01 (TNS V1-V3) boracle.com 301316116
48.11 2203 MOCONNEL oracle@moconnel-lnx (TNS V1-V3) boracle.com 301316116

Targeted Performance Analysis

- **Use ASH (Active Session History) for targeted performance analysis into different dimensions:**
  - 1st dimension by a Time, then by
    - SQL ID
    - Session ID
    - Wait Class
    - Service, Module, Action, Client ID



ASH Report: Over 5 mins by a SQL

Main Sections

ASH Report For BUGAP/bug1ap

(1 Report Target Specified)

DB Name	DB Id	Instance	Inst num	Release	RAC	Host
BUGAP	1679034986	bug1ap	1	10.2.0.1.0	YES	dbs232

CPUs	SGA Size	Buffer Cache	Shared Pool	ASH Buffer Size
4	2,576M (100%)	1,200M (46.6%)	1,109M (43.0%)	8.0M (0.3%)

	Sample Time	Data Source
Analysis Begin Time:	21-Sep-06 13:13:20	V$ACTIVE_SESSION_HISTORY
Analysis End Time:	21-Sep-06 13:18:20	V$ACTIVE_SESSION_HISTORY
Elapsed Time:	5.0 (mins)	Missing 1.0 mins (20%) of activity
Sample Count:	1,330	
Average Active Sessions:	4.43	
Avg. Active Session per CPU:	1.11	
Report Target:	SQL_ID like 'cyaj7dkrbqs95'	4% of total database activity

ASH Report

- Top Events
- Load Profile
- Top SQL
- Top Sessions
- Top Objects/Files/Latches
- Activity Over Time

ASH Report:
Top Events for that SQL

Top User Events

Event	Event Class	% Activity	Avg Active Sessions
db file sequential read	User I/O	68.80	3.05
gc buffer busy	Cluster	12.33	0.55
buffer busy waits	Concurrency	9.25	0.41
read by other session	User I/O	5.64	0.25
gc cr disk read	Cluster	1.28	0.06

ASH Report:
Activity for that SQL over the same 5 mins

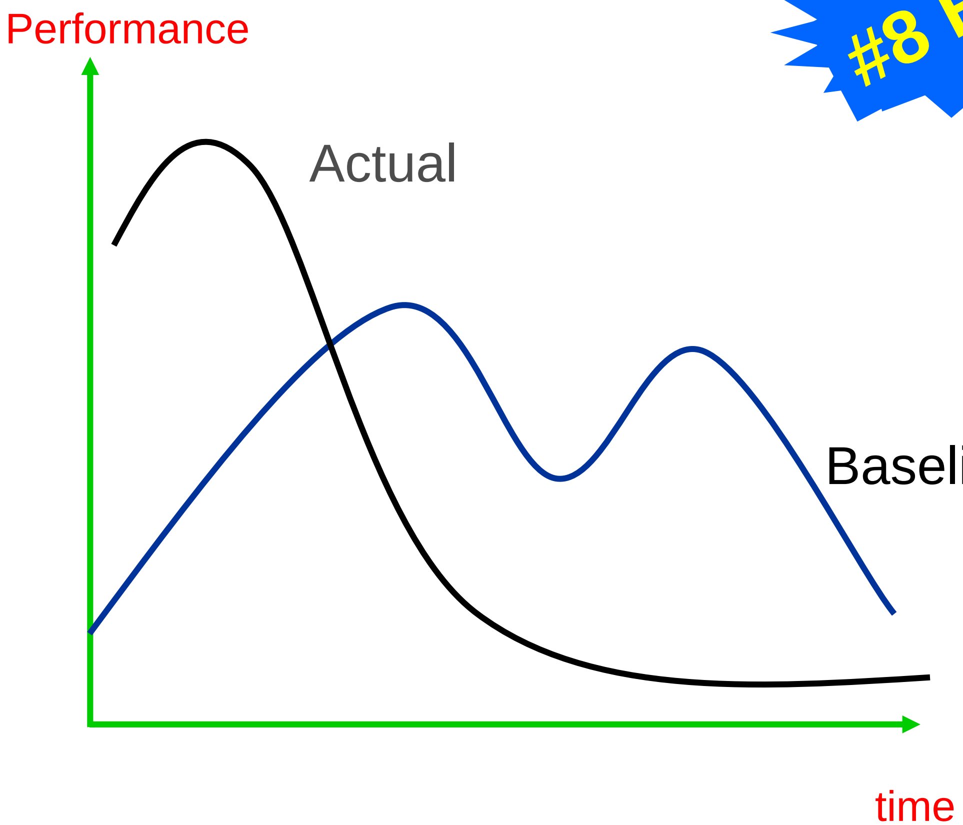
Activity Over Time

- Analysis period is divided into smaller time slots
- Top 3 events are reported in each of those slots
- 'Slot Count' shows the number of ASH samples in that slot
- 'Event Count' shows the number of ASH samples waiting for that event in that slot
- '% Event' is 'Event Count' over all ASH samples in the analysis period

Slot Time (Duration)	Slot Count	Event	Event Count	% Event
13:14:00 (1.0 min)	220	db file sequential read	163	12.26
		gc buffer busy	27	2.03
		buffer busy waits	12	0.90
13:15:00 (1.0 min)	295	db file sequential read	222	16.69
		gc buffer busy	32	2.41
		buffer busy waits	22	1.65
13:16:00 (1.0 min)	305	db file sequential read	211	15.86
		gc buffer busy	43	3.23
		read by other session	23	1.73
13:17:00 (1.0 min)	295	db file sequential read	199	14.96
		gc buffer busy	35	2.63
		buffer busy waits	28	2.11
13:18:00 (20 secs)	100	db file sequential read	46	3.46
		buffer busy waits	27	2.03
		gc buffer busy	14	1.05

ORACLE

Comparative Performance Analysis



- **Use Automatic Workload Repository (AWR) Baseline for comparative performance analysis** to
  - Guide set alert thresholds
  - Monitor performance
  - Compare advisor reports
- Enables performance comparison of two periods
- Makes analysis of workload variations and performance diagnosis easier
- Automatic creation and management of reference AWR baselines
- Out-of-box Moving Window AWR Baseline in 11g

