



ORACLE®

Database Testing: Introducing DB Infrastructure Level Change with Full Confidence

**Prabhaker Gongloor (GP)
Product Management Director
Real Application Testing and Diagnosability**



Outline

- Database Testing: Current State
- Real Application Testing
 - Database Replay
 - SQL Performance Analyzer
- Usage scenarios and recommended testing methodology
 - 10.2 → 11g Database upgrade
 - RAC
 - Platform migration
 - Using SPA in production environment
- Conclusion



Database Testing: Current Landscape

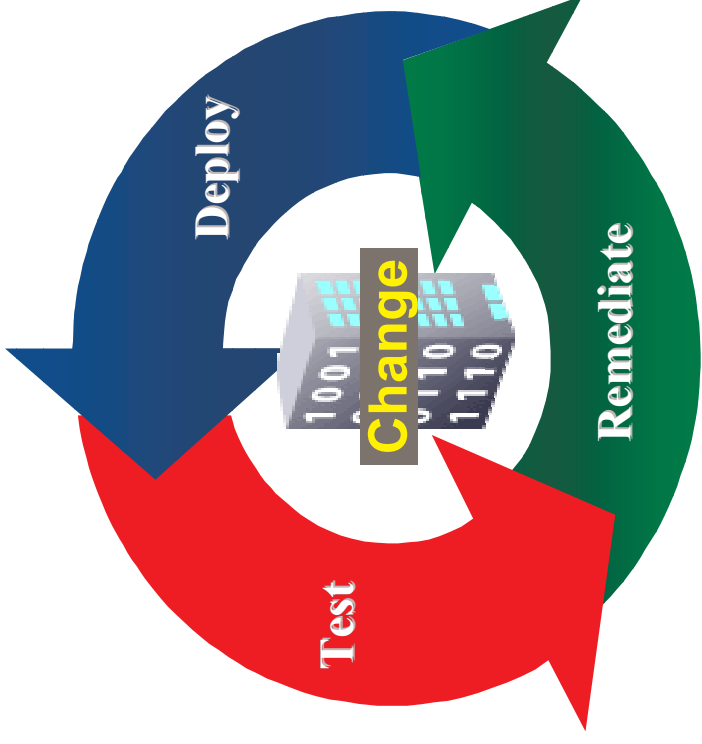
- **Businesses need to adapt to changes to stay competitive, compliant and evolve**
 - DB upgrades, RAC, HW/OS/storage, Exadata, config changes, etc.
 - Optimizer and SQL performance related changes
 - Current testing often still results in missed SLAs, instabilities
- **Current testing landscape and limitations**
 - Large workloads (100K SQL statements)
 - Testing through non-production, synthetic workloads
 - Partial workflow coverage – Typically <5%
 - Time consuming – Months required & human capital intensive
 - No end-to-end solution – How to build test system, fix regressions, tun



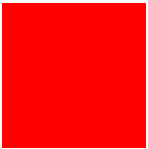
Real Application Testing allows comprehensive & easy-to-use method for system change assessment

Real Application Testing

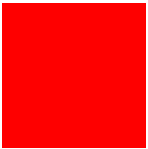
- Value
 - Rapid technology adoption
 - Higher testing quality
- Business Benefit
 - Lower cost
 - Lower risk
- Features
 - Database Replay
 - SQL Performance Analyzer (SPA)



**Business Agility through
Superior Testing**

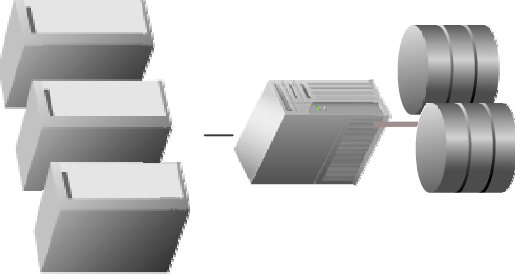
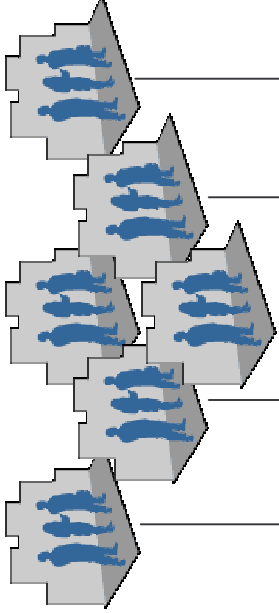


