Real Application Testing
Never Get Caught By Change Again

Michael R. Messina, Management Consultant

Rolta-TUSC, NYOUG 2011 (60 min)

Introduction

- Michael Messina
- Management Consultant with Rolta-TUSC
- Background includes Performance Tuning, High Availability and Disaster Recovery
- Using Oracle for approximately 17 years
- Oracle ACE
- Oracle OCP 9i/11g
- messinam@tusc.com
- www.tusc.com

Audience Experience

- How Many Have Used Real Application Testing
  - SQL Performance Analyzer
  - Database Replay
- Positive Experience
- Not so Positive Experience

Agenda

- Challenges of Change
- Real Application Testing Overview
- SQL Performance Analyzer
- Database Replay
- SQL Performance Analyzer Case
- Database Replay Case
- Conclusions

CHALLENGES OF CHANGE

Challenges of Change

- Database Upgrades
  - Optimizer Changes and Updates
  - New Features
- Database Parameter Changes
  - Optimizer adjustments
  - Using New Features
- Database Change
  - Move to RAC
  - Move From RAC to Single Instance

Challenges of Change

- Application Changes and Updates
  - Schema Changes and Updates
  - Application SQL Updates
- Infrastructure Changes
  - Storage
  - Servers
  - Platform Change
  - Solid State Disk

REAL APPLICATION TESTING OVERVIEW

Real Application Testing

- Nick Name RAT
- SQL Performance Analyzer
  - Get your SQL
  - Run Your SQL
- Database Replay
  - Get your actual Production Workload
  - Rerun Actual production workload
  - Run workloads from 9i and 10g on11g

Real Application Test for Version prior to 11g

Source DB	Replay Target	Patch Requirement
9.2.0.8	> 11.1.0.6	one off patch 6973309
10.2.0.2	> 11.1.0.6	one off patch 6870469
10.2.0.3	> 11.1.0.6	one off patch 6974999
10.2.0.4	> 11.1.0.6	Functionality Exists in 10.2.0.4 patchset

SQL PERFORMANCE ANALYZER

SQL Performance Analyzer

- Nick Name SPA
- Examine affects database and system changes have on SQL
- Integrated with SQL Tuning Set (STS)
- Integrated with SQL Tuning Advisor
- Integrated with SQL Plan Management
- Great with extremely large SQL workloads

SQL Performance Analyzer

- Impact of changes on SQL execution plans
- Impact of change on SQL execution statistics
- Compares the SQL execution result, before and after the change
- Report outlining the net benefit on the workload due to the changes
- Set of regressed SQL statements along with executions plan details and any recommendations

SQL Performance Analyzer

- Great for
  - Database Upgrades and Patches
  - Database Initialization Parameter Changes
  - Schema Changes
    - New Indexes
    - Remove Indexes
    - Partitioning
    - Compression
  - Cost Based Optimizer Statistic Changes
  - Implementation of Tuning Recommendations
  - OS Changes and upgrades
  - Hardware Changes

SQL Performance Analyzer

- Capture SQL into SQL Tuning Set (STS)
  - Cursor C ache
  - Automatic Workload Repository (AWR)
  - Existing SQL Tuning Set(s)
  - User Provided SQL
- Incremental SQL workload capture
  - Capture full system SQL workload
  - Repeat review cursor cache & update STS
  - Can focus on specified criteria such as user, service, action, module, etc.
  - overhead of incremental capture is < 1%

SQL Performance Analyzer

- Transfer SQL Tuning Set
  - Export SQL Tuning Set
  - Import SQL Tuning Set
  - Utilizes Data Pump
  - Use OEM Grid Control or Manually with API
- Allows capture of Production SQL Workload and then Test various Changes outside production
- System as Close to Production as Possible to ensure good impact measure

SQL Performance Analyzer

- Execute Baseline
  - After Import of SQL Tuning Set
  - Executes SQL Workload Prior to changes
  - Only query part of DML executed
  - Executes SQL sequentially and not necessarily in the same order they were captured
    - There is some control available to order such as longest response time first.
  - Can just Generate plans to reduce load, but provides lowest overall value.
  - Records information on execution

