CON8247
DBA's New Best Friend for Mistake-Free Administration: Oracle Real Application Testing

Björn Bolltoft
Principal Product Manager
Database Manageability Real Application Testing

Kevin, Callanan
Senior Oracle DBA and Team Manager, Allied Irish Banks

Oct 01, 2014

September 28–
October 2, 2014
San Francisco

ORACLE OPEN WORLD

ORACLE

Safe Harbor Statement

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.

ORACLE®

Program Agenda

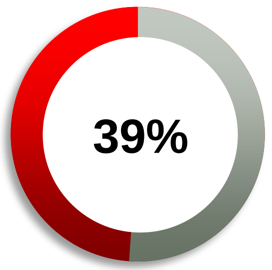
1. Real Application Testing
2. Use case: Statistics Refresh
3. Use case: Schema Optimization
4. Use case: Validate In-Memory with Query Only Replay

Top Challenges

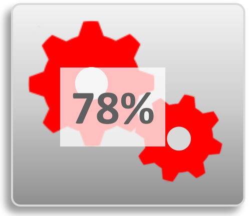
Database Management

IOUG
independent oracle users group
For the Complete Technology & Database Professional

Key Takeaway:
Improve & Ensure Higher Quality of Service



Handle more than 50 DBs each



Downtime resulting from untested changes

ORACLE

Source: From Database Clouds to Big Data: 2013 IOUG Survey on Database Manageability

Production System Changes

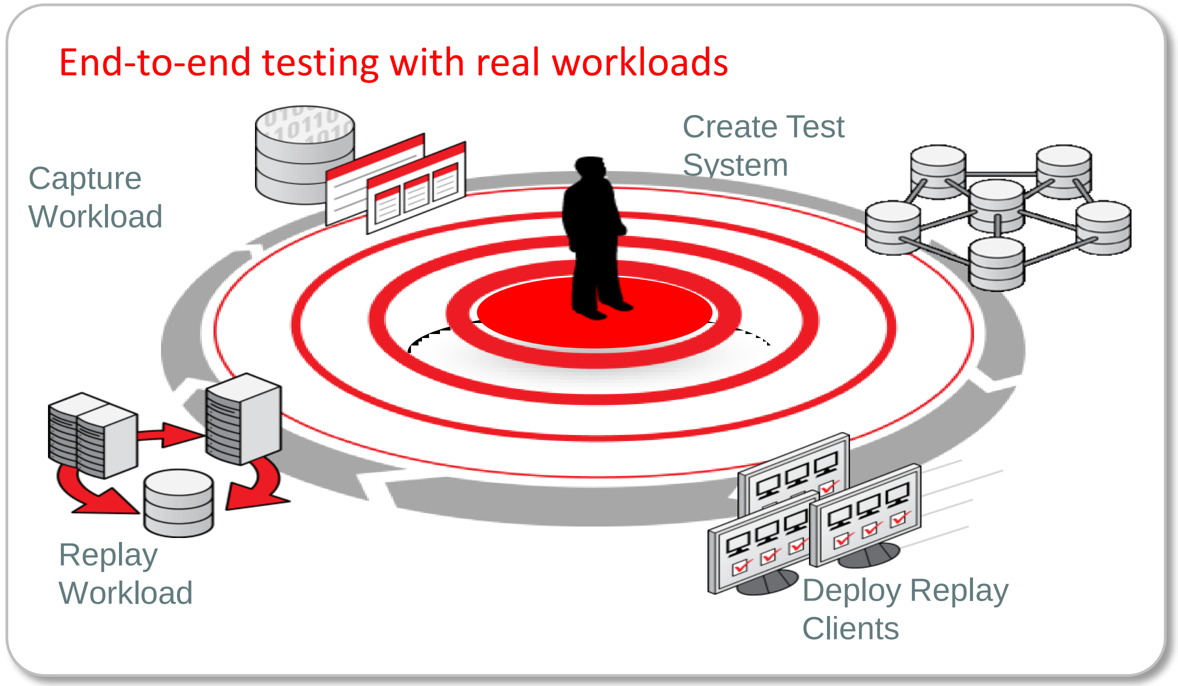
- Operational changes
  - Implement SQL Profiles
  - Refresh statistics on table, schema or database level
  - Change optimizer related init.ora parameters like OPTIMIZER_MODE...
  - Change memory related init.ora like PGA_AGGREGATE_TARGET...
- Non-operational changes
  - Adding or dropping indexes, table partitioning...
  - New features like Compression, In-Memory...
  - Infra structure changes like server, storage, interconnect...
  - Consolidation
  - Upgrades and patching 11g -> 12c, 12.1.0.1 -> 12.1.0.2, PSU 2...

Program Agenda

1. Real Application Testing
2. Use case: Statistics Refresh
3. Use Case: Schema Optimization
4. Use case: Validate In-memory with Query Only Replay

ORACLE

Real Application Testing Features



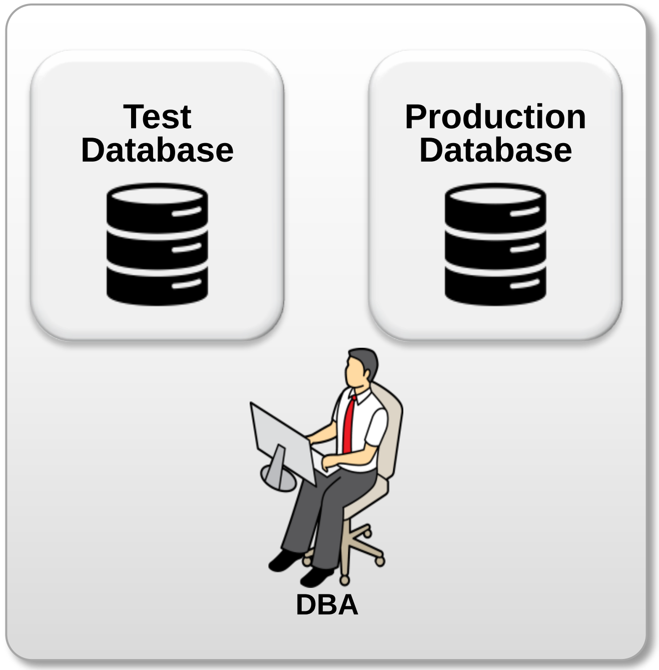
- SQL Performance Analyzer
  - SQL unit testing for response time
  - Identify and tune regressed SQL
  - Use SPA first
- Database Replay
  - Load, performance testing for throughput
  - Remediate application concurrency problems

SPA Challenges

Running SPA on:

Test System: Safe but…

- Requires separate HW
- Data in test system should be same as production
- Lengthy, error-prone task



Production System: Easier but…

- Could be resource intensive and impact production performance
- Changes needs to be manually scoped to private session
- Could take a long time to finish
- No resource control by default

SPA Quick Check

Optimized

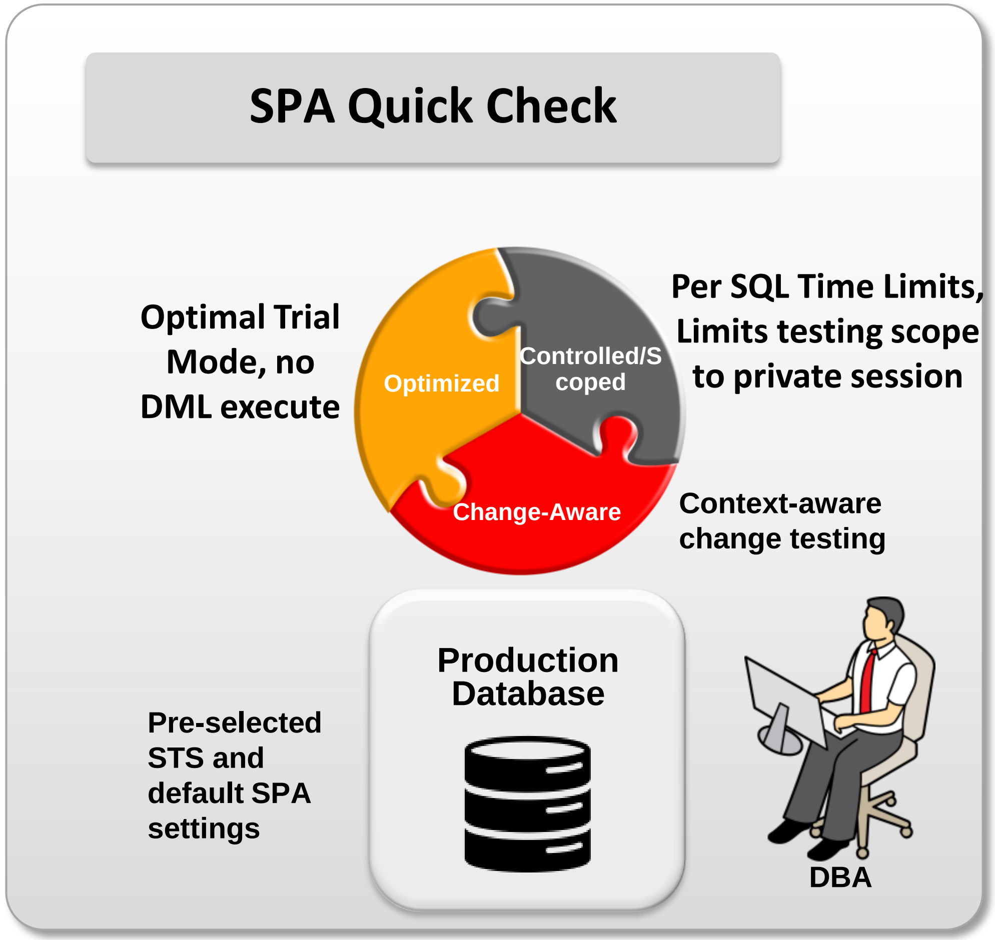
- Optimized for use on prod systems
- Optimal Trial or Explain Plan mode
- Disable multi-executions, full DML execute disabled