Database Replay

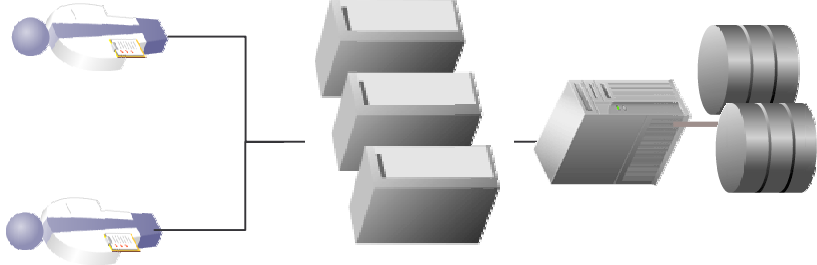


Testing Today

Test – 1-2 testers trying to be 1,000s of users

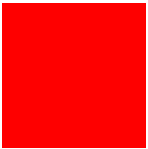


PRODUCTION



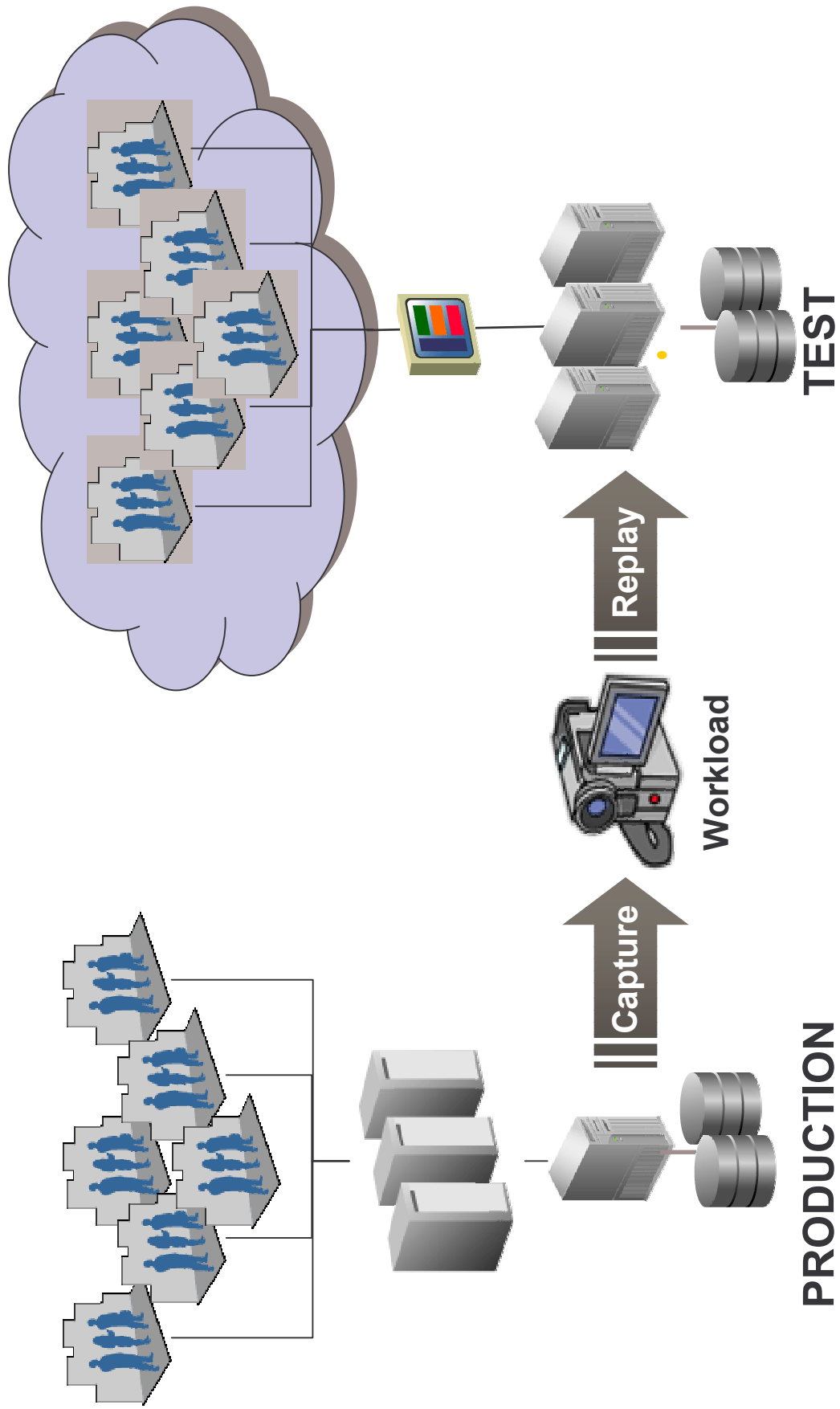
TEST





Database Replay

Test your system changes at production levels



Database Replay

- 1 Capture
- 2 Replay
- 3 Analyze & Remediate



- **Capture Workload in Production**

- Capture full production workload with real load, timing & concurrency characteristics
- Move the captured workload to test system

- **Replay Workload in Test**

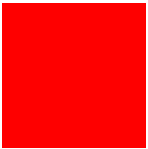
- Make the desired changes in test system
- Replay workload with full production characteristics

- **Analyze & Report**

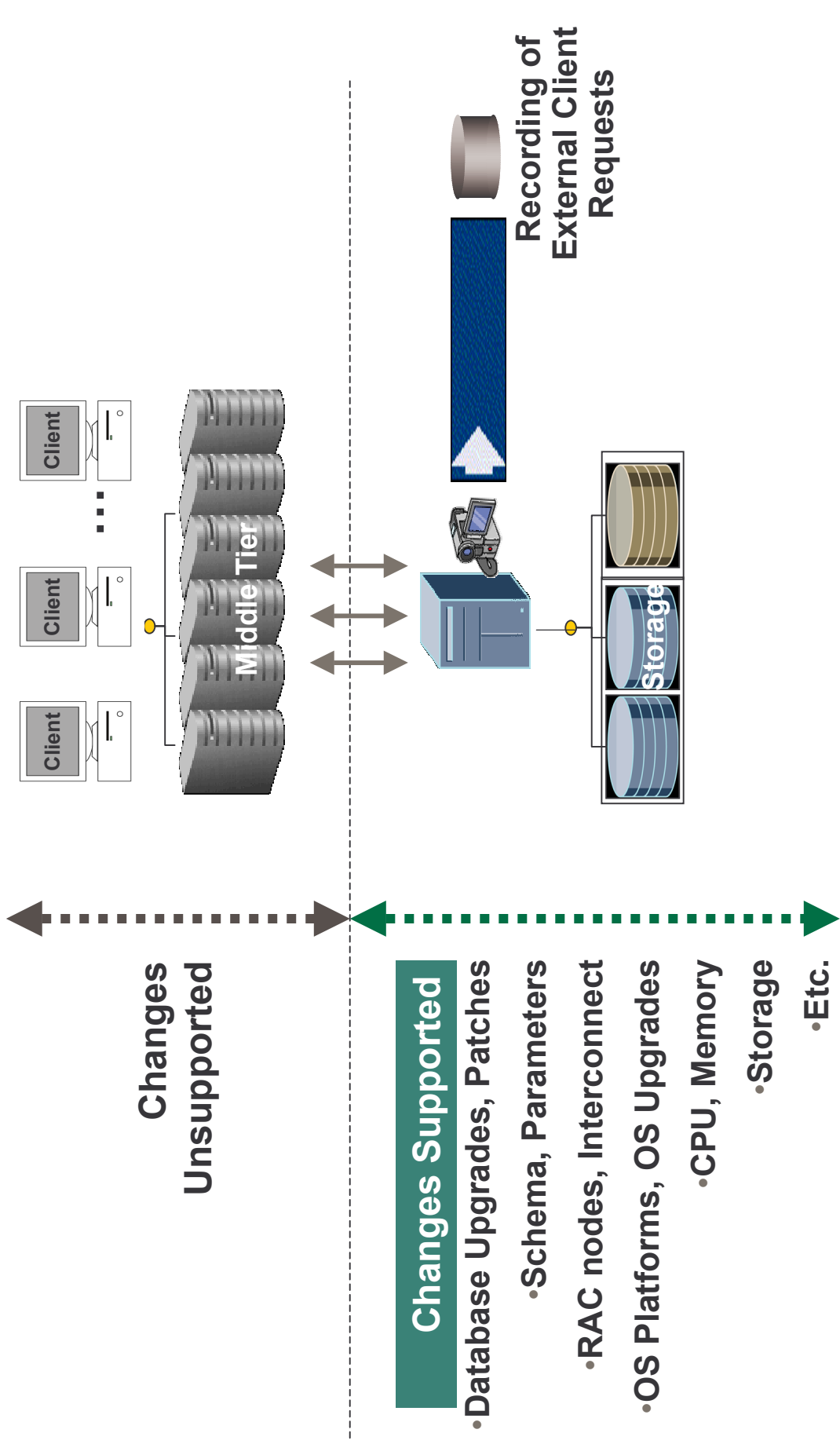
- Capture and Replay Reports
- AWR, ASH, Replay Compare Period Reports