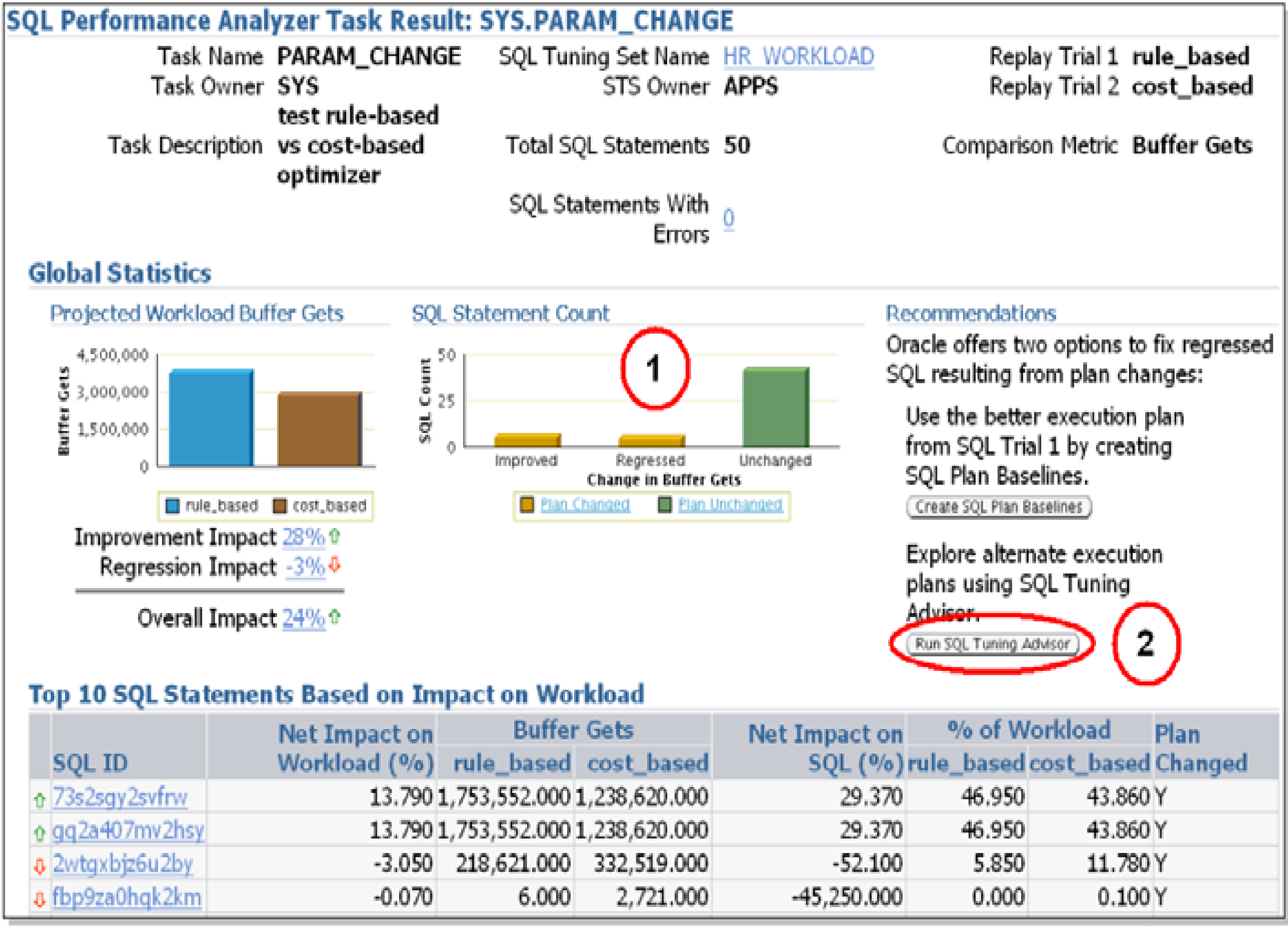
SQL Performance Analyzer

- Make Changes
  - – database upgrade,
  - – New index creation
  - – initialization parameter changes
  - – optimizer statistics refresh
  - – Etc.
- Re-execute STS
  - – Executes SQL Workload after change(s)
  - – Only query part of DML executed
  - – Records Post Change Performance

SQL Performance Analyzer

- Compare
  - Produces a report
  - Takes into account the number of executions of SQL statement for weight of each SQL
  - Uses elapsed time as the comparison metric by default
  - Alternative Comparison Metrics
    - Disk reads
    - CPU time
    - Buffer gets
    - Etc.