AWR Compare Period Report

WORKLOAD REPOSITORY COMPARE PERIOD REPORT

Snapshot Set	DB Name	DB Id	Instance	Inst num	Release	Cluster	Host	Std Block Size
First (1st)	PROD	1545480911	prod	1	11.1.0.6.0	NO	tdsrat01-d1.oracleads.com	8192
Second (2nd)	PROD	1545480911	prod	1	11.1.0.6.0	NO	tdsrat01-d1.oracleads.com	8192

Snapshot Set	Begin Snap Id	Begin Snap Time	End Snap Id	End Snap Time	Avg Active Users	Elapsed Time (min)	DB time (min)
1st	339	27-Mar-08 20:17:34 (Thu)	340	27-Mar-08 20:22:07 (Thu)	0.68	4.54	3.09
2nd	396	27-Mar-08 20:52:52 (Thu)	397	27-Mar-08 20:56:18 (Thu)	0.30	3.44	1.04
%Diff					*-55.88*	*-24.23*	*-66.40*

AWR Compare Period Report:
Configuration

Host Configuration Comparison

	1st	2nd	Diff	%Diff
Number of CPUs:	1	1	0	0.00
Physical Memory:	2972M	2972M	0M	0.00
Load at Start Snapshot:	.57	.55	-.02	-3.51
Load at End Snapshot:	.84	.72	-.12	-14.29
%User Time:	20.93	8.89	-12.04	-57.53
%System Time:	37.14	32.41	-4.73	-12.74
%Idle Time:	41.93	58.7	16.77	40.00
%IO Wait Time:	2.79	.25	-2.54	-91.04

System Configuration Comparison

	1st	2nd	Diff	%Diff
SGA Target:			0M	0.00
Buffer Cache:	236M	208M	-28M	-11.86
Shared Pool Size:	336M	356M	20M	5.95
Large Pool Size:	4M	4M	0M	0.00
Java Pool Size:	12M	20M	8M	66.67
Streams Pool Size:	8M	8M	0M	-0.06
Log Buffer:	5,076K	5,076K	0K	0.00
PGA Aggregate Target:	M	M	0M	0.00
Undo Management:	AUTO	AUTO		

AWR Compare Period Report:
Load Profile

Load Profile

	1st per sec	2nd per sec	%Diff	1st per txn	2nd per txn	%Diff
DB time:	0.68	0.30	-55.88	0.09	0.03	-66.67
CPU time:	0.36	0.20	-44.44	0.05	0.02	-60.00
Redo size:	141,784.30	186,369.33	31.45	18,478.25	18,542.89	0.35
Logical reads:	30,539.38	1,289.19	-95.78	3,980.09	128.27	-96.78
Block changes:	726.20	949.25	30.71	94.64	94.45	-0.20
Physical reads:	6,790.88	0.61	-99.99	885.03	0.06	-99.99
Physical writes:	2.88	1.68	-41.67	0.38	0.17	-55.26
User calls:	338.11	447.90	32.47	44.06	44.56	1.13
Parses:	15.58	17.39	11.62	2.03	1.73	-14.78
Hard parses:	0.83	0.24	-71.08	0.11	0.02	-81.82
Sorts:	4.57	9.52	108.32	0.60	0.95	58.33
Logons:	0.09	0.11	22.22	0.01	0.01	0.00
Executes:	344.89	449.90	30.45	44.95	44.76	-0.42
Transactions:	7.67	10.05	31.03			

	1st	2nd	Diff
% Blocks changed per Read:	2.38	73.63	71.25
Recursive Call %:	28.01	21.66	-6.35
Rollback per transaction %:	0.96	0.68	-0.28
Rows per Sort:	51.54	11.15	-40.39
Avg DB time per Call (sec):	0.00	0.00	-0.00

AWR Compare Period Report:

Top SQL by Elapsed Time

	Elapsed Time % of DB time					Elapsed Time (ms) per Exec		#Exec/sec (DB time)		CPU Time (ms) per Exec		Physical Reads per Exec		#Rows Processed per Exec		#Executions		#Plans	
SQL Id	1st	1st Total	2nd	2nd Total	Diff	1st	2nd	1st	2nd	1st	2nd	1st	2nd	1st	2nd	1st	2nd	1st/2nd/Both	SQL Text
b6v4z72bxvp2y	39.21	39.21	0.23	0.23	-38.98	1,967	4	0.20	0.59	893	1	50,001.08	0.05	3.00	3.00	37	37	1/1/2	SELECT count(pnum) ...
22x9qxj96n6vx	35.09	74.31	2.98	3.22	-32.11	7,236	207	0.05	0.14	3,282	134	129.11	1.56	31.00	31.00	9	9	1/1/2	SELECT /* DSS_Q54 */ ...
1vu8j8vxpak4v			12.44	15.66	12.44		862		0.14		363		4.56		1.00		9		BEGIN :1 := dbms_workload_repl...
gmtgm98c05ag1	3.73	78.04	13.99	29.65	10.26	0	0	467.45	1,391.37	0	0	0.00	0.00	1.00	1.00	86,748	86,748		INSERT into po values (:"SYS_...
auu0bcau5ff55			5.48	35.13	5.48		380		0.14		80		1.11		1.00		9	/1/1	SELECT XMLCONCAT(:B1 , DBMS ...