Analysis & Reporting



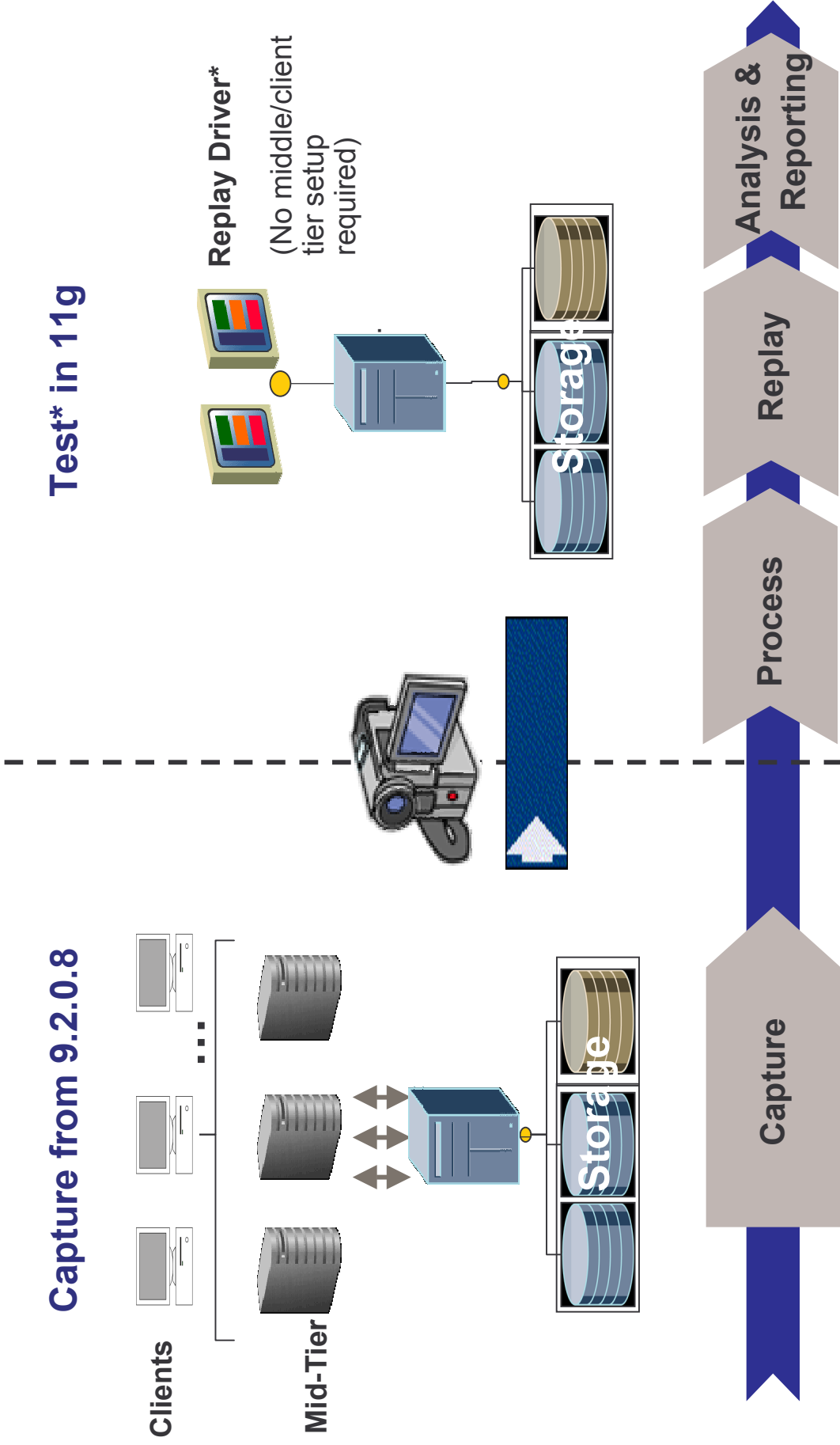
Supported Changes





Database Replay Workflow

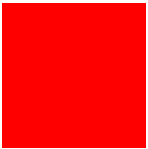
Example: Upgrading from Oracle Database 9.2.0.8 to 11g



ORACLE

† ML 560977.1: Real Application Testing for Earlier Releases

* Use SPA for pre-11g release testing



Using Database Replay for Upgrade Testing



10.2 → 11g Database Upgrade

Scenario:

How can I use Database Replay to check if my peak workload will perform as expected after upgrade from 10.2 to 11g?