SQL Performance Analyzer



SQL Performance Analyzer Task Result: SYS.PARAM_CHANGE

Task Name	PARAM_CHANGE	SQL Tuning Set Name	HR_WORKLOAD	Replay Trial 1	rule_based
Task Owner	SYS	STS Owner	APPS	Replay Trial 2	cost_based
Task Description	test rule-based vs cost-based optimizer	Total SQL Statements	50	Comparison Metric	Buffer Gets
		SQL Statements With Errors	0		

Global Statistics

Projected Workload Buffer Gets

Improvement Impact 28% ⇧
Regression Impact -3% ⇩

Overall Impact 24% ⇧

SQL Statement Count

Recommendations

Oracle offers two options to fix regressed SQL resulting from plan changes:

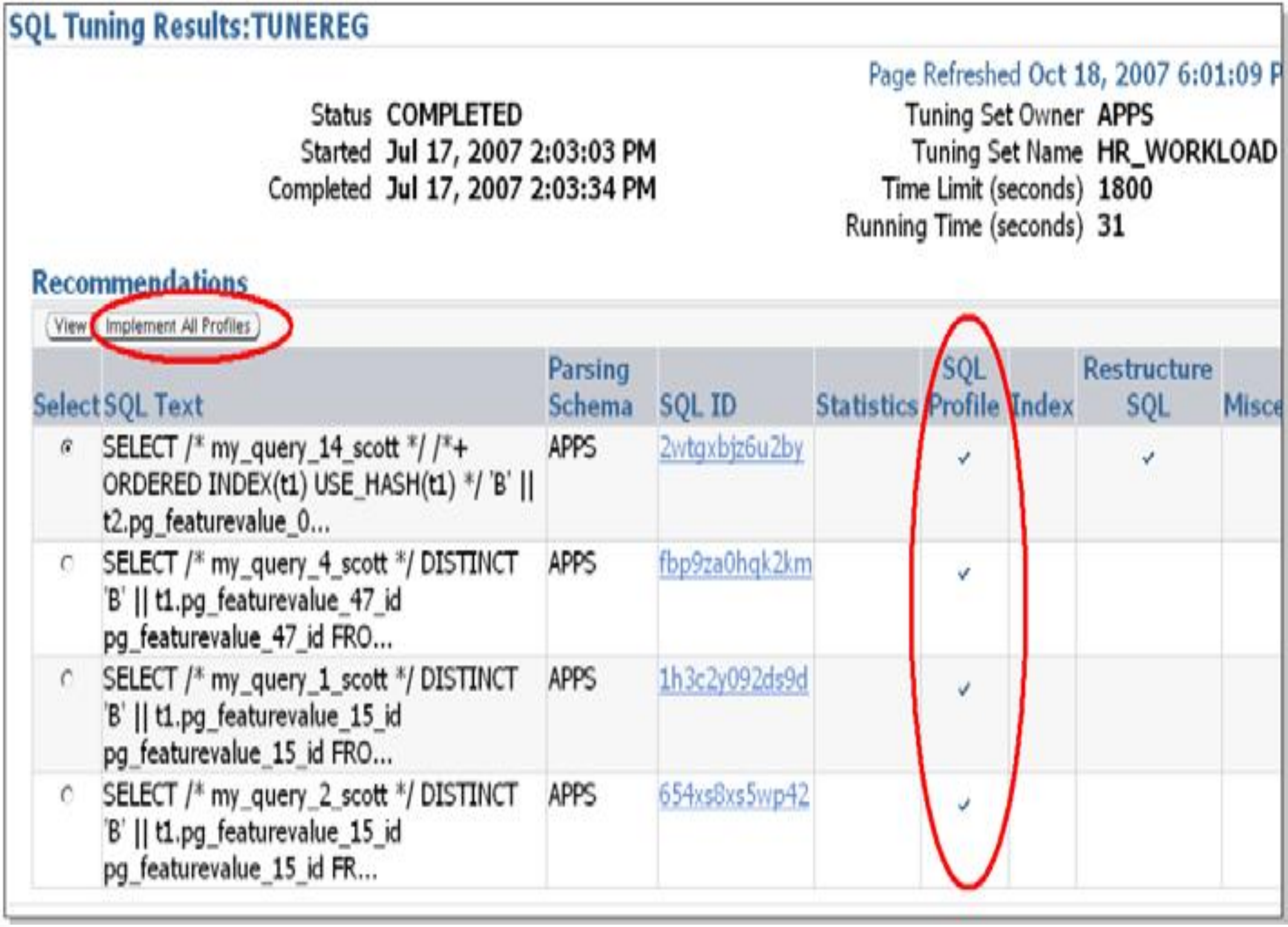
Use the better execution plan from SQL Trial 1 by creating SQL Plan Baselines.