- ☑ Performance Diagnostics
- ☑ **SQL Optimization**
- ☑ Space Management
- ☑ Q & A

SQL Optimization

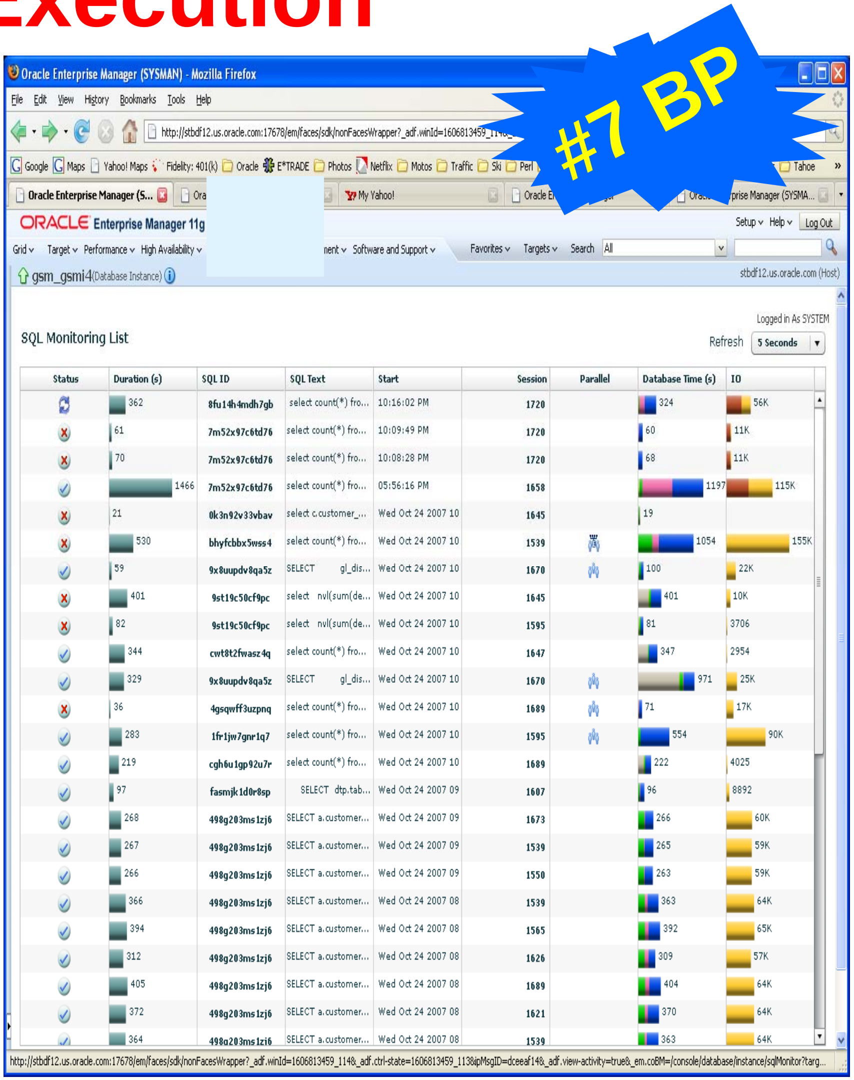
SQL Optimization Topics

- Manual SQL Tuning
- Automatic SQL Tuning
- Optimizer Statistics Management

Use Real-time SQL Monitoring to Understand SQL Execution

ORACLE DATABASE 11g

- Shows what's happening inside SQL execution
- Automatically monitors long running SQL
- Enabled out-of-the-box with no performance impact
- Monitors each SQL execution
- Exposes monitoring statistics
  - Global execution level
  - Plan operation level
  - Parallel Execution level



Demo

Real-time SQL Monitoring

SQL Monitoring Use Case: Big Plans

Monitored SQL Execution Detail

Text Report

Overview

SQL ID	65q6bpbqs9v0t
Execution Started	Thu Feb 14 2008 01:57:51 PM
Last Refresh Time	Thu Feb 14 2008 02:02:03 PM
Execution ID	16777216
Session	76
Fetch Calls	8

Time

Duration	4.2m
Database Time	4.2m
PL/SQL & Java	0.0s

IO & Wait Statistics

IO Count	1069K
Buffer Gets	913K
Wait Activity %	100

User I/O: direct path read temp - 114 samples (50%)

Detail

Plan Statistics | Activity

Plan Hash Value 495342630

Operation	Name	Estima...	Cost	Timeline(253s)	Exe...	Actu...	Memor...	Tem...	CPU Activity %	Wait Activity %
SELECT STATEMENT			251K		1	99				
COUNT STOPKEY					1	99				
VIEW		2406	251K		1	99				
SORT GROUP BY STOPKEY		2406	251K		1	99	6144			
HASH JOIN		65K	250K		1	3423K	3438K	662M	30	70
TABLE ACCESS FULL	LINEITEM	2400K	223K		1	48M			30	29
SORT AGGREGATE		1			1	1				
NESTED LOOPS OUTER		33	730		1	1776				
HASH JOIN RIGHT OUTER		31	699		1	1776	671K			
INDEX FULL SCAN	I_USER2	71	1		1	72				
NESTED LOOPS OUTER		31	698		1	1776				
NESTED LOOPS OUTER		31	671		1	1776				
HASH JOIN		31	668		1	1776	691K			
HASH JOIN		1323	448		1	1325	645K			
TABLE ACCESS FULL	USER$	71	3		1	72				
HASH JOIN		1323	445		1	1325	683K			
HASH JOIN		1323	237		1	1325	573K			
MERGE JOIN CART...		10	6		1	10				