Goal:

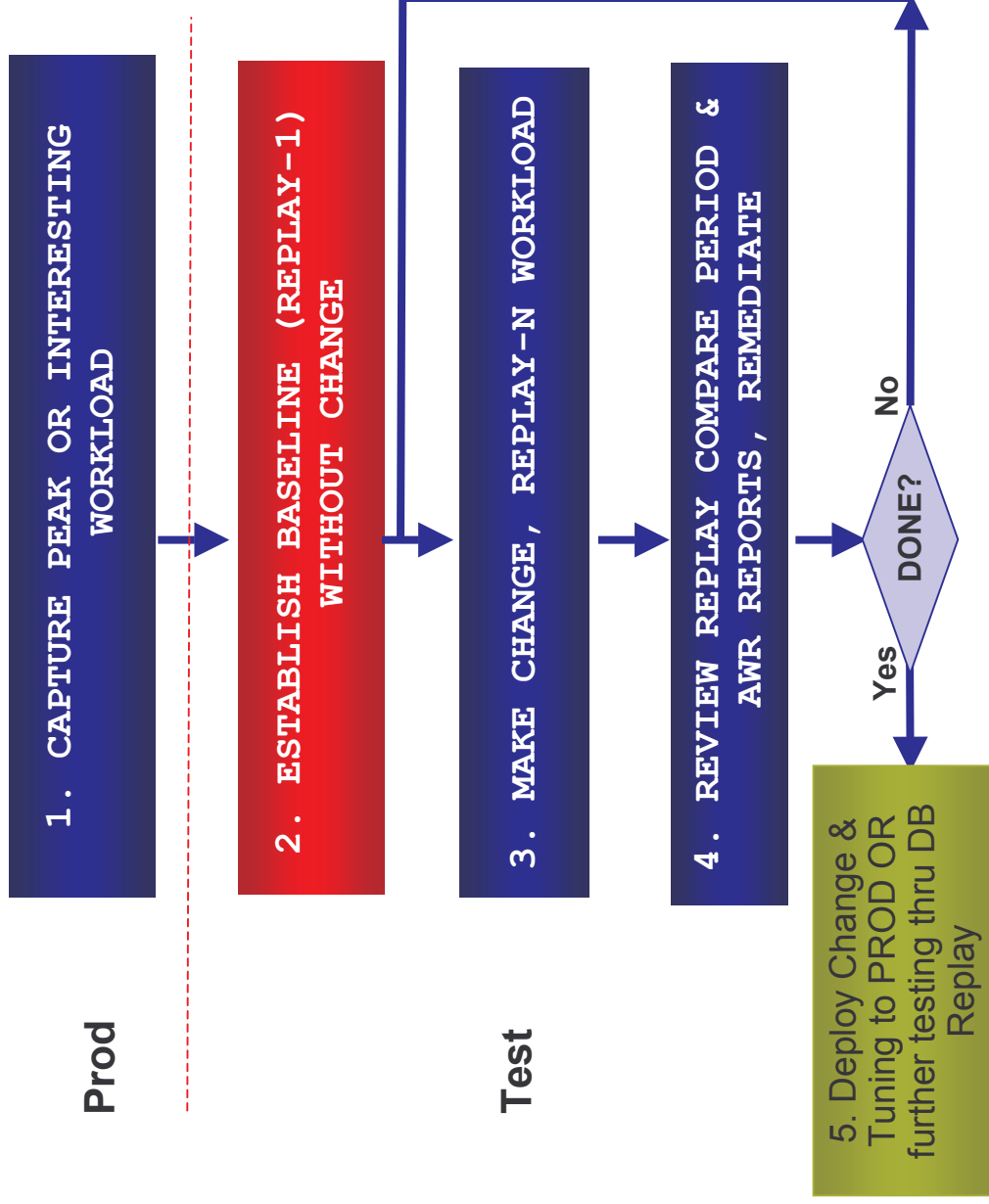
Test the impact of Oracle Database 11g upgrade on the peak workload captured on production system & make sure are no negative effects of the change



Siebel Workload Description

- Siebel PSPP workload used for testing DB upgrade scenario
 - Used internally for upgrade certification and new feature uptake
- Siebel 8.0, 1300 users: 700 financial call center, 600 financial partner manager
- Financial call center scenario:
 - Creates new contact
 - Creates new opty for the contact
 - Add products to the opty
 - Creates quotes
 - Converts quotes to order
- Financial partner manager scenario
 - Creates a new service request
 - Assigns the service request

Recommended Testing Methodology with Database Replay





Recommended Testing Methodology with Database Replay

- Use identical Test system if possible
- Test one change at a time to understand causality
- Use **Replay Compare Period Report** to understand Baseline (Replay 1) deviations to production capture
- Compare two replays in the same environment
 - Baseline to Replay N
 - Replay N-1 to Replay N for incremental changes and tuning
- Validate replay using application metrics (e.g., order entered/sec)



Replay Compare Period Report



Enhanced
in 11.2!

- Your new **best friend** in analyzing replay information!!
 - Provides holistic view of the experiment – covers functional and performance aspects of testing
 - “Replay Divergence Summary” categorization indicates if further analysis is necessary: LOW/MED/HIGH
- Two reports are available
 - Capture Vs Replay, Replay Vs Replay
- Identifies interference from other workloads, e.g., maintenance windows or other non-replayed workload
- Automatically runs ADDM
- Reports provide more accurate performance analysis
 - Uses enhanced ASH infrastructure for capture/replay sessions

Replay Summary (For Siebel Workload)

Database Instance: [siamst](#) > [Database Replay](#) > [Replay Workload](#) >

View Workload Replay: [wrr-20090903-122103](#)

Logged in As SYSTEM

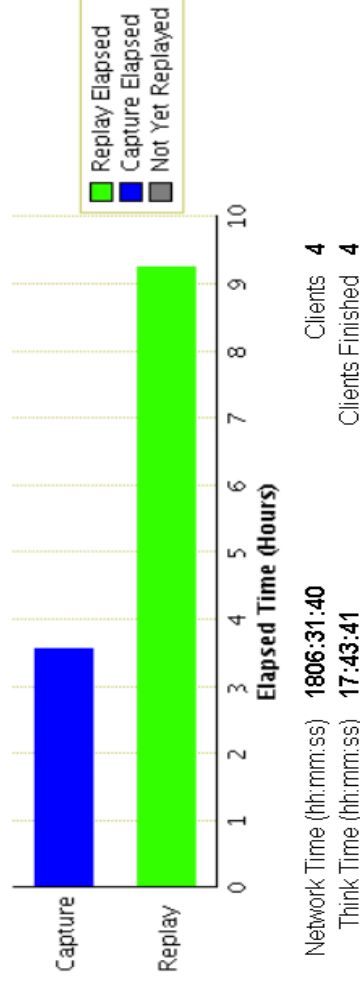
OK

Summary

Replay Name	wrr-20090903-122103	Capture Name	capture
Status	Completed	Duration (hh:mm:ss)	09:15:09
Directory Object	siebel	Prepare Time	Sep 3, 2009 12:21:04 PM PDT
Database Name	SIAMST	Start Time	Sep 3, 2009 12:34:28 PM PDT
DBID	2968440095	End Time	Sep 3, 2009 9:49:37 PM PDT
Replay Error Code	n/a		
Replay Error Message	None		

Workload Profile | [Connection Mappings](#) | [Replay Parameters](#) | [Report](#)

Chart Type: **Elapsed Time Comparison**



Assessing the Replay

The Elapsed Time Comparison chart shows how much time the replayed workload has taken to accomplish the same amount of work as captured.