Controlled

- Per SQL time limits
- Testing scoped to private session
- Associate with Resource Consumer Group

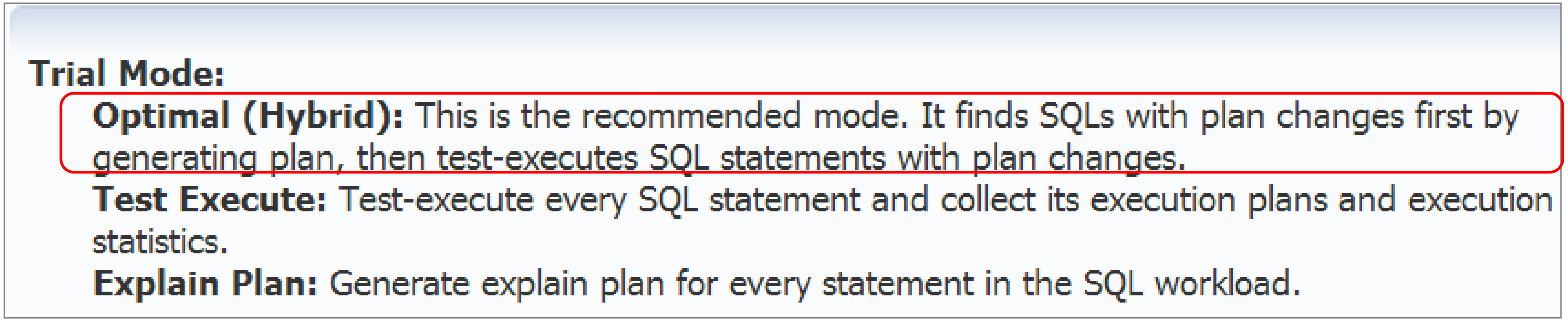
Change-Aware

- Context-aware change testing workflows, such as,
  - Optimizer gather statistics
  - Init.ora parameter changes


SPA Quick Check
Optimal Trial Mode, no DML execute
Optimized
Controlled/Scoped
Per SQL Time Limits, Limits testing scope to private session
Change-Aware
Context-aware change testing
Production Database
Pre-selected STS and default SPA settings
DBA

SPA Quick Check

Optimized



Trial Mode:

Optimal (Hybrid): This is the recommended mode. It finds SQLs with plan changes first by generating plan, then test-executes SQL statements with plan changes.

Test Execute: Test-execute every SQL statement and collect its execution plans and execution statistics.

Explain Plan: Generate explain plan for every statement in the SQL workload.

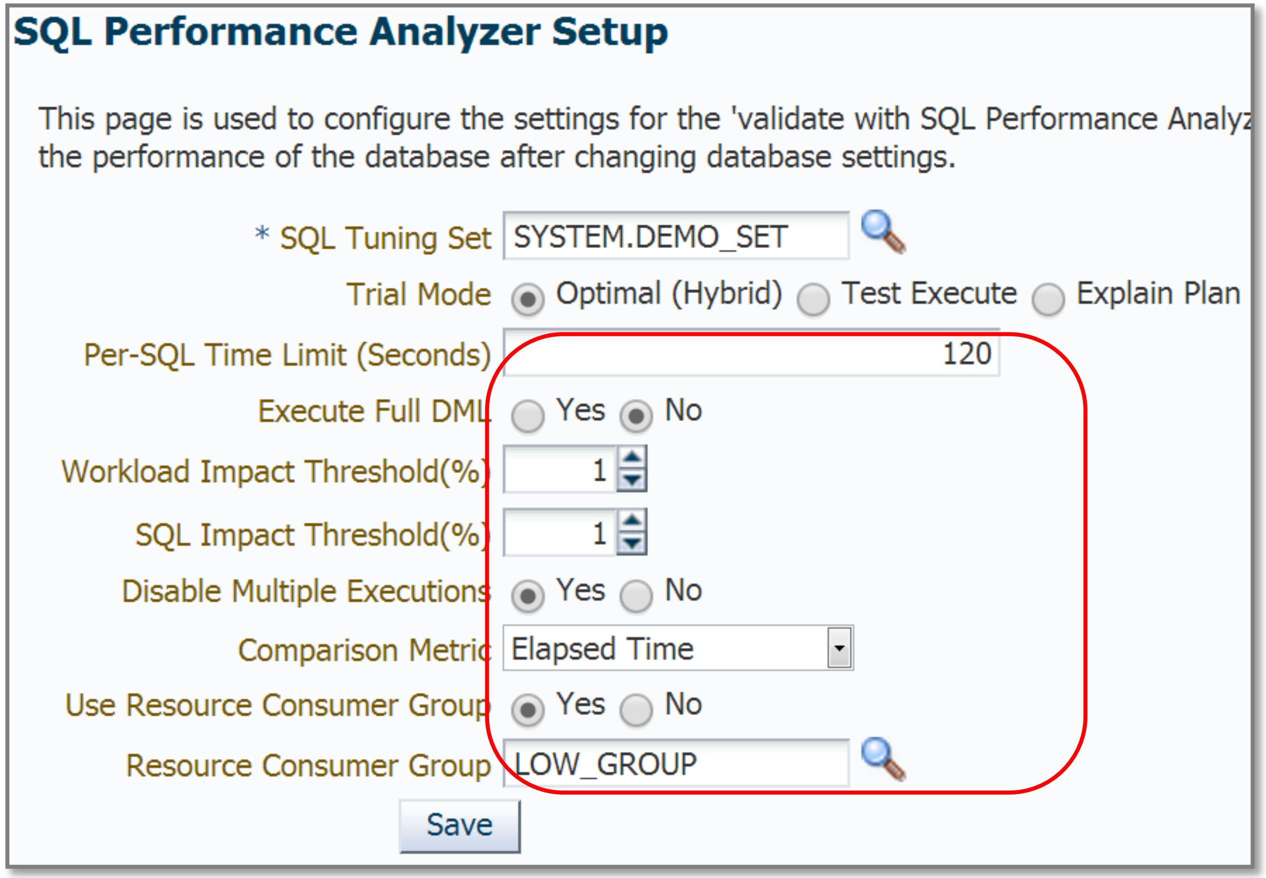
Identifies subset SQL workload with plan changes first	Test-executes only SQLs with plan changes	Minimizes use of production resources dramatically – up to 10x reduction	Multiple executions disabled	No full DML (execute Select part of workload)

SPA Quick Check
Controlled

Per-SQL time limit – protects from runaway SQL

Resource throttling - Associate with Resource Consumer Group

Testing scope limited to private session



SQL Performance Analyzer Setup

This page is used to configure the settings for the 'validate with SQL Performance Analyz
the performance of the database after changing database settings.

* SQL Tuning Set	SYSTEM.DEMO_SET
Trial Mode	[✓ Optimal (Hybrid)] Test Execute Explain Plan
Per-SQL Time Limit (Seconds)	120
Execute Full DML	Yes [✓ No]
Workload Impact Threshold(%)	1
SQL Impact Threshold(%)	1
Disable Multiple Executions	[✓ Yes] No
Comparison Metric	Elapsed Time
Use Resource Consumer Group	[✓ Yes] No
Resource Consumer Group	LOW_GROUP