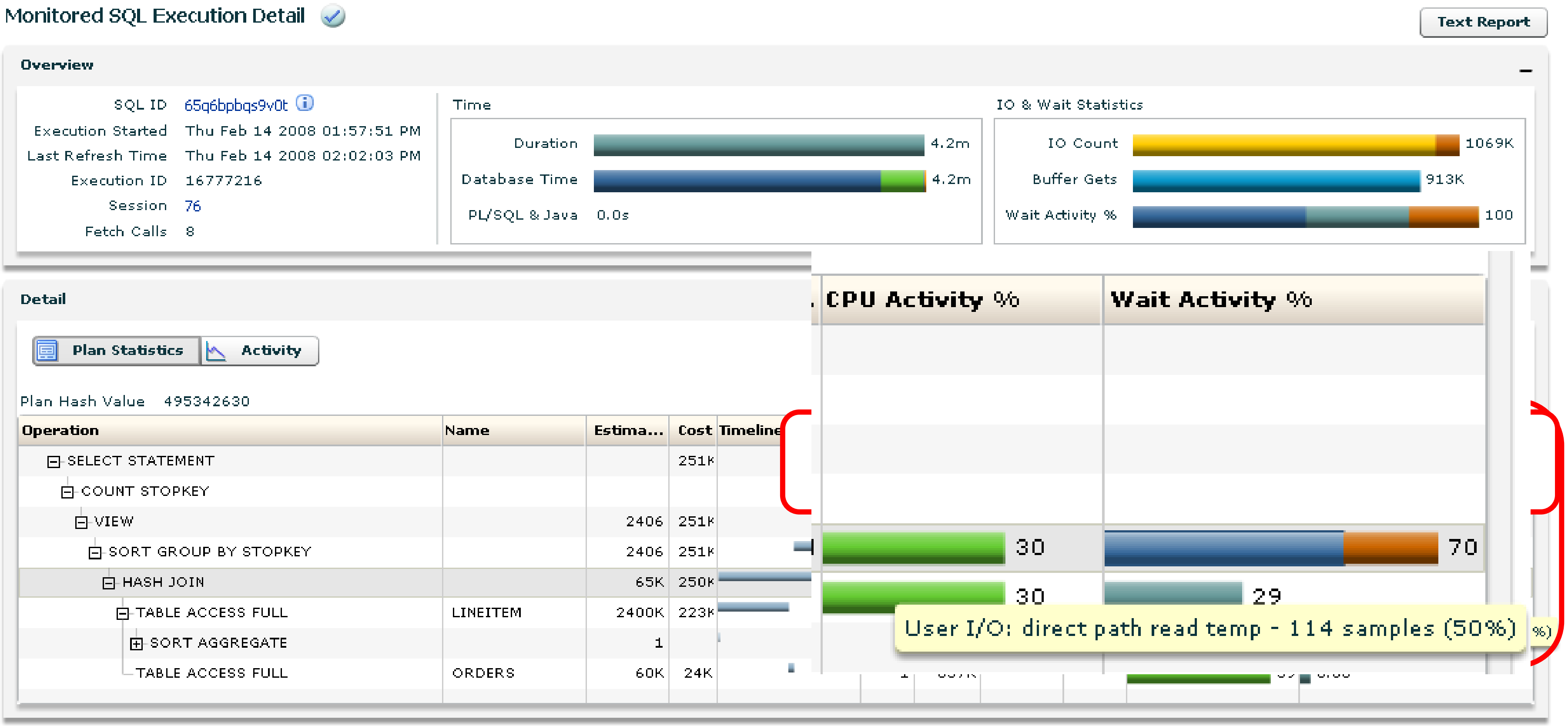
SQL Monitoring Use Case: Big Plans

SORT AGGREGATE		1		1	1	
NESTED LOOPS OUTER		33	730	1	1776	
HASH JOIN RIGHT OUTER		31	699	1	1776	671K
INDEX FULL SCAN	I_USER2	71	1	1	72	
NESTED LOOPS OUTER		31	698	1	1776	
NESTED LOOPS OUTER		31	671	1	1776	
HASH JOIN		31	668	1	1776	691K
HASH JOIN		1323	448	1	1325	645K
TABLE ACCESS FULL	USER$	71	3	1	72	
HASH JOIN		1323	445	1	1325	683K
HASH JOIN		1323	237	1	1325	573K
MERGE JOIN CART...		10	6	1	10	
HASH JOIN		1	1	1	1	198K
FIXED TABLE F...	X$KSPPI	1		1	1	
FIXED TABLE F...	X$KSPPCV	100		1	1944	
BUFFER SORT		10	6	1	10	2048
TABLE ACCESS ...	TS$	10	5	1	10	
TABLE ACCESS FULL	TAB$	1323	230	1	1325	
TABLE ACCESS FULL	OBJ$	61K	207	1	61K	
VIEW	DBA_OBJECTS	58K	219	1	60K	
UNION-ALL				1	60K	
FILTER				1	60K	
HASH JOIN		61K	214	1	61K	630K
TABLE ACCESS F...	USER$	71	3	1	72	
HASH JOIN		61K	210	1	61K	659K
INDEX FULL SCAN	I_USER2	71	1	1	72	
TABLE ACCESS ...	OBJ$	61K	208	1	61K	
TABLE ACCESS BY ...	IND$	1	2	2077	1535	
INDEX UNIQUE S...	I_IND1	1	1	2077	2077	
NESTED LOOPS		1	2			
INDEX FULL SCAN	I_USER2	1	1			
INDEX RANGE SCAN	I_OBJ4	1	1			
HASH JOIN		14	5	1	14	180K
INDEX FULL SCAN	I_LINK1	14	1	1	14	

SQL Monitoring Use Case: Big Plans

HASH JOIN		1323	448		1	1325	645K			
TABLE ACCESS FULL	USER$	71	3		1	72				
HASH JOIN		1323	445		1	1325	683K			
HASH JOIN		1323	237		1	1325	573K			
MERGE JOIN CART...		10	6		1	10				
HASH JOIN		1	1		1	1	198K			
FIXED TABLE F...	X$KSPPI	1			1	1				
FIXED TABLE F...	X$KSPPCV	100			1	1944				
BUFFER SORT		10	6		1	10	2048			
TABLE ACCESS ...	TS$	10	5		1	10				
TABLE ACCESS FULL	TAB$	1323	230		1	1325				
TABLE ACCESS FULL	OBJ$	61K	207		1	61K				
VIEW	DBA_OBJECTS	58K	219		1	60K				
UNION-ALL					1	60K				
FILTER					1	60K				
HASH JOIN		61K	214		1	61K	630K			
TABLE ACCESS F...	USER$	71	3		1	72				
HASH JOIN		61K	210		1	61K	659K			
INDEX FULL SCAN	I_USER2	71	1		1	72				
TABLE ACCESS ...	OBJ$	61K	208		1	61K				
TABLE ACCESS BY ...	IND$	1	2		2077	1535				
INDEX UNIQUE S...	I_IND1	1	1		2077	2077				
NESTED LOOPS		1	2							
INDEX FULL SCAN	I_USER2	1	1							
INDEX RANGE SCAN	I_OBJ4	1	1							
HASH JOIN		14	5		1	14	180K			
INDEX FULL SCAN	I_LINK1	14	1		1	14				
TABLE ACCESS FULL	USER$	71	3		1	72				
INDEX RANGE SCAN	I_OBJ1	1	1		1776	93				
INDEX RANGE SCAN	I_OBJ1	1	1		1776	1087				0.44
TABLE ACCESS CLUSTER	SEG$	1	1		1776	1107				
INDEX UNIQUE SCAN	I_FILE#_BLOCK#	1			1776	1107				
TABLE ACCESS FULL	ORDERS	60K	24K		1	857K			39	0.88