When the Replay bar is shorter than the Capture bar, the replay environment is processing the workload faster than the capture environment.

The divergence table gives information about both the data and error discrepancies between the replay and capture environments, which can be used as a measure of the replay quality.

Replay Summary (Contd.) : Errors and Data Divergence

Divergence		Number of Calls	Percentage of Total Calls
Error Divergence:		0	0.00
Session Failures Seen During Replay		24	0.00
Errors No Longer Seen During Replay		2489	0.02
Errors Mutated During Replay		853	0.01
New Errors Seen During Replay			
Data Divergence:			
DMLs with Different Number of Rows Modified		200	0.00
SELECTs with Different Number of Rows Fetched		425	0.00

Detailed Comparison			
	Capture	Replay	Percentage of Capture
Duration (hh:mm:ss)	03:33:47	09:15:09	259.68
Database Time (hh:mm:ss)	09:07:54	17:16:07	189.11
Average Active Sessions	2.56	1.87	72.82
User Calls	11,212,622	11,212,577	100.00

Replay Errors and Data Divergence Analysis

Database Instance: [siamst](#) > [Database Replay](#) > [Replay Workload](#) > [View Workload Replay: wrr-20090903-122103](#) > Logged in As SYSTEM

Diverged Calls During Replay: wrr-20090903-122103

View the most relevant set of replayed calls that have diverged from the capture by filtering out all but the ones of interest. A large number of calls may be relevant. If so, consider grouping them by an attribute value they have in common.

▼ Conditions for Displaying Diverged Calls

Filter Conditions for Diverged Calls

Only the calls that meet all the following filter conditions will be displayed among the results. Conditions with empty values will not contribute to filtering.

Type of Divergence Observed
Select the blank choice to prevent filtering by type.

SQL ID

Service

Errors Mutated During Replay

Session Failures Seen During Replay

Errors No Longer Seen During Replay

Errors Mutated During Replay

New Errors Seen During Replay

DMLs with Different Number of Rows Modified

SELECTs with Different Number of Rows Fetched

SQL ID	Timestamp	Type of Divergence	Divergence Details	Rows Observed	Error Observed	Service	Module	Action	Session
	Sep 3, 2009 12:40:06 PM PDT	Error Mutated During Replay	Expected Error Number: 25228		15566	siamst	emagent_AQMetrics DEQ		210:106
	Sep 3, 2009 12:40:06 PM PDT	Error Mutated During Replay	Expected Error Number: 25228		15566	siamst	emagent_AQMetrics DEQ		210:106

Replay Errors and Data Divergence Grouping

Diverged Calls During Replay: wrr-20090903-122103

View the most relevant set of replayed calls that have diverged from the capture by filtering out all but the ones of interest. A large number of calls may be relevant. If so, consider grouping them by an attribute value they have in common.

▼ Conditions for Displaying Diverged Calls

Filter Conditions for Diverged Calls

Only the calls that meet all the following filter conditions will be displayed among the results. Conditions with empty values will not contribute to filtering.

Type of Divergence Error Observed

Select the blank choice to prevent filtering by type.

SQL ID

Session (ID:Serial#) :

Service

Module

Action

Grouping of Filtered Calls

Summarize the diverged calls displayed using one of the following options:

Grouping Attribute

Grouping on filtered attributes will be ignored.

Fetch Diverged Calls

Diverged Calls

Below is the grouped table of diverged calls that matches the specified attribute. Expand the table row to see the individual calls in the group. **TIP** The SQL ID link for individual calls allows you to drill down to call details have not yet been loaded into the server. Click the link to load the call details for each call currently displayed.

Details	Session (ID:Serial#)	Completed Without Failure	SQL ID	Timestamp	Type of Divergence	Divergence Details	Rows Observed	Error Observed	Service	Module	Action	Session
Show	210:106	✓	Sep 3, 2009 12:40:06 PM PDT	Error Mutated During Replay	Expected Error Number: 25228	15566	siamst	emagent_ACMetrics DEQ	210:106			
Show	12:380	✓	Sep 3, 2009 12:40:06 PM PDT	Error Mutated During Replay	Expected Error Number: 25228	15566	siamst	emagent_ACMetrics DEQ	210:106			
			Sep 3, 2009 12:40:06 PM PDT	Error Mutated During Replay	Expected Error Number: 25228	15566	siamst	emagent_ACMetrics DEQ	210:106			
			Sep 3, 2009 12:40:06 PM PDT	Error Mutated During Replay	Expected Error Number: 25228	15566	siamst	emagent_ACMetrics DEQ	210:106			

Replay Analysis: Reports

Database Instance: siarnst > Database Replay > Replay Workload >
View Workload Replay: wrr-20090909-165137

Summary

[Workload Profile](#) [Connection Mappings](#) [Replay Parameters](#) [Report](#)

Workload Replay Report [Run Report](#)

Compare Period Report

First Workload Capture or Replay mycapture (Sep 16, 2009 10:20:52 AM) [▼](#)

Second Workload Capture or Replay REPLAY-orclj.us.oracle.com-20090929142400 (Sep 29, 2009 2:25:16 PM) [▼](#)

[Run Replay_Compare_Period_Report](#) [Run_AWR_Compare_Period_Report](#)

AWR Report

Workload Capture or Replay REPLAY-orclj.us.oracle.com-20090929142400 (Sep 29, 2009 2:25:16 PM) [▼](#)

[Run Report](#)

ASH Report

Workload Capture or Replay REPLAY-orclj.us.oracle.com-20090929142400 (Sep 29, 2009 2:25:16 PM) [▼](#)

Start Date [📅](#)
(example: Oct 9, 2009)