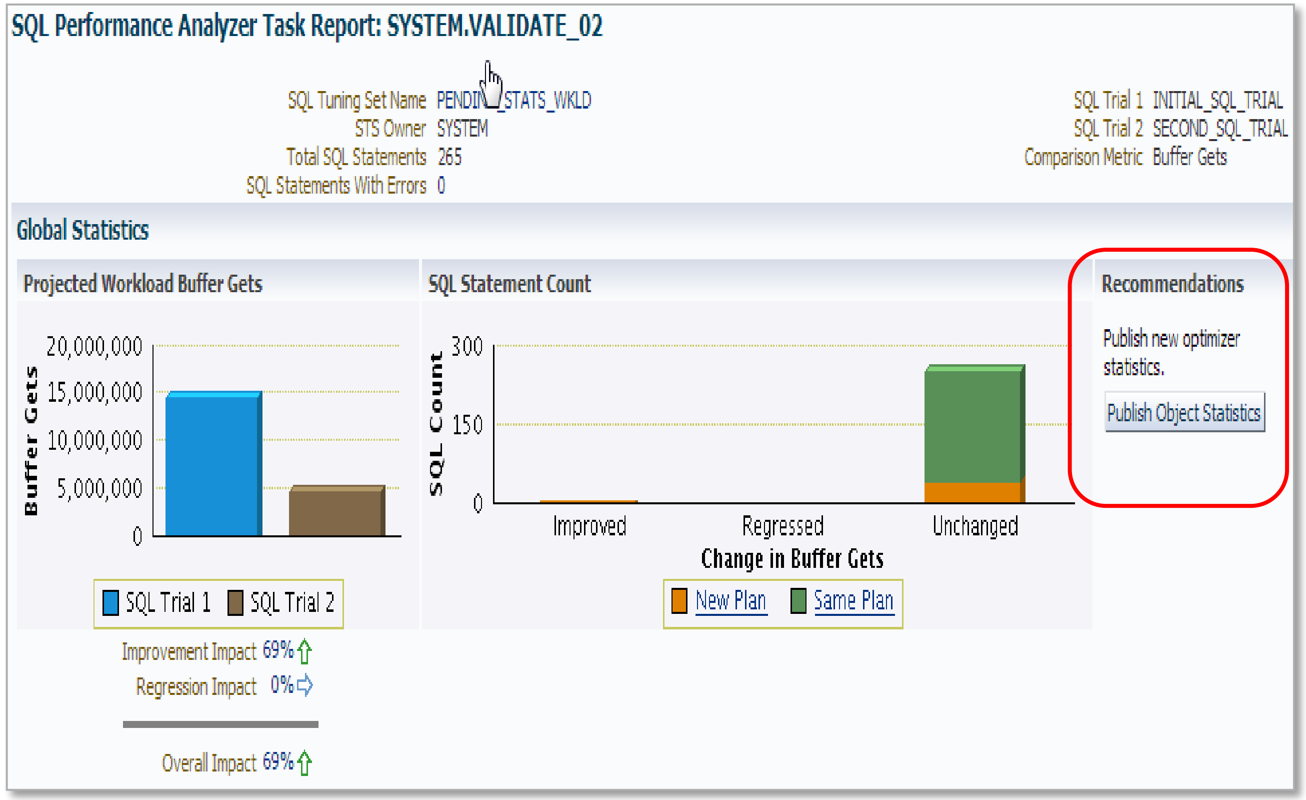
SPA Quick Check

Change-aware

Change-aware: Knows what change is being tested

In-line with routine DBA tasks such as statistics gathering, init.ora parameter changes

Intelligently limits impact to private test session



SQL Performance Analyzer Task Report: SYSTEM.VALIDATE_02

SQL Tuning Set Name	PENDIN_STATS_WKLD	SQL Trial 1	INITIAL_SQL_TRIAL
STS Owner	SYSTEM	SQL Trial 2	SECOND_SQL_TRIAL
Total SQL Statements	265	Comparison Metric	Buffer Gets
SQL Statements With Errors	0		

Global Statistics

Improvement Impact 69%
Regression Impact 0%
Overall Impact 69%

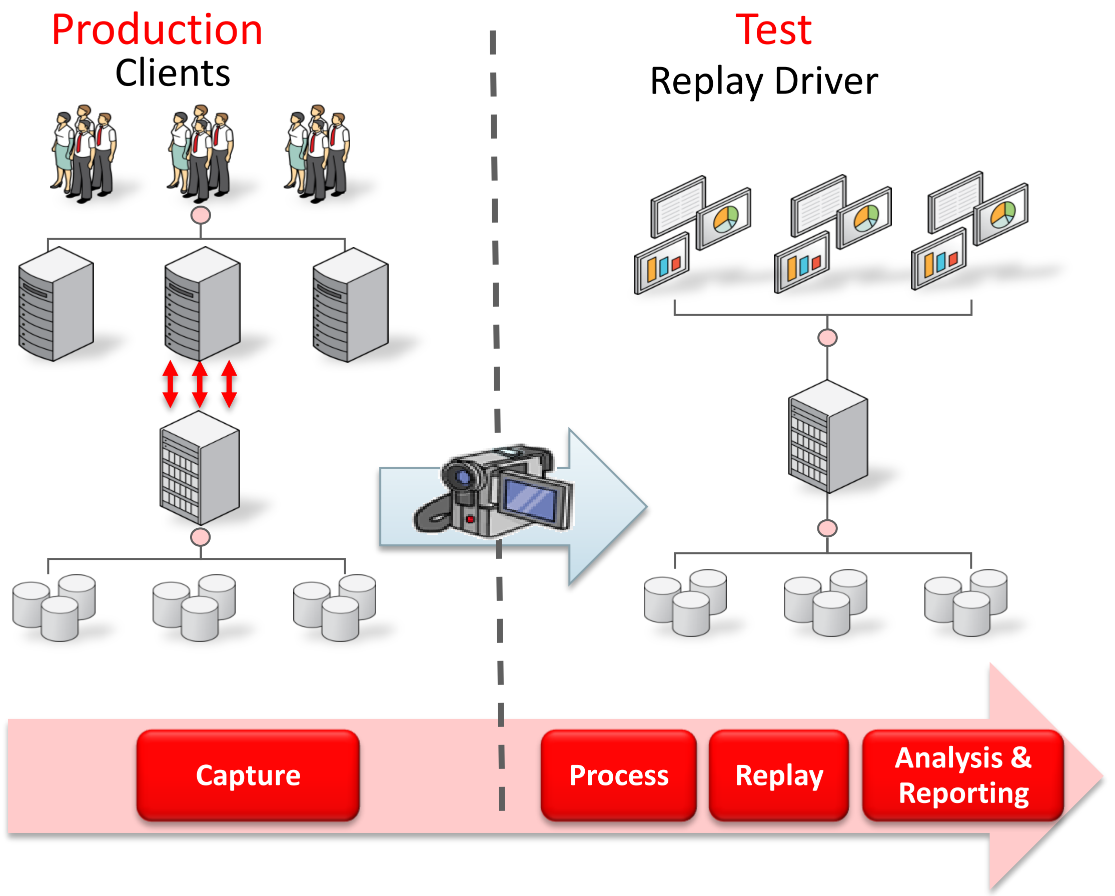
Recommendations

Publish new optimizer statistics.

Publish Object Statistics

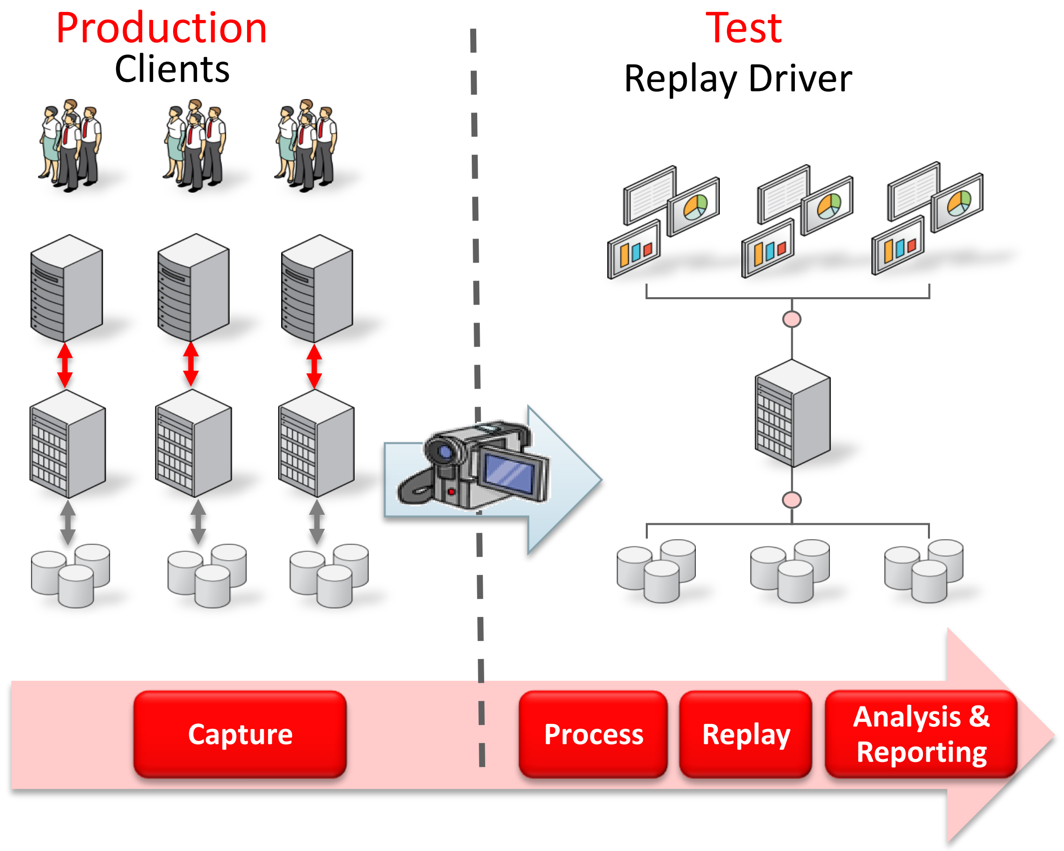
Database Replay

- Database load and performance testing with real production workloads
  - Production workload characteristics such as timing, transaction dependency, think time, etc., fully maintained
- Test and measure transaction throughput improvements
- Identify application scalability and concurrency problems
- Use for server and OS consolidation
  - Capture individual workloads
  - Replay workloads concurrently