Explore alternate execution plans using SQL Tuning Advisor.

Top 10 SQL Statements Based on Impact on Workload

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	% of Workload		Plan Changed
			rule_based	cost_based		rule_based	cost_based	
⇧	73s2sqy2svfrw	13.790	1,753,552.000	1,238,620.000	29.370	46.950	43.860	Y
⇧	gq2a407mv2hsy	13.790	1,753,552.000	1,238,620.000	29.370	46.950	43.860	Y
⇩	2wtgxbjz6u2by	-3.050	218,621.000	332,519.000	-52.100	5.850	11.780	Y
⇩	fbp9za0hqk2km	-0.070	6.000	2,721.000	-45,250.000	0.000	0.100	Y

SQL Performance Analyzer



SQL Tuning Results:TUNEREG

Page Refreshed Oct 18, 2007 6:01:09 P

Status	COMPLETED
Started	Jul 17, 2007 2:03:03 PM
Completed	Jul 17, 2007 2:03:34 PM

Tuning Set Owner	APPS
Tuning Set Name	HR_WORKLOAD
Time Limit (seconds)	1800
Running Time (seconds)	31

Recommendations

Select	SQL Text	Parsing Schema	SQL ID	Statistics	SQL Profile	Index	Restructure SQL	Misce
●	SELECT /* my_query_14_scott */ /*+ ORDERED INDEX(t1) USE_HASH(t1) */ 'B' || t2.pg_featurevalue_0...	APPS	2wtgxbjz6u2by		✓		✓	
○	SELECT /* my_query_4_scott */ DISTINCT 'B' || t1.pg_featurevalue_47_id pg_featurevalue_47_id FRO...	APPS	fbp9za0hqk2km		✓			
○	SELECT /* my_query_1_scott */ DISTINCT 'B' || t1.pg_featurevalue_15_id pg_featurevalue_15_id FRO...	APPS	1h3c2y092ds9d		✓			
○	SELECT /* my_query_2_scott */ DISTINCT 'B' || t1.pg_featurevalue_15_id pg_featurevalue_15_id FR...	APPS	654xs8xs5wp42		✓			

SQL Performance Analyzer

- Query SQL Tuning Sets

```
SELECT  name,
        created,
        statement_count
FROM dba_sqlset ;
```

- Query Active SQL Tuning Set References

```
SELECT  id,
        sqlset_owner,
        sqlset_name,
        description
 FROM DBA_SQLSET_REFERENCES ;
```

SQL Performance Analyzer

- Remove Active SQL Tuning Set
  - ** Must be remove prior to removing STS

```
DBMS_SQLTUNE.REMOVE_SQLSET_REFERENCE
  ('STS_SPA_1', 2) ;
```

- Delete SQL Tuning Set
  - `DBMS_SQLTUNE.DROP_SQLSET ('STS_SPA_1') ;`

DATABASE REPLAY

Database Replay

- Measure Impact of Changes Affecting the Database
  - Database Upgrade
  - Operating System Upgrade
  - Change Disk Storage
  - Change Database Operating System
  - Change Database Hardware Platform
  - Database Parameter Changes
- Measure Impact on Entire Database Using a Real Database Workload

Database Replay

- Eliminate Needs to create artificial workloads can use actual production workload.
- Can Eliminate long coordinated Testing projects to measure impact of database changes.
- Can Greatly Reduce time to measure impacts of changes.

Database Replay

- Get Copy of database Prior to start of Capture. This will be used as the start point for the replay database.
  - RMAN Backup is Perfect for this.
  - Same Start Point can be used for Multiple Replays of the Same Workload
  - Have used Flashback database to for multiple replay executes to get to common starting point.