Start Time AM PM

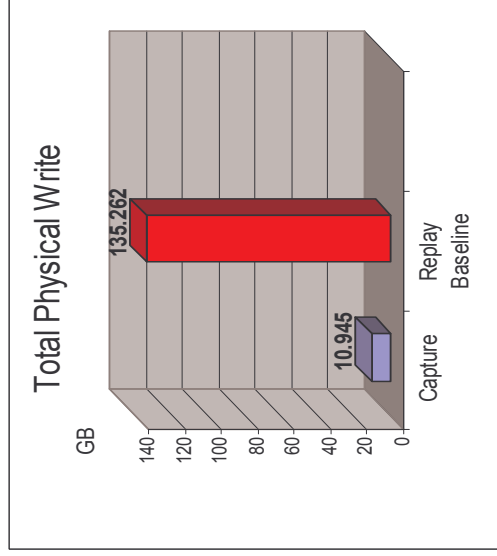
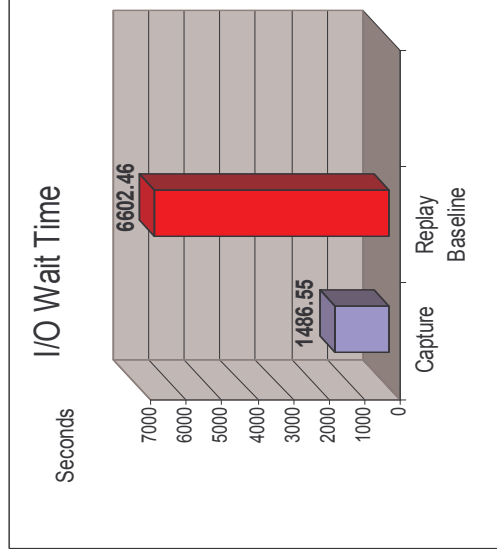
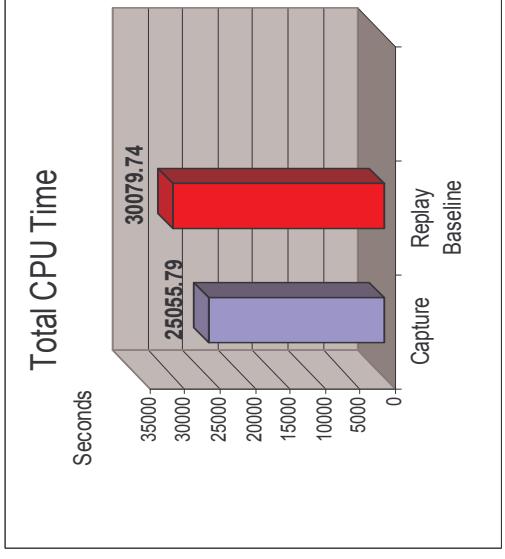
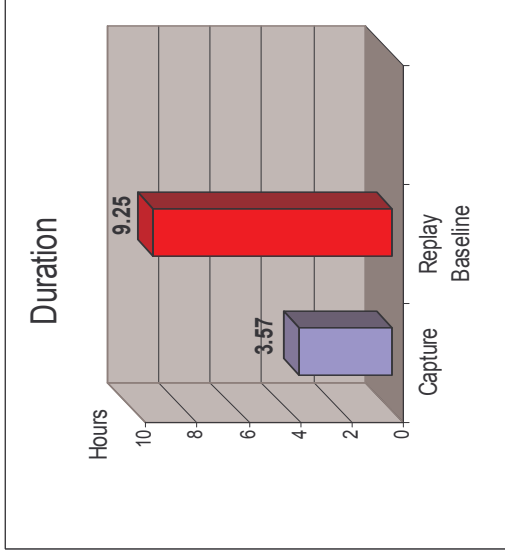
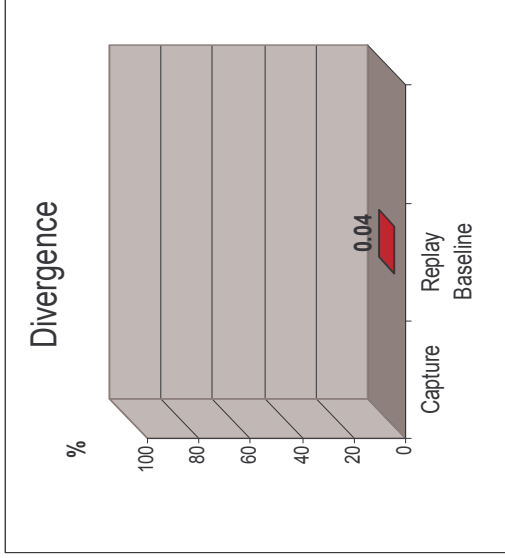
End Date [📅](#)
(example: Oct 9, 2009)

End Time AM PM

Filter [▼](#)

[Run Report](#)

Replay Compare Period: 10.2.0.4 Capture Vs 11.2.0.1 Replay Baseline



[Compare Period Report Link](#)

Important Changes between Capture and Replay Baseline

(-) Changes to Important Parameters

	Capture	Replay
sessions	566	792
compatible	10.2.0.4	11.2.0

(-) Changes to Optimizer-Relevant Parameters

	Capture	Replay
optimizer_capture_sql_plan_baselines	NULL	FALSE
optimizer_use_invisible_indexes	NULL	FALSE
optimizer_use_pending_statistics	NULL	FALSE
optimizer_use_sql_plan_baselines	NULL	TRUE
optimizer_features_enable	10.2.0.4	11.2.0.1
result_cache_mode	NULL	MANUAL

No Changes to Memory Configuration Parameters

(-) Instances of the Capture Database

instance number	instance name	host name	number of CPU cores	number of CPU sockets	physical memory	instance type
1	siamst	stase07	8	8	31.73 G	RDBMS

(-) Instances of the Replay Database

instance number	instance name	host name	number of CPU cores	number of CPU sockets	physical memory	instance type
1	siamst	stase07	4	4	31.73 G	RDBMS