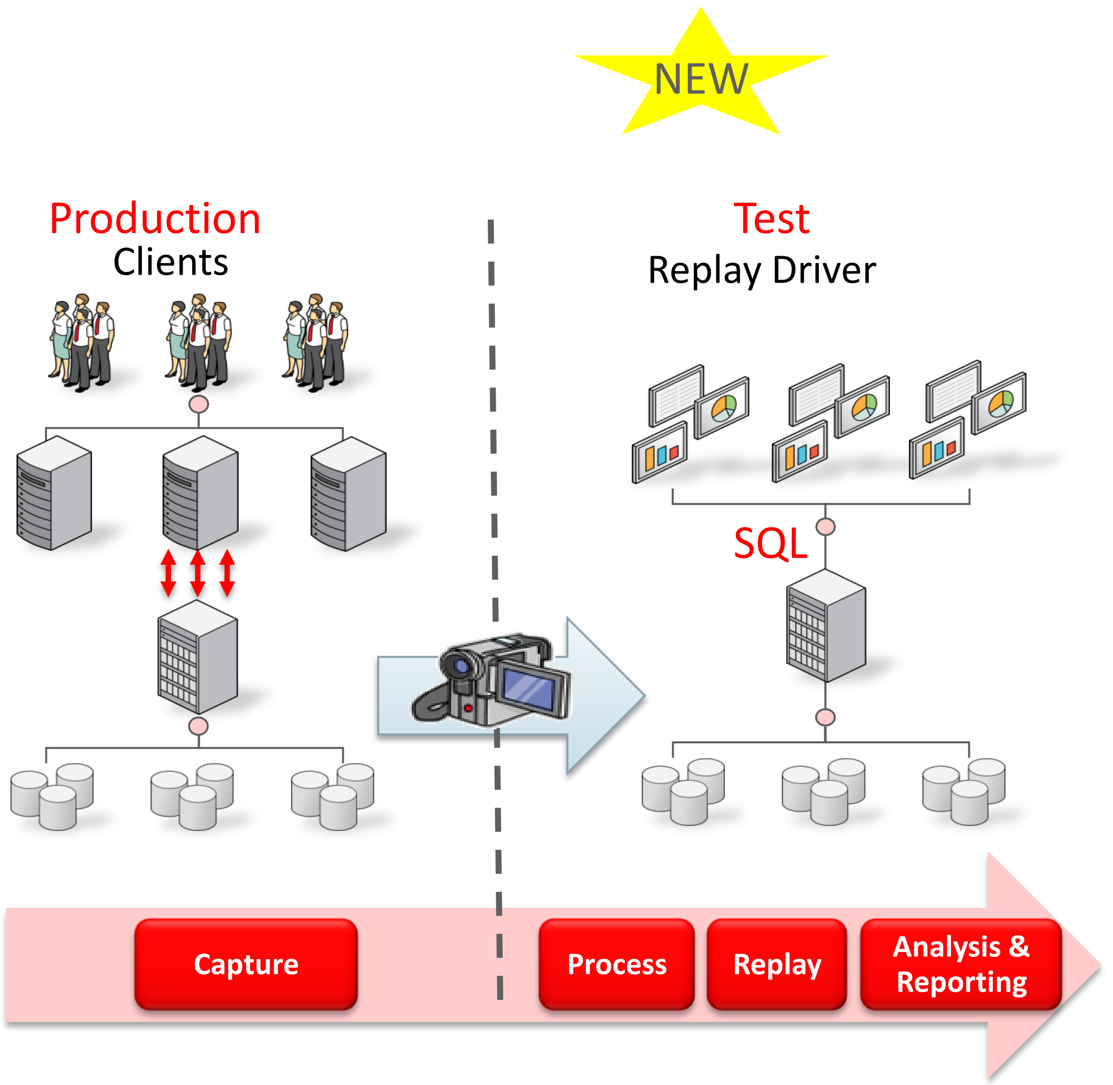
Consolidated Database Replay

- Workload captured on different databases (including different supported platforms, versions) can be replayed concurrently
- Works for schema consolidation and Pluggable Databases
- Identify and remediate inter-application scalability and concurrency problems
- Allows scale-up, subsetting, scheduling of multiple workloads
- Available for 11.2.0.2 and above, MOS Note: 1453789.1



Query only Replay

- Database load and performance testing with production workloads
  - Production workload characteristics such as timing, think time, etc., fully maintained
  - No DML
- Test and measure SQL throughput improvements
- Identify application scalability and concurrency problems
- Use for server and OS consolidation
  - Capture individual workloads
- Database state unchanged after Query only replay
  - No database restore required
- No limitation
  - Can be used for any application
- Allows scale-up, subsetting, scheduling of multiple workloads
- Optimal state of the database: Post capture



Which feature to use for a given change?

Change	Description	SPA	Query Only Replay	Database Replay	Concurrent Replay
SQL Profiles	Implement SQL profiles	✓	✓	✗	✗
Schema Tuning	Adding or droping indexes, Partitioning...	✓	✓	✓	✗
Optimizer Statistics	Refresh statistics on Table,schema or database level	✓	✓	✗	✗
Init.ora Optimizer	DB_FILE_MULTIBLOCK_READ_COUNT, OPTIMIZER_MODE...	✓	✓	✗	✗
Init.ora Memory	SGA_MAX_SIZE, PGA_AGGREGATE_TARGET (Concurrency related)	✗	✓	✓	✗
Features/Options	Compression, In-Memory...	✓	✓	✓	✗
Infra structure	Server, storage, Interconnect...	✓	✓	✓	✗
Upgrades	11g -> 12c, 12.1.0.1 -> 12.1.0.2...	✓	✓	✓	✗
Consolidation	Server Consolidation, Multitenant...	✓	✓	✓	✓
Capacity planning	Server Consolidation, Increasing user activity...	✓	✓	✓	✓
Reactive SQL Performance regression analysis	Find changes in plans and workloads between different days by using Baseline SQL tuning set.	✓	✗	✗	✗
Proactive Identification of high risk SQL statements	Find SQL statements whare SQL plans can change on increasing Data Volumes...	✓	✗	✗	✗

ORACLE

Program Agenda with Highlight

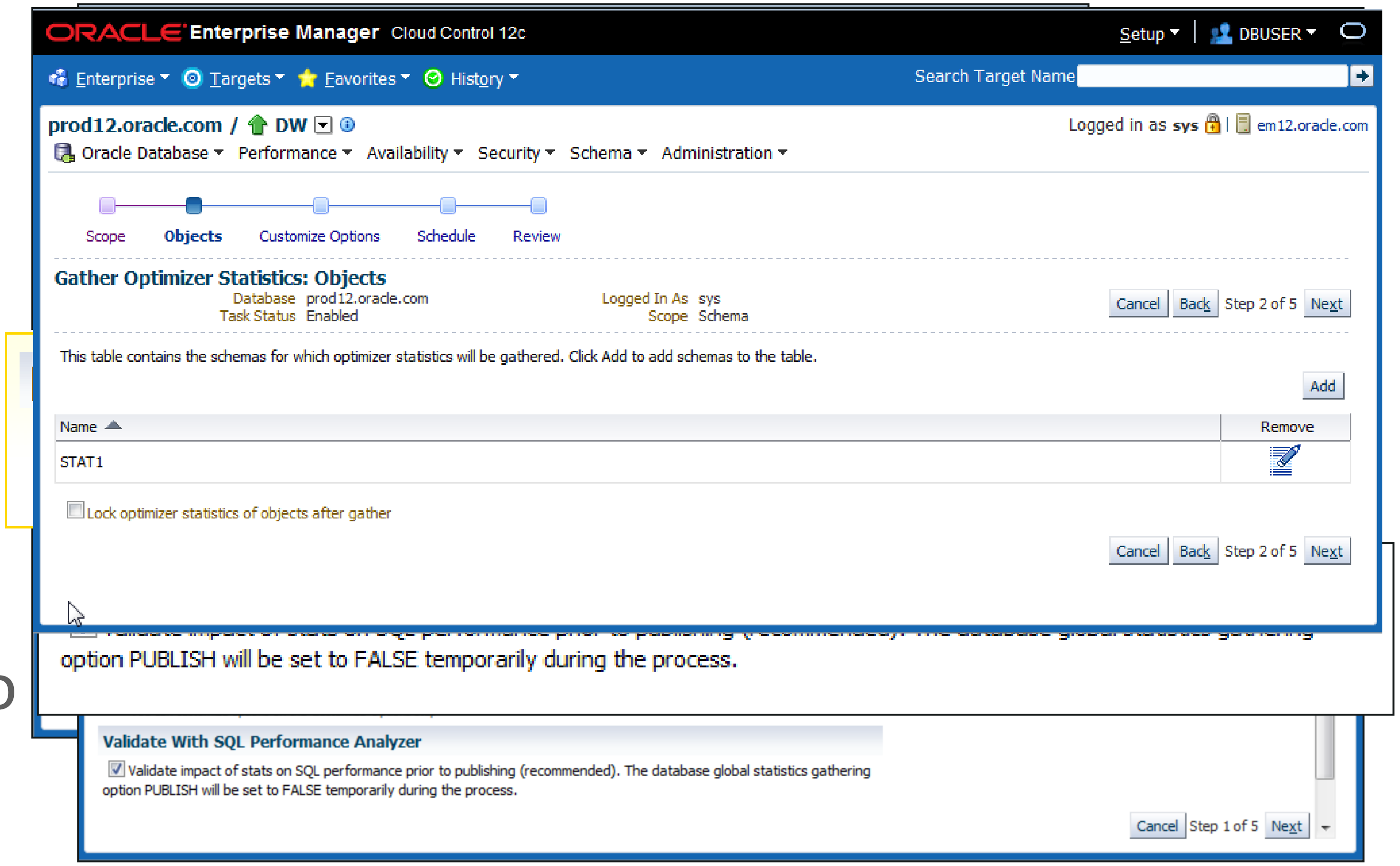
Optimizer Statistics Refresh

- Data growth requires
  - Up to date statistics for optimal query plans
- New statistics
  - Can lead to regression
- Which way to go?
  - Stale statistics (Slowly degenerated performance)
  - New statistics (Will there be any regression?)
- Let's find out!
  - And make sure there are no regression

Optimizer Statistics Refresh

Gather Statistics

- Go to Optimizer Statistics Page
- Choose Gather Statistics
- Choose the extent
- Choose to validate with SPA
- Choose object accoring to dialog
- Submit



Gather Optimizer Statistics: Objects

Database	prod12.oracle.com	Logged In As	sys
Task Status	Enabled	Scope	Schema

Cancel Back Step 2 of 5 Next

This table contains the schemas for which optimizer statistics will be gathered. Click Add to add schemas to the table.