Database Replay

- Capture
  - Processing to captures all database activity executed against a
  - Generate a Report on the Capture Processing.
- Prepare for Capture
  - Create OS directory for Capture Files
  - Create Database Directory pointing to OS directory for Capture Files.
  - Set any Capture Filters Needed
    - User
    - Service
    - Program

Database Replay

- Start Capture

```
DBMS_WORKLOAD_CAPTURE.START_CAPTURE(
     name => 'DB_TO_EXADATA'
   , dir => 'CAPTURE_DIR'
   , duration => NULL
   , default_action => 'INCLUDE'
   , auto_unrestrict => TRUE) ;
```

- Run Normal Database Activity
- Stop Capture

```
dbms_workload_capture.finish_capture();
```

Database Replay

- Processing and initializing of the Captured Workload
  - Done on the server/database where workload will be replayed
  - Remap client connections
  - Adjust speed in which workload will replay
  - Determine number of workload replay clients needed.
  - Filter any activity from Replay

Database Replay

- Restore database in new location/OS/etc.
- Prepare Workload
  - Create location for Replay where Replay database is located.
  - Create Directory in database that points to the Replay location.
  - Copy Capture Files to Directory
- Process Captured Workload

```
dbms_workload_replay.process_capture
(replay_dir) ;
```

Database Replay

- Initialize Replay

```
dbms_workload_replay.initialize_replay
(replay_name, replay_dir) ;
```

- Prepare Workload for Replay

```
dbms_workload_replay.prepare_replay(
    synchronization=>FALSE) ;
```

- Determine Replay Clients Needed
  - Goto the replay OS directory
    - wrc mode=calibrate

Database Replay

- Replays that capture workload on a copy of the database with various changes.
  - Different Database Version
  - Different Operating System
  - Different Server Architecture
  - Different Storage Architecture
- Utilizes workload Replay clients

Database Replay

- Replay Workload
  - Start the Number of Replay Clients Indicated by Calibrate

```
wrc system/passwrd@db
  CONNECTION_OVERRIDE=TRUE SERVER=DB
  replaydir=/data1/FS2/rat-dir
```

  - Start the Replay

```
dbms_workload_replay.start_replay ;
```

  - Generate Replay Report

SQL PERFORMANCE ANALYZER TEST CASE

INDEX CHANGE

SQL Performance Analyzer Case

- Create SQL Tuning Set

```
BEGIN
-- Create the sql set
  DBMS_SQLTUNE.CREATE_SQLSET(sqlset_name =>
  'STS_SPA_1');
-- Limit the sql in the set to Just on the
  ORDERS and ORDER_ITEMS
DBMS_SQLTUNE.CAPTURE_CURSOR_CACHE_SQLSET(
  sqlset_name => 'STS_SPA_1`,
  basic_filter=> 'UPPER(sql_text) LIKE || '''' ||
  '%ORDER%' || '''',
   time_limit  => 300,
   repeat_interval => 2    );
END;
/
```

SQL Performance Analyzer Case

- Create Task

```
dbms.sqlpa.create_analysis_task
   (sqlset_name => ‘STS_SPA_1‘,
    task_name => ‘my_spa_task‘,
    description => ‘test index changes‘);
```

- Execute Task Prior to Changes

```
dbms_sqlpa.execute_analysis_task
   (task_name => ‘my_spa_task‘,
    execution_type => ‘test execute‘,
    execution_name => ‘before_index_change‘);
```

SQL Performance Analyzer Case

- Make our Changes
  - Add Indexes
  - Gather Statistics on New Indexes
- Re-execute our Task after Changes

```
dbms_sqlpa.execute_analysis_task
(task_name => ‘my_spa_task‘,
 execution_type =>‘test execute‘,
 execution_name =>‘after_index_change‘);
```

SQL Performance Analyzer Case

- Compare/Analysis Task

```
dbms_sqlpa.execute_analysis_task
(task_name =>‘my_spa_task‘,
 execution_type =>‘compare performance‘,
 execution_name =>‘analysis_results‘,
 execution_params => dbms_advisor.arglist
  (‘execution_name1‘, ‘before_index_change’,
   ‘execution_name2‘, ‘after_index_change‘,
   ‘comparison_metric‘,‘buffer_gets‘));
```

SQL Performance Analyzer Case

- Generate Analysis Report

```
SPOOL SPA_COMPARE_ANALYSIS_REPORT.out

SELECT DBMS_SQLPA.REPORT_ANALYSIS_TASK
 ('my_spa_task')
from dual;

SPOOL off
```