Replay Period Report: Top SQL Changes

Force Matching Signature	example SQL_ID	Change in DB Time	Change in Average Response Time	Capture DB time	Replay DB time	example sql text
9011963922445766669	44u0kwtwmd7h5	21209.57 seconds	7175.79%	295.57 seconds	21505.14 seconds	(+) SELECT T9.CONFLICT_ID, T9.LAST_UPD, T9.CRE [...]
8567836812353523192	628g90748qrd5	-5947.74 seconds	-99.3%	5989.66 seconds	41.92 seconds	(+) SELECT T34.CONFLICT_ID, T34.LAST_UPD, T34. [...]
9389583114292788166	dry9f79q87ngv	-3150.3 seconds	-99.01%	3181.74 seconds	31.44 seconds	(+) SELECT T34.CONFLICT_ID, T34.LAST_UPD, T34. [...]
1542715800513830963	6fb88urfgyjsja	-2790.54 seconds	-100%	2790.54 seconds	0 seconds	(+) SELECT T34.CONFLICT_ID, T34.LAST_UPD, T34. [...]
11530247703233490999	8q1jjwp4u7pn6	-2547.13 seconds	-100%	2547.13 seconds	0 seconds	(+) SELECT /*+ BEGIN_OUTLINE_DATA IGNORE_OPTIM_EMBED [...]
17598323506657830469	5kcd99qf8y6yh	2386.13 seconds	6861.99%	34.77 seconds	2420.9 seconds	(+) SELECT T9.CONFLICT_ID, T9.LAST_UPD, T9.CRE [...]

5.9 hr

0.6 hr

Next Step: Tune Regressed SQL Statements using SQL Plan Baselines & Replay workload

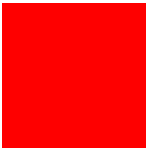
The Culprit!

```
SELECT T9.CONFLICT_ID, T9.LAST_UPD, T9.CREATED, T9.LAST_UPD_BY, T9.CREATED_BY, T9.MODIFICATION_NUM,
T9.ROW_ID, T4.KEY_VALUE, T25.PR_DEPT_OU_ID, T1.INTEGRATION_ID, T1.PRTNR_FLG, T25.PR_REGION_ID,
T25.CITIZENSHIP_CD, T6.ATTRIB_07, T25.EMAIL_ADDR, T25.EMP_FLG, T25.FAX_PH_NUM, T25.FST_NAME,
T25.CITIZENSHIP_CD, T25.HOME_PH_NUM, T25.AGENT_FLG, T25.JOB_TITLE, T25.LAST_NAME, T25.SEX_MF,
T25.MEMBER_FLG, T25.MID_NAME, T25.OWNER_PER_ID, T9.NAME, T25.PERSON_UID, T25.PRIV_FLG, T20.STATUS,
T8.PR_EMP_ID, T3.NAME, T25.CURR_PRI_LST_ID, T25.PR_OU_ADDR_ID, T1.NAME, T20.PR_ADDR_ID,
T25.PR_EMAIL_ADDR_ID, T25.PR_ALT_PH_NUM_ID, T25.PR_REP_DNRM_FLG, T25.PR_REP_MANL_FLG,
T25.PR_REP_SYS_FLG, T25.PR_MKT_SEG_ID, T22.PR_EMP_ID, T8.PR_EMP_ID, T13.LOGIN, T18.LOGIN, T17.PR_FAX_NUM_ID,
T25.PR_GRP_OU_ID, T25.PR_INDUST_ID, T25.PR_NOTE_ID, T25.PR_OPTY_ID, T25.BU_ID, T25.PR_SYNC_USER_ID,
T25.PR_PER_ADDR_ID, T25.PR_PER_PAY_PFL_ID, T25.PR_POSTN_ID, T25.PR_PROD_LN_ID, T25.PR_RESP_ID,
T17.PR_SMS_NUM_ID, T25.PR_SECURITY_ID, T5.NAME, T25.MED_SPEC_ID, T25.PR_STATE_LIC_ID, T25.PR_TERR_ID,
T25.PROVIDER_FLG, T12.OWN_INST_ID, T12.INTEGRATION_ID, T11.SHARE_HOME_PH_FLG, T25.CUST_SINCE_DT,
```

Next Steps:

- Tune Regressed SQL Statements using SQL Plan Baselines
- Replay workload

```
ORAPERF.S_ADDR_PER T23, ORAPERF.S_ORG_EXT_FNX T24, ORAPERF.S_CONTACT T25 WHERE T25.PR_DEPT_OU_ID =
T1.PAR_ROW_ID (+) AND T1.PR_POSTN_ID = T22.PAR_ROW_ID (+) AND T25.CURR_PRI_LST_ID = T3.ROW_ID (+) AND
T25.PR_POSTN_ID = T8.PAR_ROW_ID (+) AND T9.ROW_ID = T20.CON_ID (+) AND T20.POSTN_ID (+) = :1 AND T22.PR_EMP_ID
= T13.PAR_ROW_ID (+) AND T8.PR_EMP_ID = T18.PAR_ROW_ID (+) AND T25.PR_PER_ADDR_ID = T23.ROW_ID (+) AND
T25.MED_SPEC_ID = T5.ROW_ID (+) AND T9.ROW_ID = T4.CONTACT_ID (+) AND T9.ROW_ID = T25.PAR_ROW_ID AND
T9.ROW_ID = T17.PAR_ROW_ID (+) AND T9.ROW_ID = T11.PAR_ROW_ID (+) AND T9.ROW_ID = T6.PAR_ROW_ID (+) AND
T9.ROW_ID = T12.PAR_ROW_ID (+) AND T19.POSTN_ID = :2 AND T25.ROW_ID = T19.CON_ID AND T16.ROW_ID =
T19.POSTN_ID AND T19.POSTN_ID = T7.PAR_ROW_ID (+) AND T7.PR_EMP_ID = T15.PAR_ROW_ID (+) AND
T25.PR_DEPT_OU_ID = T2.ROW_ID (+) AND T25.PR_DEPT_OU_ID = T21.PAR_ROW_ID (+) AND T25.PR_DEPT_OU_ID =
T24.PAR_ROW_ID (+) AND T25.PR_SYNC_USER_ID = T10.ROW_ID (+) AND T25.PR_SYNC_USER_ID = T14.PAR_ROW_ID (+)
AND ((T25.PRIV_FLG = 'N') AND (T19.CON_LAST_NAME >= :3)) AND (T9.ROW_ID IN ( SELECT SQ1_T1.PERSON_ID FROM
ORAPERF.S_PARTY_PER SQ1_T1, ORAPERF.S_PARTY SQ1_T2, ORAPERF.S_ORG_EXT SQL_T3 WHERE (SQ1_T2.ROW_ID =
SQL_T3.PAR_ROW_ID AND SQ1_T1.PARTY_ID = SQ1_T2.ROW_ID) AND ((SQ1_T3.INT_ORG_FLG != 'Y' OR SQ1_T3.PARTNR_FLG
!= 'N') AND SQ1_T3.ACCNT_FLG != 'N') AND (SQ1_T3.NAME LIKE :4))) ORDER BY T19.POSTN_ID, T19.CON_LAST_NAME,
T19.CON_FST_NAME
```