Add

Name	Remove
STAT1	

☐ Lock optimizer statistics of objects after gather

Cancel Back Step 2 of 5 Next

option PUBLISH will be set to FALSE temporarily during the process.

Validate With SQL Performance Analyzer

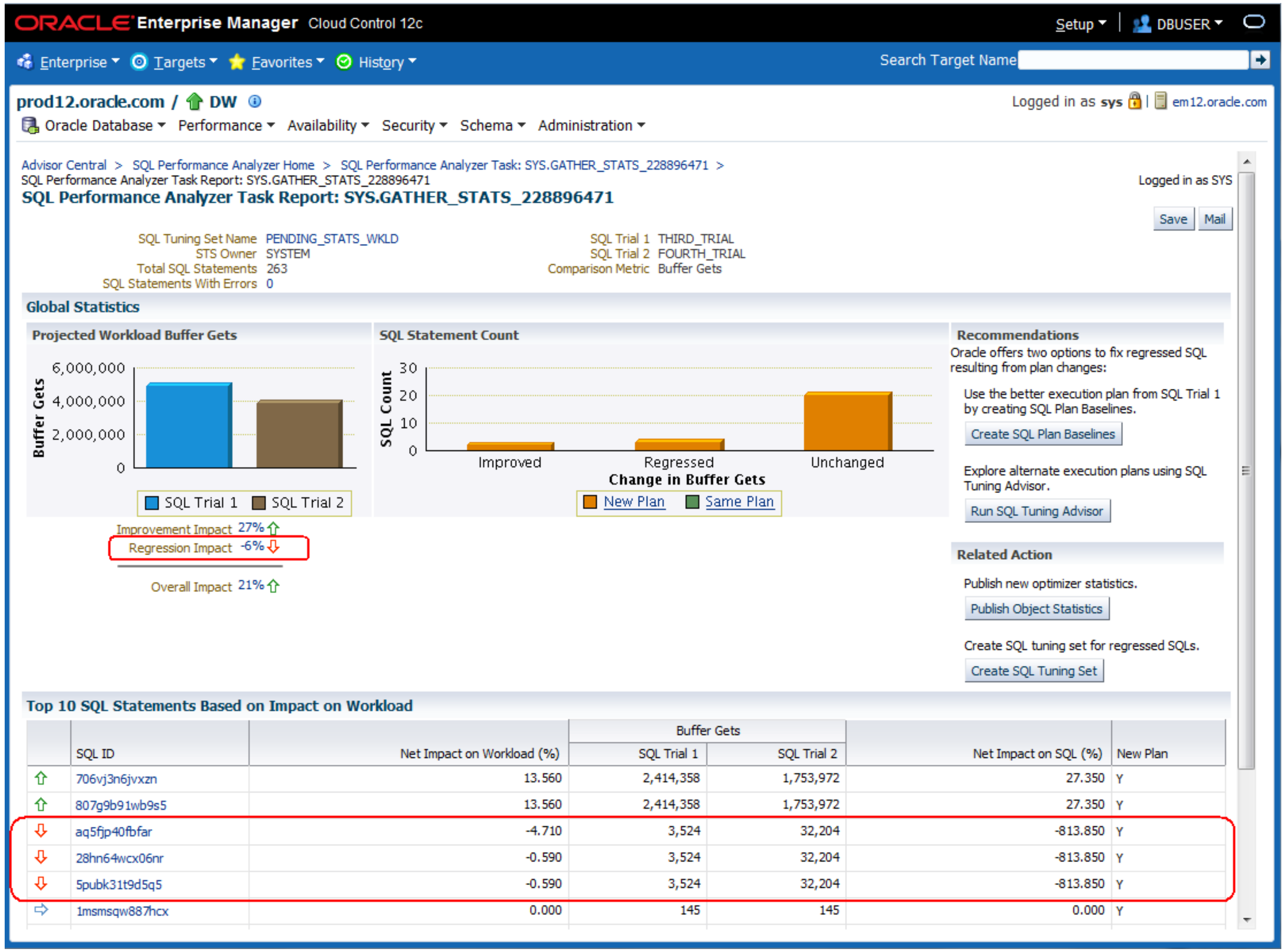
☑ Validate impact of stats on SQL performance prior to publishing (recommended). The database global statistics gathering option PUBLISH will be set to FALSE temporarily during the process.

Cancel Step 1 of 5 Next

Optimizer Statistics Refresh

SPA Validation

- Select your SPA Task
- Select the comparison report
  - Report between First and Second Trial identifies queries with plan changes
  - Report between Third and Fourth trial higligt differances during execution
- Identify regression



Advisor Central > SQL Performance Analyzer Home > SQL Performance Analyzer Task: SYS.GATHER_STATS_228896471 >
SQL Performance Analyzer Task Report: SYS.GATHER_STATS_228896471

SQL Performance Analyzer Task Report: SYS.GATHER_STATS_228896471

SQL Tuning Set Name	PENDING_STATS_WKLD	SQL Trial 1	THIRD_TRIAL
STS Owner	SYSTEM	SQL Trial 2	FOURTH_TRIAL
Total SQL Statements	263	Comparison Metric	Buffer Gets
SQL Statements With Errors	0		

Global Statistics

Projected Workload Buffer Gets

Improvement Impact 27%
Regression Impact -6%
Overall Impact 21%

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.

Related Action

Publish new optimizer statistics.

Create SQL tuning set for regressed SQLs.

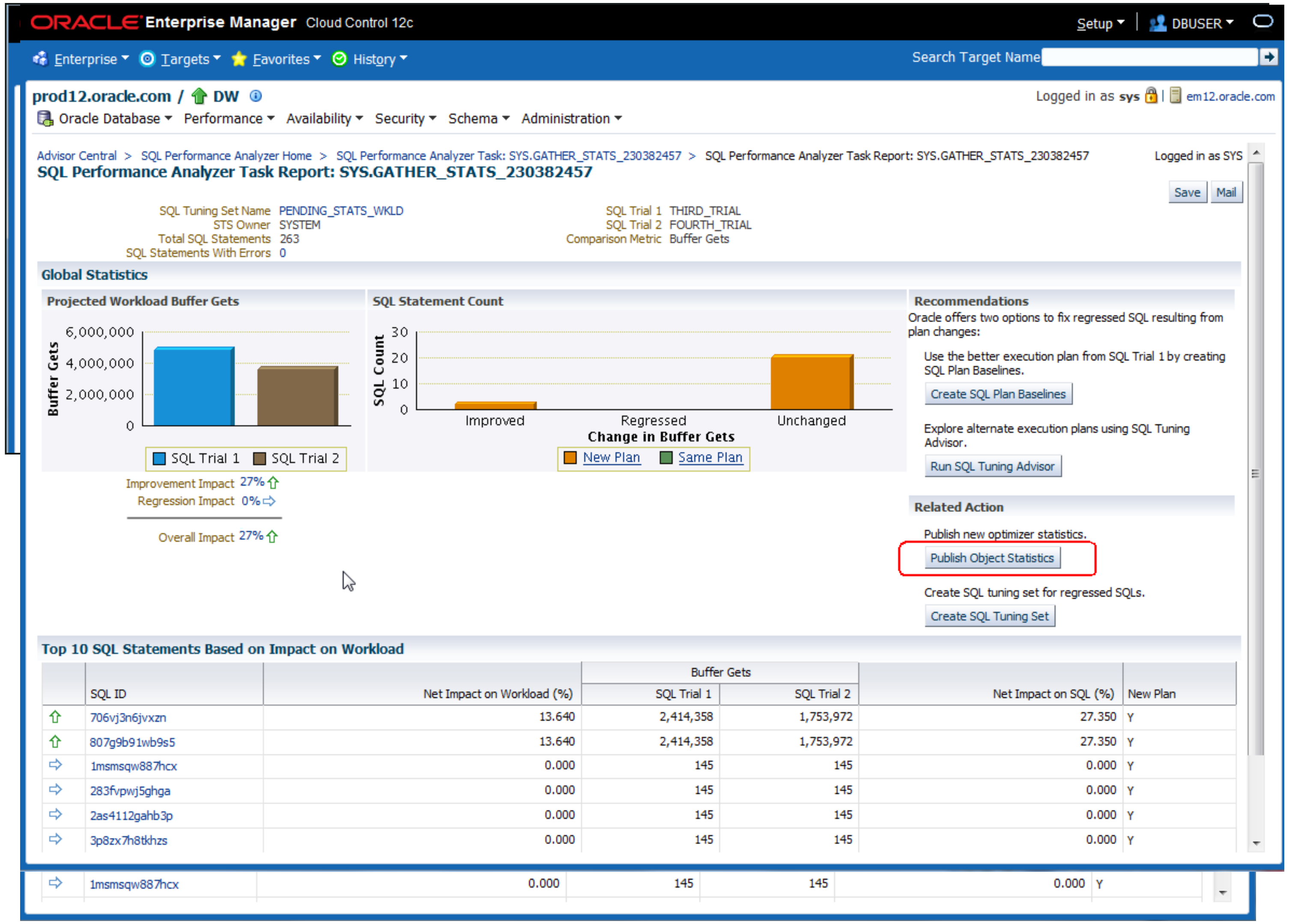
Top 10 SQL Statements Based on Impact on Workload