SQL Monitoring Use Case: Big Plans



Use SQL Tuning Advisor to Tune SQL

#6 BP

SQL

SQL Tuning Advisor

Comprehensive Analysis

- Statistics Analysis
- Access Structure Analysis (Single SQL)
- SQL Design Analysis
- Plan Tuning (SQL Profiling)

Recommendation

- Gather Missing or Stale Statistics
- Add Missing Indexes (B* Tree)
- Modify SQL Constructs
- Create a SQL Profile

Live vs. Remote Tuning

- Resource Consumption
  - Limited mode: Resource consumption minimal
    - Stats, index and SQL restructure analysis is cheap
    - Average is less than 1 second per SQL statement
  - Comprehensive mode: Resource consumption may be significant
    - SQL Profiling can potentially consume non-trivial resources
    - Roughly comparable to amount of resources/time consumed when executing SQL statement(s)
- Live tuning
  - Run SQL Tuning Advisor in Limited mode only if system does not have spare resources – otherwise run in Comprehensive mode (recommended)
- Remote tuning
  - Tuning remotely if
    - Cumulative resources/time consumed by all SQL statements significant
    - System cannot spare resources
  - Use SQL Profile and SQL Tuning Set export/import capabilities

More Best Practices when using SQL Tuning Advisor

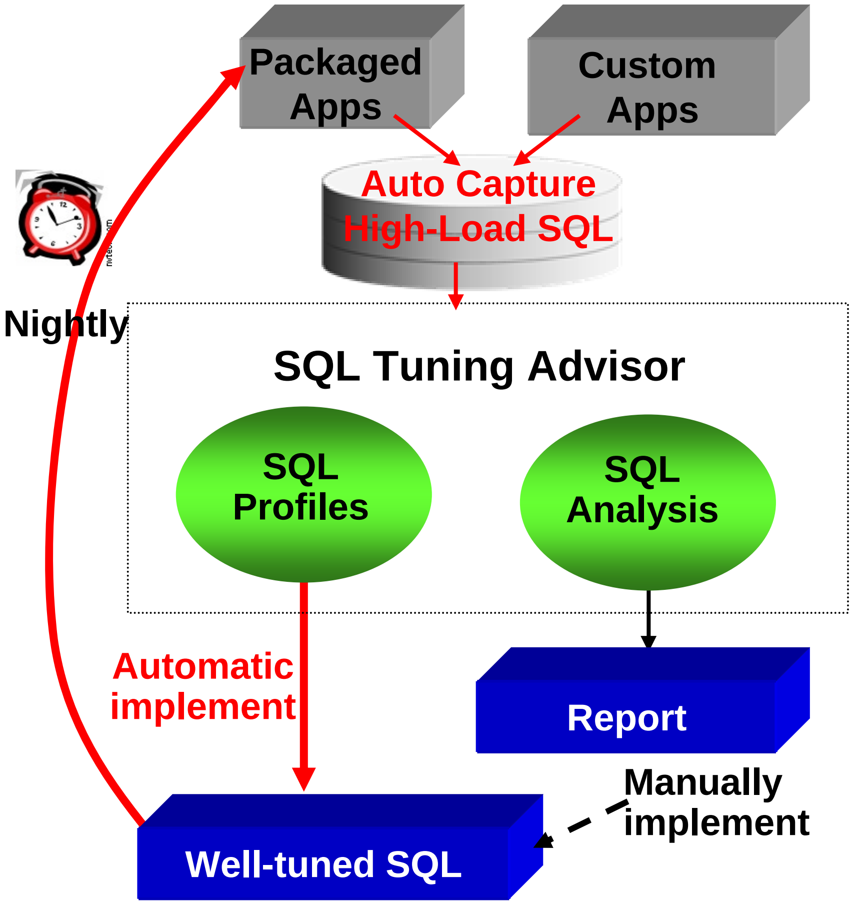
- Use Automatic SQL Capture feature of SQL Tuning Set (STS) to capture SQL Workload
- Always validate SQL Profiles before enabling them

```
DBMS_SQLTUNE.ACCEPT_SQL_PROFILE (task_name => ‘<tuning
task name>’, category => ‘MY_CATEGORY’);

ALTER SESSION SET SQLTUNE_CATEGORY=‘MY_CATEGORY’ ;
```

- For remote tuning, ensure test system is similar to production system
  - Schema
  - Data distribution
  - Volume
- If test system smaller than production, set optimizer stats manually

Automatic SQL Tuning



- Automatically captures high-load SQL
- Automatically tunes SQL without changing application by creating SQL Profiles
- Automatically validates SQL Profiles by test executing them
- Automatically implements (optional) greatly improved SQL plans
- Automatically reports analysis
- Automatically runs during maintenance window

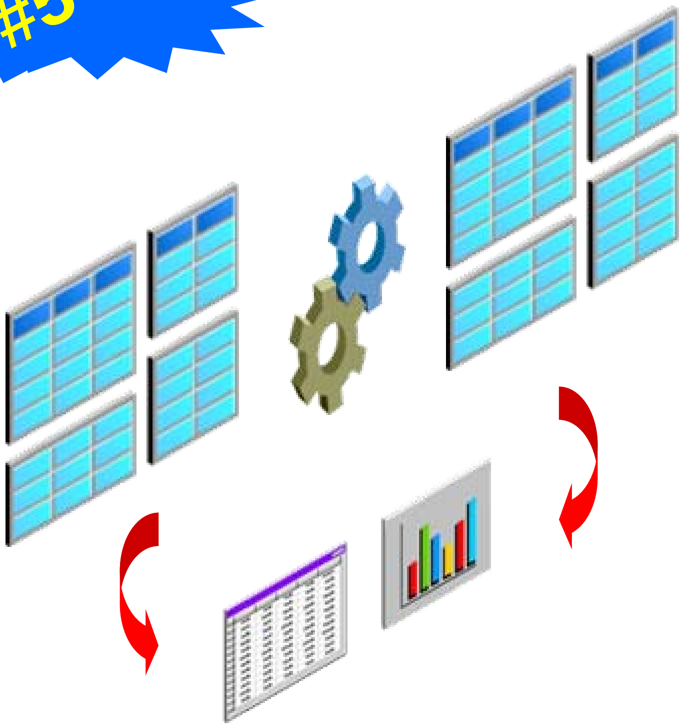
Optimizer Statistics Management

Use Automatic statistics collection to manage Optimizer Statistics

- Out-of-the box, runs in maintenance window
- configuration can be changed
- Restartable
- Gathers statistics on user and dictionary objects
- Parameters chosen automatically based on
  - DML monitoring
  - Column usage monitoring
  - Iterative sampling