SQL Performance Analyzer Case

- Generate Summary Report

```
SPOOL SPA_COMPARE_SUMMARY_REPORT.out

SELECT DBMS_SQLPA.REPORT_ANALYSIS_TASK
('my_spa_task',
 'TEXT',
 'TYPICAL',
 'SUMMARY')
FROM DUAL;

SPOOL off
```

SQL Performance Analyzer Case

- Generate Findings Report

```
SPOOL SPA_COMPARE_FINDINGS_REPORT.out
SELECT DBMS_SQLPA.REPORT_ANALYSIS_TASK
 ('my_spa_task',
  'TEXT',
  'TYPICAL',
  'FINDINGS',
  5)
from dual;

SPOOL off
```

DATABASE REPLAY TEST CASE

MOVE TO EXADATA FROM 3 NODE WINDOWS RAC CLUSTER

Database Replay Case

- Backup of Windows Database
- Capture Production Windows Database Workload
  - Filtered Out OEM Activity

```
DBMS_WORKLOAD_CAPTURE.ADD_FILTER(
fname => 'ORACLE MANAGEMENT AGENT (DEFAULT)'
,fattribute => 'PROGRAM'
,fvalue => 'emagent%');

DBMS_WORKLOAD_CAPTURE.ADD_FILTER(
fname => 'ORACLE MANAGEMENT SERVICE
(DEFAULT)‘
,fattribute => 'PROGRAM'
,fvalue => 'OMS');
```

Database Replay Test Case

- Captured Workload

```
DBMS_WORKLOAD_CAPTURE.START_CAPTURE(
name => v_capture_name
,dir => v_capture_dir
,duration => NULL
,default_action => 'INCLUDE'
,auto_unrestrict => TRUE
```

- Copied Workload Capture Files to Exadata database server

Database Replay Test Case

- Restored Windows RAC Database to Exadata Linux RAC Database
- Process Captured Workload

```
dbms_workload_replay.process_capture
(v_replay_dir) ;
```

- Initialize replay

```
 dbms_workload_replay.initialize_replay
(replay_name, replay_dir) ;
```

Database Replay Test Case

- Prepare replay

```
dbms_workload_replay.prepare_replay(THINK_TIME_SCALE=>0,synchronization=> FALSE);
```

- Calibrate the workload

```
wrc mode=calibrate
```

Database Replay Test Case

```
Workload Replay Client: Release 11.2.0.1.0 - Production on Tue Nov 9 19:35:48 2010

Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.

Report for Workload in: .
-----------------------
Recommendation:
Consider using at least 14 clients divided among 4 CPU(s)
You will need at least 153 MB of memory per client process.
If your machine(s) cannot match that number, consider using more clients.

Workload Characteristics:
- max concurrency: 568 sessions
- total number of sessions: 5762

Assumptions:
- 1 client process per 50 concurrent sessions
- 4 client process per CPU
- 256 KB of memory cache per concurrent session
- think time scale = 100
- connect time scale = 100
- synchronization = TRUE
```

Database Replay Test Case

- Started 14 workload replay clients

```
wrc system/password@prdrmed
  CONNECTION_OVERRIDE=TRUE SERVER=PRDRMED
  replaydir=/data1/FS2/rat-dir
```

- Started Replay

```
dbms_workload_replay.start_replay ;
```

- Monitored Replay

```
select id, name,
 to_char(start_time,'mm/dd/yyyy hh24:mi:ss'),
 to_char(end_time,'mm/dd/yyyy hh24:mi:ss'),
 num_clients,think_time_scale, ELAPSED_TIME_DIFF
from dba_workload_replays ;
```

Database Replay Test Case

- Generated Replay Report

Replay Information from Report

Information	Replay	Capture
Name	PRDRMED_REPLAY_1	PRDRMED_CAPTURE_1
Status	COMPLETED	COMPLETED
Database Name	PRDRMED	PRDRMED
Database Version	11.2.0.1.0	10.2.0.4.0
Start Time	11-11-10 12:45:03	11-11-10 09:01:53
End Time	11-11-10 13:07:53	11-11-10 09:31:43
Duration	22 minutes 50 seconds	29 minutes 50 seconds
Directory Object	RAT_DIR	RAT_DIR
Directory Path	/data1/FS2/rat-dir	/data1/FS2/rat-dir

Questions/Discussion

THANK YOU