			Buffer Gets			
	SQL ID	Net Impact on Workload (%)	SQL Trial 1	SQL Trial 2	Net Impact on SQL (%)	New Plan
⇧	706vj3n6jvxzn	13.560	2,414,358	1,753,972	27.350	Y
⇧	807g9b91wb9s5	13.560	2,414,358	1,753,972	27.350	Y
⇩	aq5fjp40fbfar	-4.710	3,524	32,204	-813.850	Y
⇩	28hn64wcx06nr	-0.590	3,524	32,204	-813.850	Y
⇩	5pubk31t9d5q5	-0.590	3,524	32,204	-813.850	Y
⇨	1msmsqw887hcx	0.000	145	145	0.000	Y

Optimizer Statistics Refresh

Remedy and publish

- Remedy regression
  - SQL Plan Baseline
  - Tuning advisor
- Implement (for this example)
  - SQL Plan Baseline
- Validate again
- Publish Statistics



ORACLE Enterprise Manager Cloud Control 12c
Setup
DBUSER
Enterprise
Targets
Favorites
History
Search Target Name
prod12.oracle.com / DW
Logged in as sys | em12.oracle.com
Oracle Database
Performance
Availability
Security
Schema
Administration
Advisor Central > SQL Performance Analyzer Home > SQL Performance Analyzer Task: SYS.GATHER_STATS_230382457 > SQL Performance Analyzer Task Report: SYS.GATHER_STATS_230382457
Logged in as SYS
Save
Mail

SQL Performance Analyzer Task Report: SYS.GATHER_STATS_230382457

SQL Tuning Set Name	PENDING_STATS_WKLD
STS Owner	SYSTEM
Total SQL Statements	263
SQL Statements With Errors	0
SQL Trial 1	THIRD_TRIAL
SQL Trial 2	FOURTH_TRIAL
Comparison Metric	Buffer Gets

Global Statistics

Improvement Impact 27%
Regression Impact 0%
Overall Impact 27%

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.

Create SQL Plan Baselines

Explore alternate execution plans using SQL Tuning Advisor.

Run SQL Tuning Advisor

Related Action

Publish new optimizer statistics.

Publish Object Statistics

Create SQL tuning set for regressed SQLs.

Create SQL Tuning Set

Top 10 SQL Statements Based on Impact on Workload

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇧	706vj3n6jvxzn	13.640	2,414,358	1,753,972	27.350	Y
⇧	807g9b91wb9s5	13.640	2,414,358	1,753,972	27.350	Y
⇨	1msmsqw887hcx	0.000	145	145	0.000	Y
⇨	283fvpwj5ghga	0.000	145	145	0.000	Y
⇨	2as4112gahb3p	0.000	145	145	0.000	Y
⇨	3p8zx7h8tkhzs	0.000	145	145	0.000	Y
⇨	1msmsqw887hcx	0.000	145	145	0.000	Y

Program Agenda

Schema Optimization
Should I retain existing indexes?

- My workload on Exadata is not running faster than on my old machine
- How can I make it go faster?
- Should my queries use smart scan or Index range scan?
- Similarly, should I drop my indexes with In-Memory Option
- Let’s find out using invisible indexes!

Schema Optimization

- Drop Indexes
  - May impact workload performance
  - Time consuming to recreate
- No custom EM SPA workflow available
  - Let's do it manually with SPA Quick Check method using invisible indexes
- This should be done during maintenance window
  - We are going to change query plans

Exec DBMS_SQLPA...
Exec ....

Schema Optimization

Step1

- Create Analysis Task
  - DBMS_SQLPA.CREATE_ANALYSIS_TASK

Step2

- Run Explain Plan on all statements with current indexing
  - EXECUTE_ANALYSIS_TASK….execution_type => 'EXPLAIN PLAN'….

Step 3

- Hide indexes
  - alter index Index_name1 INVISIBLE;

Step 4

- Run Explain Plan on All statements <u>without</u> current indexing
  - EXECUTE_ANALYSIS_TASK….execution_type => 'EXPLAIN PLAN'….

Schema Optimization

Step5

- Create report on statements with plan changes
  - EXECUTE_ANALYSIS_TASK....execution_type => 'compare performance'

Step6

- Create filter for SQL statements with new execution plans
  - Apply filters to target SQL with new plans

Step 7

- Expose indexes to session
  - alter session SET OPTIMIZER_USE_INVISIBLE_INDEXES=TRUE';

Step 8

- Execute All statements with plan changes with current indexing
  - EXECUTE_ANALYSIS_TASK....execution_type => 'TEST EXECUTE'....

Schema Optimization

Step9

- Hide indexes
  - alter session SET OPTIMIZER_USE_INVISIBLE_INDEXES=FALSE';

Step10

- Execute All statements with plan changes <u>without</u> current indexing
  - EXECUTE_ANALYSIS_TASK….execution_type => 'TEST EXECUTE'….

Step 11

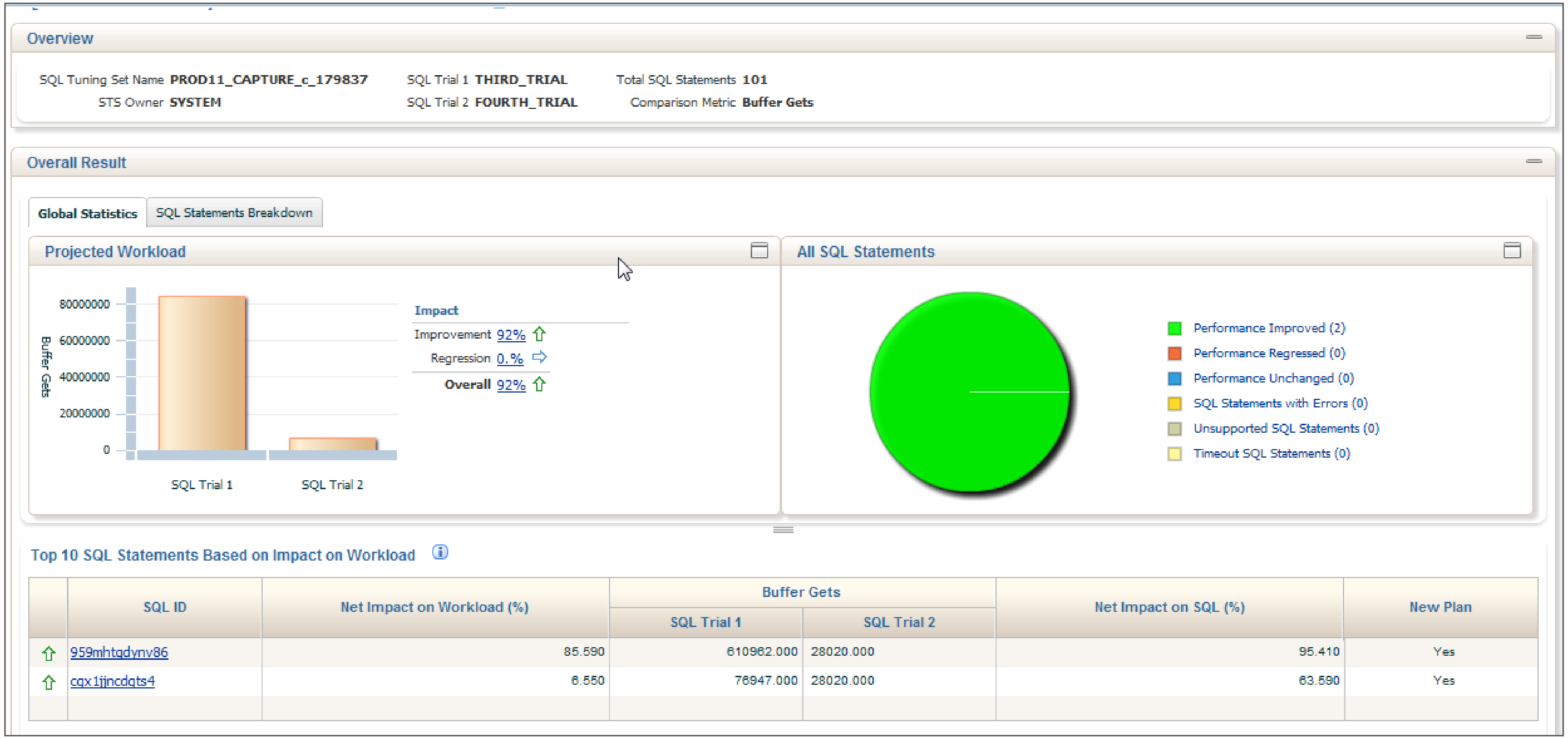
- Generate compare analysis
  - EXECUTE_ANALYSIS_TASK…execution_type => 'COMPARE PERFORMANCE'…