ORACLE DATABASE 11g Uses new collection algorithm with accuracy of compute and speed faster than sampling of 1%

ORACLE DATABASE 11g Incrementally maintains statistics for partitioned tables – very efficient



More Best Practices on Statistics Collection

- Gather statistics for all objects (dictionary and user objects)
- Volatile objects
  - Gathers statistics when object at max size and then lock table
  - Delete all statistics and lock table – dynamic sampling will be used
- Restoring old optimizer statistics
  - Used when new stats result in poor execution plan selection
  - Scope: Table, schema and database
  - History maintained for 30 days
  - API: `DBMS_STATS.RESTORE_TABLE_STATS`
- Don't use the ANALYZE command
  - Officially obsolete for optimizer statistics
  - Cannot gather GLOBAL statistics for partitioned objects
  - Cannot gather statistics for external tables, fixed tables, etc.
  - Invalidates/recompiles all dependent cursors at once
  - `DBMS_STATS` marks cursors as unusable and recompiles gradually

Optimizer Statistics Validation

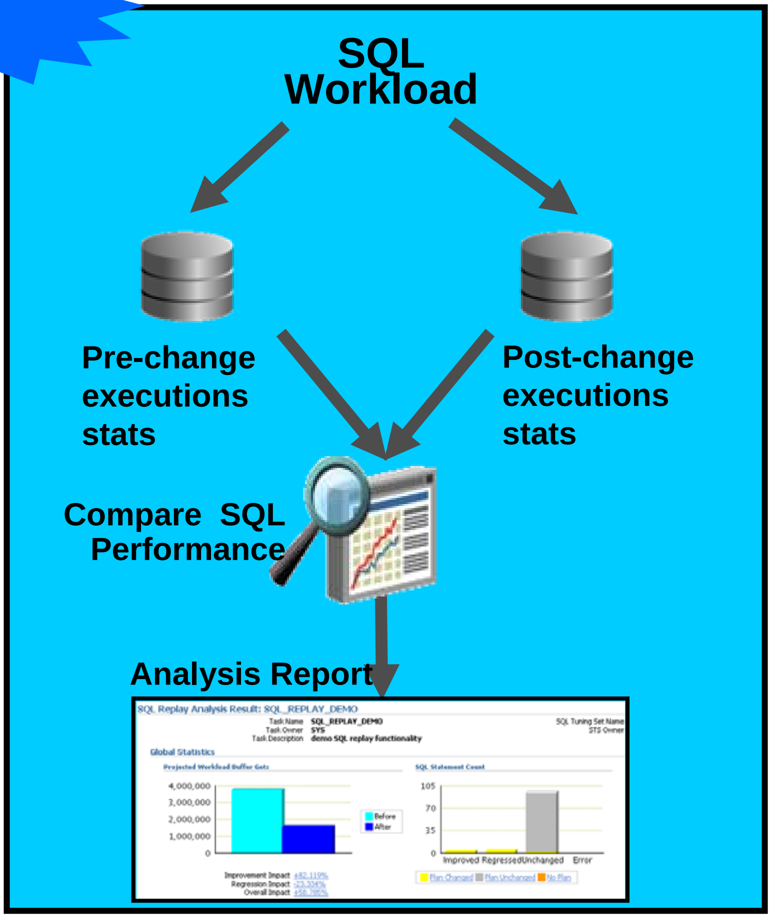
Use SQL Performance Analyzer (SPA) to validate statistics refresh

#4 BP

1. Capture SQL workload in STS using automatic cursor cache capture capability
2. Execute SPA pre-change trial
3. Refresh statistics using **PENDING** option
4. Execute SPA post-change trial
5. Run SPA report comparing SQL execution statistics

- Before **PUBLISH**ing stats
  - Remediate individual SQL for plan few regressions
  - Revert to old statistics if too many regressions observed

SQL Performance Analyzer (SPA)



Real Application Testing applicable for Pre-11g Database Releases

Feature	Capture From	Test Changes In
SQL Performance Analyzer	9*i* R2	10*g* R2 or 11*g*
	10*g* R1	10*g* R2 or 11*g*
	10*g* R2	10*g* R2 or 11*g*
Database Replay	9*i* R2	11*g*
	10*g* R2	11*g*

- **SQL Performance Analyzer (SPA)**
  - Capture on 9i, 10.1, 10.2 database releases
  - Test changes in 10.2 & above
- **Database Replay**
  - Capture on 9i, 10.2 database releases
  - Test changes in 11.1 & above

Space Management

Space Management Topics

- Permanent Tablespace Management
  - Extent management
  - Segment space management
- Temporary Tablespace Management
  - RAC

- **Goals**
  - Optimize space usage by eliminating/ minimizing fragmentation
  - Optimize data access and transaction performance

Permanent Tablespace Management

Use Locally Managed Tablespace for Extent Mgmt

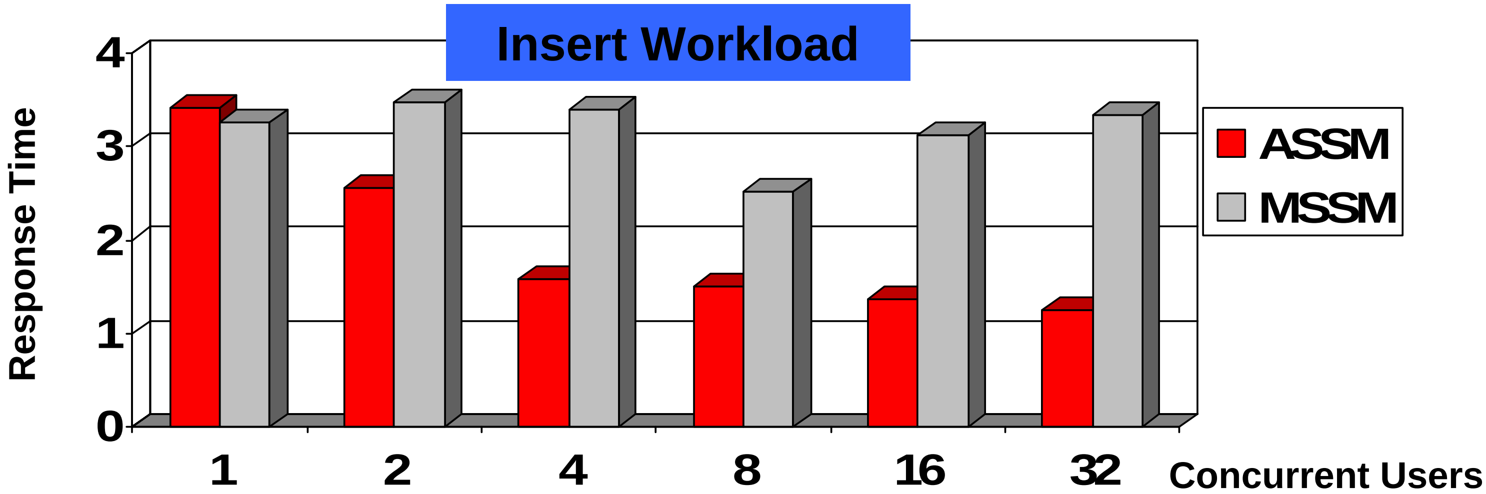
- Space managed locally by bitmaps in data file headers
- Eliminates external fragmentation
  - Efficient space utilization
- Performance
  - Serialization of space management at file level
  - Space management faster by 100-200%
- Two extent management types
  - **Auto-allocate (recommended):** Extent size determined by database
  - Uniform: All extents of same size

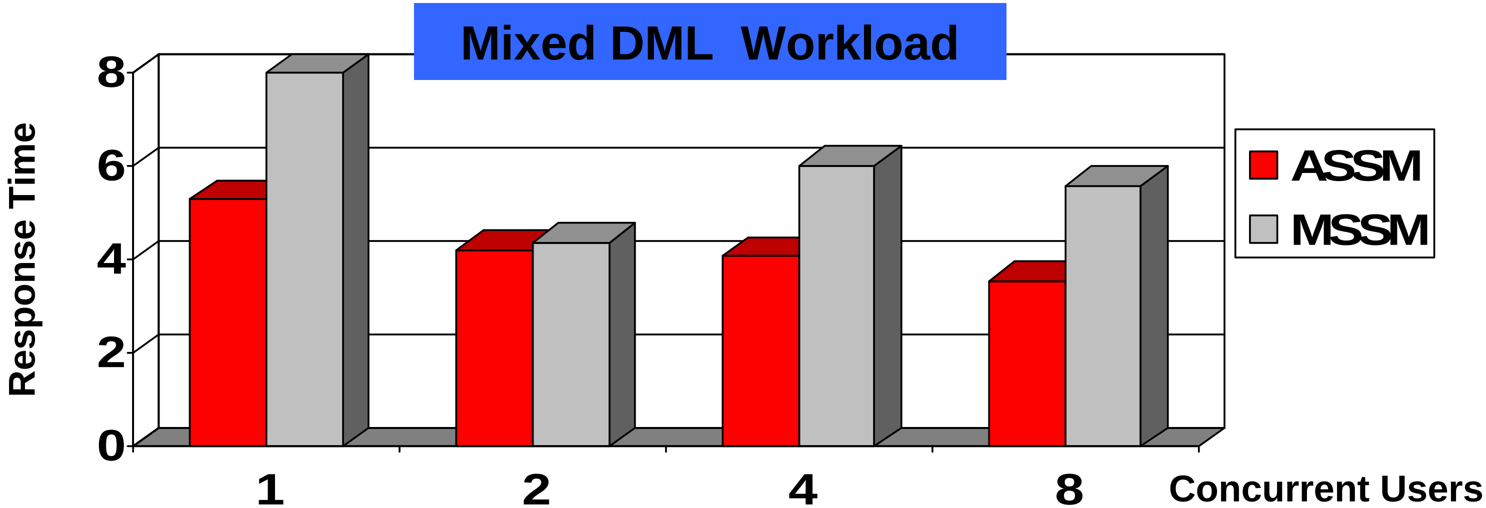
#3 BP

Use Automatic Segment Space Management for Segment Mgmt

- Segment free space managed using bitmaps
- Easier management: PCTUSED, FREELISTS, FREELIST GROUPS don’t have to be tuned or set
- Superior performance
  - Automatically manages contention on meta-data blocks
  - Inter-instance data block contention reduced by dynamic instance affinity
- Minimizes internal fragmentation

Automatic Segment Space Mgmt (ASSM) vs. Manual Segment Space Mgmt (MSSM)





Internal Fragmentation

- Fragmentation of space within a segment
  - Space under-utilization below HWM
  - Although minimized, can still occur in ASSM tablespace
- Performance Impact: slows certain access paths, e.g., full table scan
- Online Segment Shrink remedies internal fragmentation
  - ROW MOVEMENT must be ENABLED for heap organized segments
  - Segment must be in ASSM, locally managed tablespace
- Automatic Segment Advisor evaluates segments for fragmentation and makes appropriate recommendations

Space Operations	Shrink	Online Redef	Alter MOVE
Online	**Y**	**Y**	**N**
In-place	**Y**	**N**	**N**
Incremental	**Y**	**N**	**N**
Dependency Maintenance	**Y**	**N**	**N**
Segment Level Reorg	**Y**	**N**	**Y**
Parallel	**N**	**Y**	**Y**