Step 12

- Generate SPA Active Report
  - spool spa_active.html, SELECT DBMS_SQLPA.REPORT_ANALYSIS_TASK… type => 'active', …

Schema Optimization

View the result



Overview

SQL Tuning Set Name	**PROD11_CAPTURE_c_179837**	SQL Trial 1	**THIRD_TRIAL**	Total SQL Statements	**101**
STS Owner	**SYSTEM**	SQL Trial 2	**FOURTH_TRIAL**	Comparison Metric	**Buffer Gets**

Impact

Improvement	92% ⇧
Regression	0.% ⇨
Overall	92% ⇧

- 🟩 Performance Improved (2)
- 🟥 Performance Regressed (0)
- 🟦 Performance Unchanged (0)
- 🟨 SQL Statements with Errors (0)
- ▨ Unsupported SQL Statements (0)
- 🟨 Timeout SQL Statements (0)

Top 10 SQL Statements Based on Impact on Workload

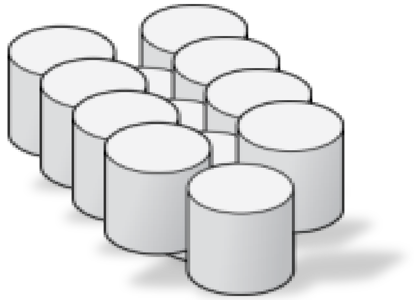
	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇧	959mhtqdynv86	85.590	610962.000	28020.000	95.410	Yes
⇧	cqx1jjncdqts4	6.550	76947.000	28020.000	63.590	Yes

Program Agenda

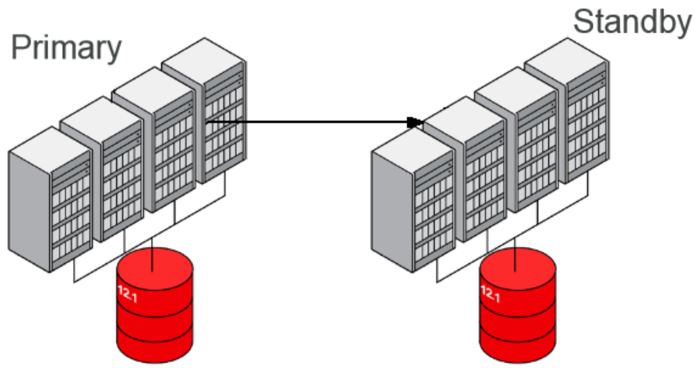
Validate In-Memory with Query Only Replay

- How should I configure In-memory?
  - Which tables to cache?
  - In-Memory Advisor

- Large production Environment
  - Time consuming to clone
  - No extra hardware



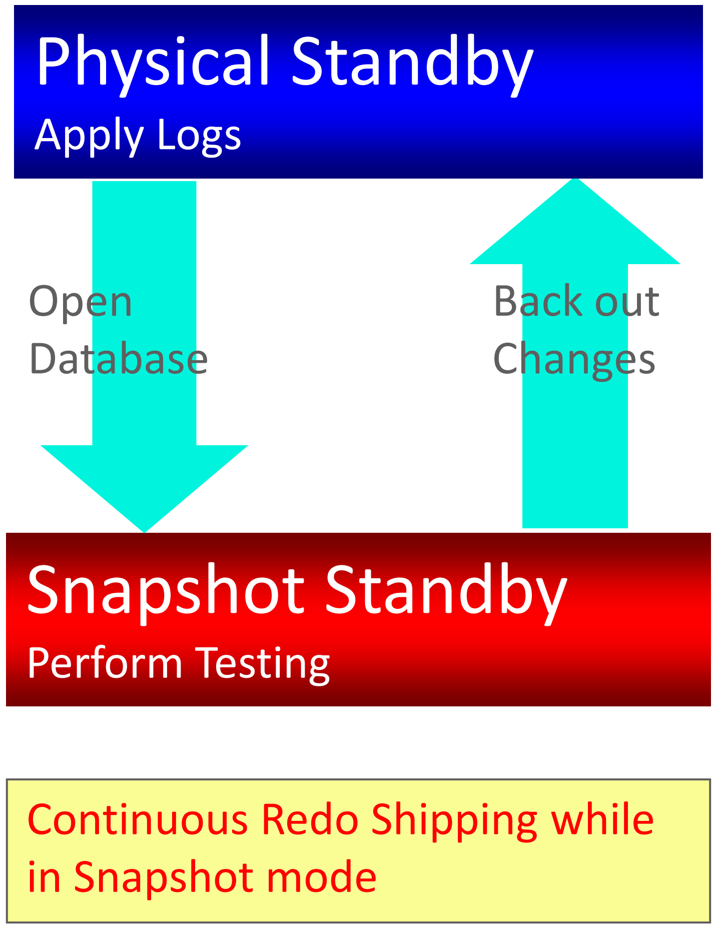
- Standby database
  - Can I use my standby database?



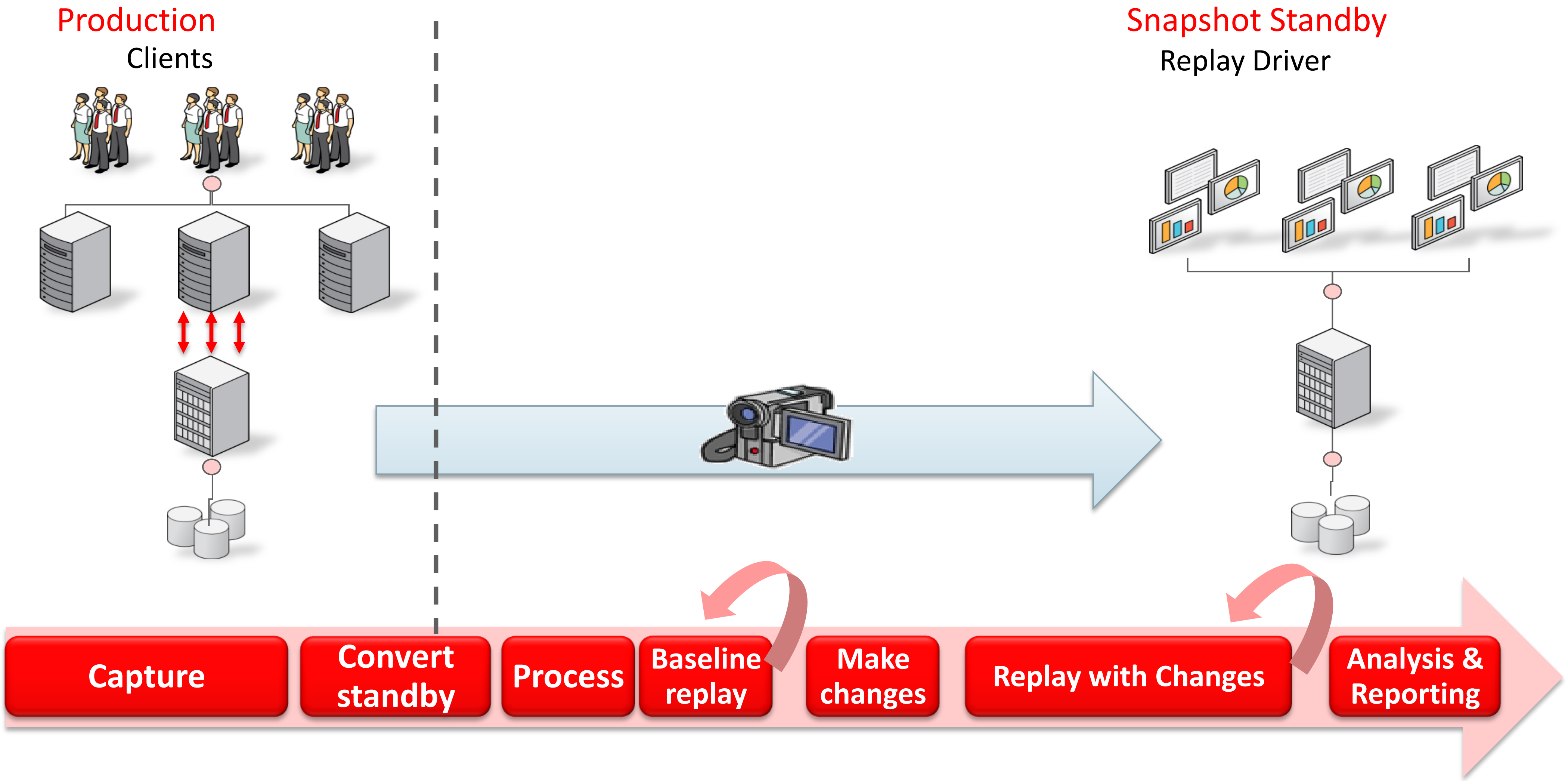
Validate In-Memory with Query Only Replay

Snapshot Standby

- Works in conjunction with Real Application Testing
  - Provides a simple way to test and maintain protection
  - Hardware available
  - Current data set



Validate In-Memory with Query Only Replay



Validate In-Memory with Query Only Replay

Step1

- Capture Workload
  - Use Enterprise manager or API (DBMS_WORKLOAD_CAPTURE.START_CAPTURE)

Step2

- Convert to Snapshot Standby
  - Use Enterprise manager or API (convert database to snapshot standby)

Step 3

- Move and process capture on Snapshot Standby
  - Use Enterprise manager or API (DBMS_WORKLOAD_REPLAY.PROCESS_CAPTURE)