Note: For tables with large number of indexes, reorg is faster

Temporary Tablespace Management

- Temporary Data
  - Data generated by operations like bitmap merges, hash join, bitmap index creation, sort
  - Persists only for duration of a transaction or session
  - Media and instance recovery is not required
  - High concurrency of space management operations is very critical
- **Use Temporary Tablespace for temporary data**
  - **ALTER DATABASE DEFAULT TEMPORARY TABLESPACE tablespace_name;**
  - **Use Locally Managed Temporary Tablespace**
  - Allows high concurrency space management
    - In steady state all space metadata cached in SGA
    - Operations serialized by SGA latch instead of db wide ST enqueue
  - Can be shrunk using SHRINK SPACE and/or SHRINK TEMPFILE commands

#2 BP

More Best Practices on Temporary Tablespace Management

- Guidelines for choosing extent size
  - 5M-10M:
    - For DSS, OLAP applications involving huge sorts, hash joins
    - Large temporary lobs are predominant
  - 64K or multiple:
    - Global temporary tables are predominant and amount of data loaded is small
    - Application is predominantly OLTP

- V$TEMPSEG_USAGE can be used to monitor space usage and workload distribution

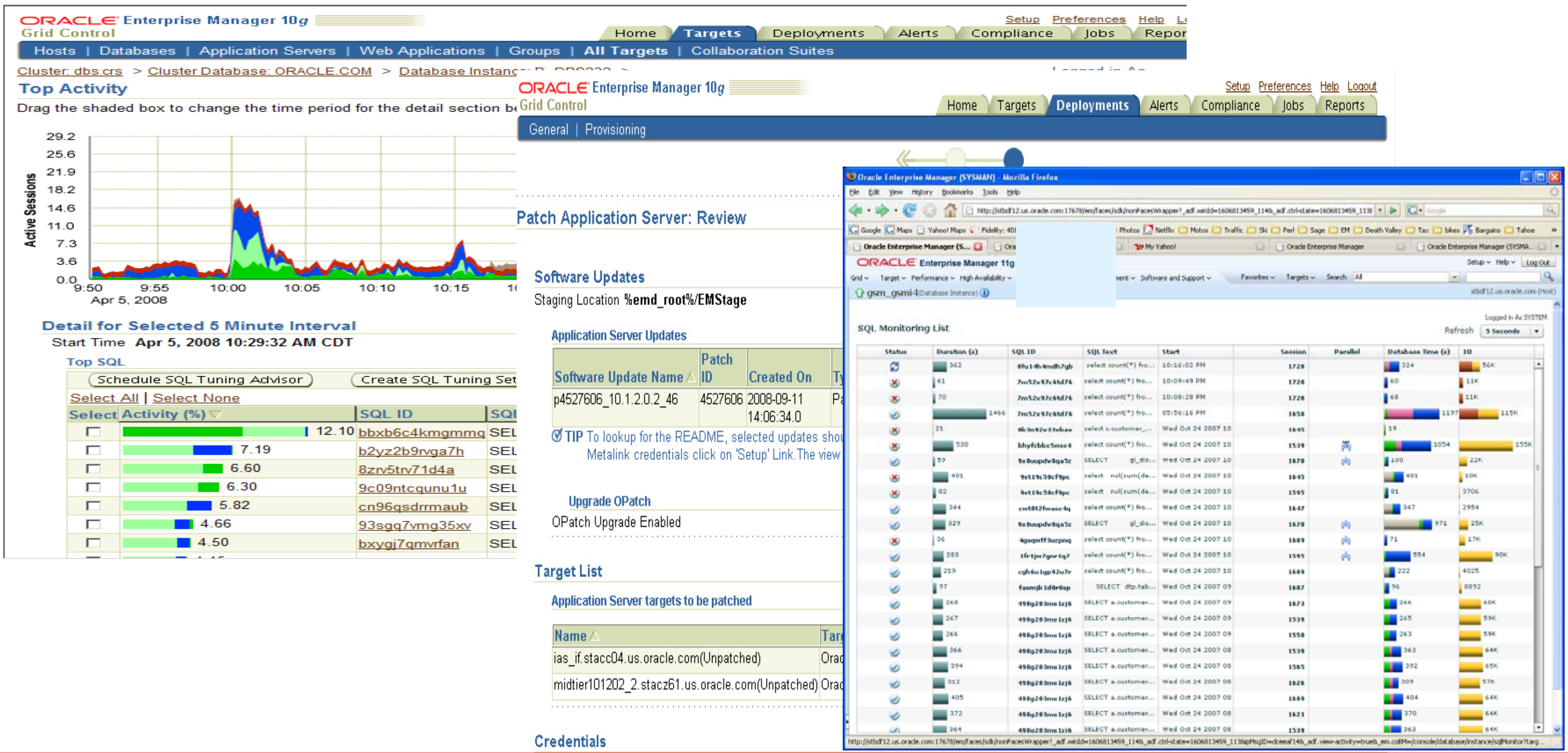
```
SESSION_NUM  USERNAME   SEGTYPE BLOCKS TABLESPACE
-----------  -------- --------- ------ ----------
        101    SCOTT       SORT    128       TEMP
        102    SCOTT   LOB_DATA    128       TEMP
        103      SYS       HASH    256       TEMP
```

Best Practices for Temporary Tablespace Management in RAC

- Use a single Temporary Tablespace for entire RAC database
- No special configuration is needed
- Each instance dynamically caches extents it has affinity to in its SGA
- Sharing of space between instances happens transparently and dynamically
  - Add space when number of waits on SS enqueue increases

Use Enterprise Manager to Manage Database

- Grid Control or Database Control (out-of-the-box)



Top 3 DBA Activities

Top 10 Best Practices for

#10 Use ADDM for database-wide performance diagnostic

#9 Use ASH for targeted performance analysis

#8 Use AWR Baseline for comparative performance analysis

#7 Use Real-time SQL Monitoring to understand SQL execution

#6 Use SQL Tuning Advisor to tune SQL

#5 Use Automatic statistics collection to manage optimizer statistics

#4 Use SQL Performance Analyzer (SPA) to validate statistics refresh

#3 Use Locally Managed TS with Auto-Allocate & Automatic Segment Space Management for Permanent Tablespace

#2 Use Locally Managed Temporary Tablespace

#1 Use Enterprise Manager to manage database

Q&A