ORACLE

Validate In-Memory with Query Only Replay

Step4

- Run baseline replay – No DML are executed so we can not compare with production (Repeat to heat the cache)
  - Need to use consolidated replay API
    - exec dbms_workload_replay.set_replay_directory('INMEM');
    - exec dbms_workload_replay.begin_replay_schedule('S1');
    - select dbms_workload_replay.add_capture(capture_dir_name => 'INMEM',…, **query_only => 'Y')** from dual;
    - exec dbms_workload_replay.end_replay_schedule…

Query Only Replay flag

Step5

- Make changes
  - alter table DISTRIBUTION_DEPT_TAB2 inmemory MEMCOMPRESS FOR QUERY LOW;
  - alter system set inmemory_size=1G scope=spfile;
    - Shutdown
      - Startup;

ORACLE

Validate In-Memory with Query Only Replay

Step6

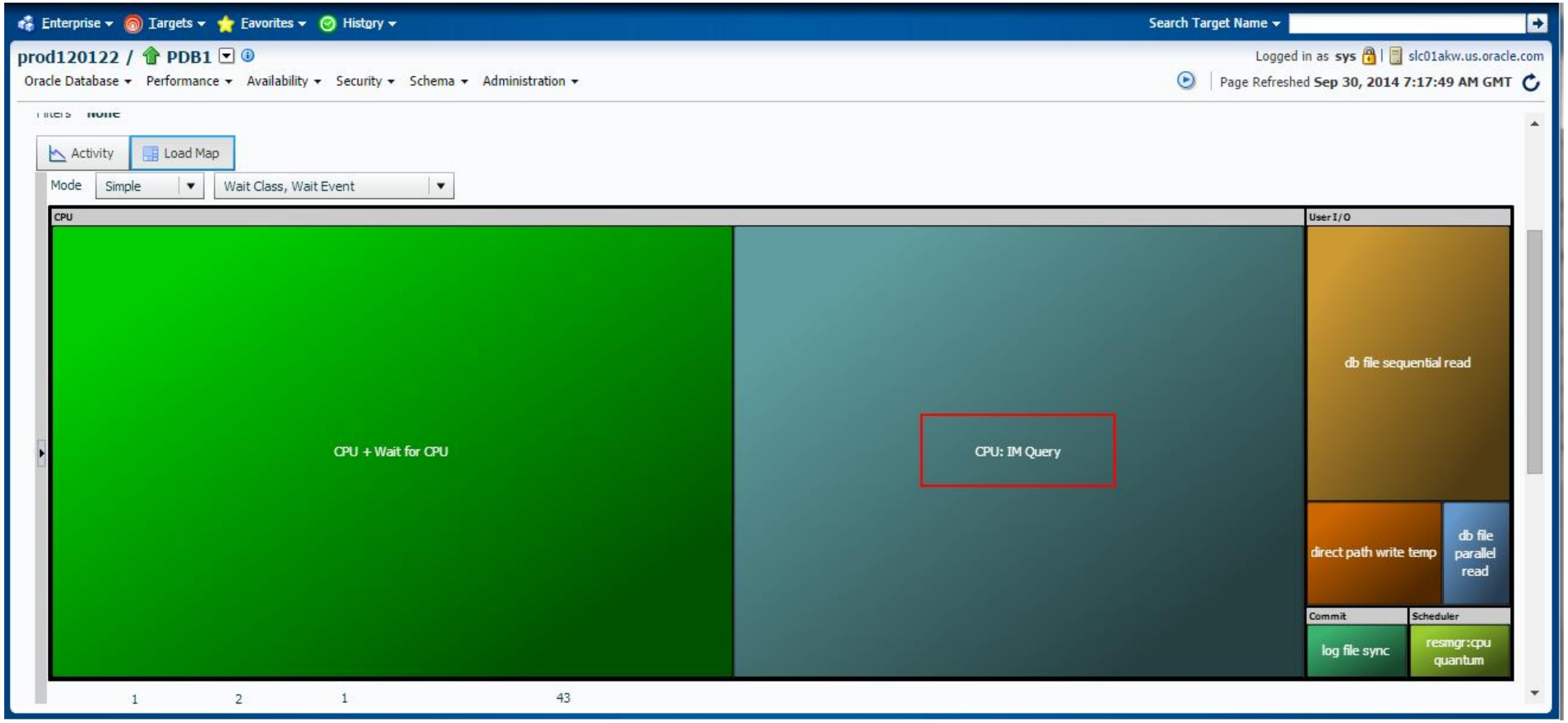
- Run In-Memory replay – No DML is executed so we can not compare with production (Repeat to heat the cache)
  - Need to use consolidated replay API
    - exec dbms_workload_replay.start_consolidated_replay;

Step 7

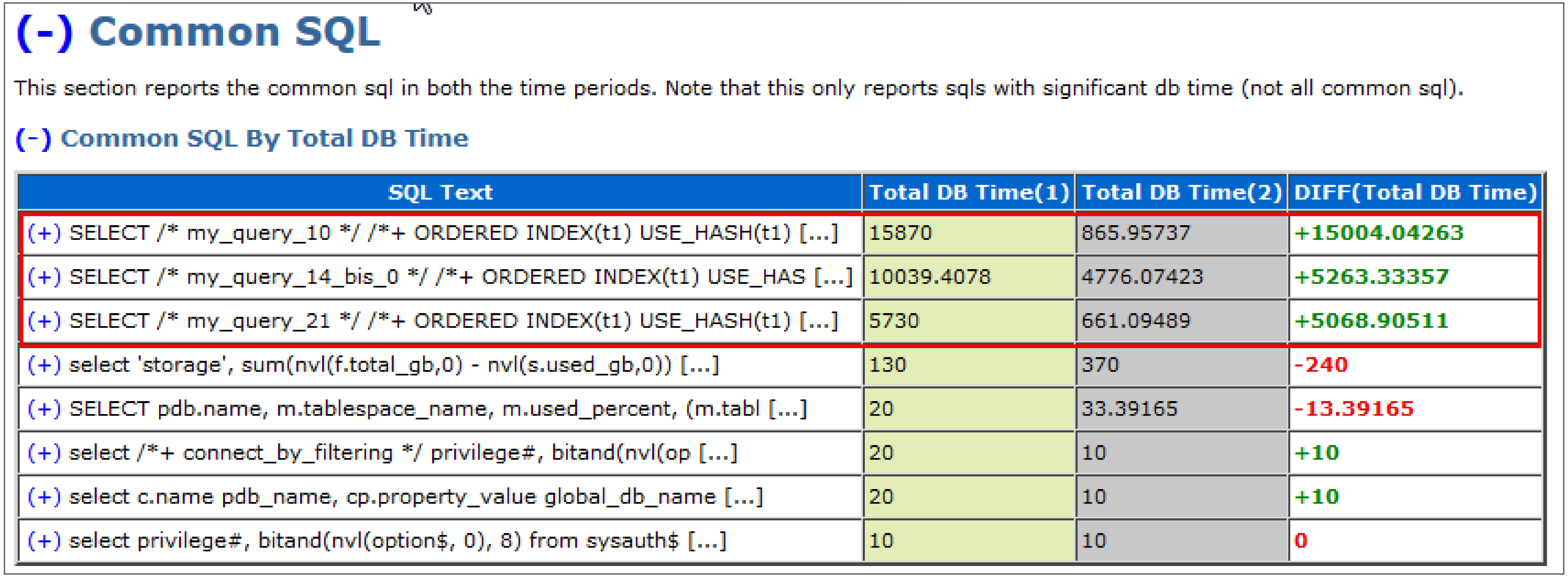
- Generate reports and analyze result
  - Use Enterprise Manager or API
    - exec DBMS_WORKLOAD_REPLAY.COMPARE_PERIOD_REPORT…

ORACLE

Monitor Replay: Use ASH Analytics



Assess In-Memory Option: Use Replay Compare Period Report



(-) Common SQL

This section reports the common sql in both the time periods. Note that this only reports sqls with significant db time (not all common sql).

(-) Common SQL By Total DB Time

SQL Text	Total DB Time(1)	Total DB Time(2)	DIFF(Total DB Time)
(+) SELECT /* my_query_10 */ /*+ ORDERED INDEX(t1) USE_HASH(t1) [...]	15870	865.95737	+15004.04263
(+) SELECT /* my_query_14_bis_0 */ /*+ ORDERED INDEX(t1) USE_HAS [...]	10039.4078	4776.07423	+5263.33357
(+) SELECT /* my_query_21 */ /*+ ORDERED INDEX(t1) USE_HASH(t1) [...]	5730	661.09489	+5068.90511
(+) select 'storage', sum(nvl(f.total_gb,0) - nvl(s.used_gb,0)) [...]	130	370	-240
(+) SELECT pdb.name, m.tablespace_name, m.used_percent, (m.tabl [...]	20	33.39165	-13.39165
(+) select /*+ connect_by_filtering */ privilege#, bitand(nvl(op [...]	20	10	+10
(+) select c.name pdb_name, cp.property_value global_db_name [...]	20	10	+10
(+) select privilege#, bitand(nvl(option$, 0), 8) from sysauth$ [...]	10	10	0

In Summary

- By adopting Real Application Testing you will:
  - Automate validation process
  - Reduce time spent on each database
  - Reduce downtime due to untested changes
  - Help you to be agile by adopting new features

Hardware and Software
Engineered to Work Together