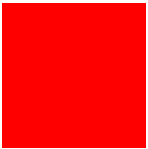


Replay Period Report: Validate Tuning

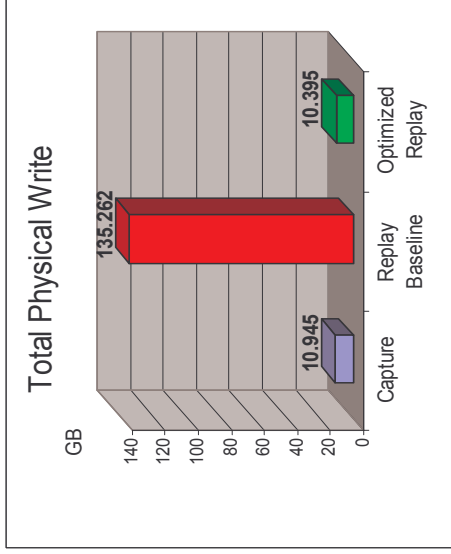
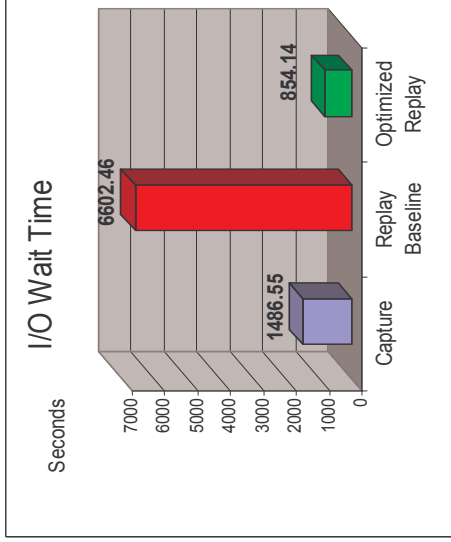
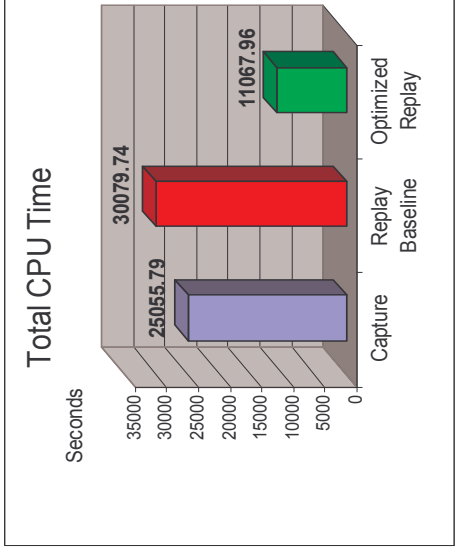
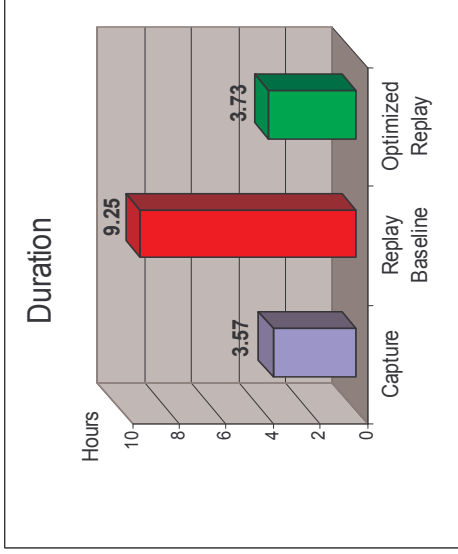
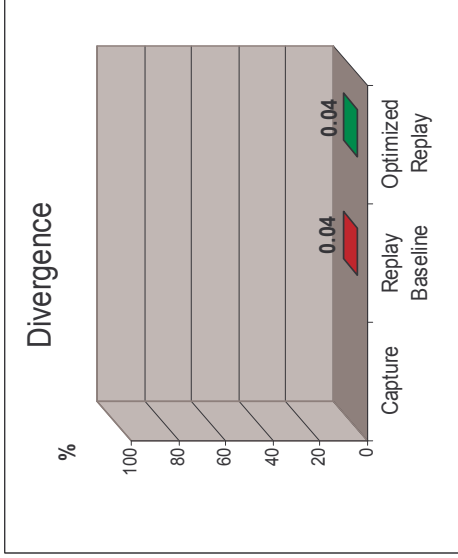
Force Matching Signature	example SQL_ID	Change in DB Time	Change in Average Response Time	1st Replay DB time	2nd Replay DB time	example sql text
9011963922445766669	44u0kwtwmd7h5	-21318.98 seconds	-99.13%	21505.14 seconds	186.16 seconds	(+) SELECT T9.CONFLICT_ID, T9.LAST_UPD, T9.CRE [...]
17598323506657830469	5kcd99qf8y6yh	-2366.15 seconds	-97.74%	2420.9 seconds	54.75 seconds	(+) SELECT T9.CONFLICT_ID, T9.LAST_UPD, T9.CRE [...]

**Savings:
5.9 hr**

**Savings:
0.6 hr**



Replay Compare Period: Replay Baseline Vs Optimized Replay



[Compare Period Report Link](#)

10.2 → 11g Upgrade: Summary



- Very low divergence rate, limited to background monitoring activity (EM)
- After 10.2 → 11g DB Upgrade, performance was almost same as before!
- Further tuning can be performed or other new features can be added one at a time....



Using Database Replay for RAC related Testing



Single Instance (SI) → RAC

Scenario:

Problem: The CPU utilization on the DB server is close to maximum limit, need to add instances (SI → 2-Node RAC) to handle increased month-end load. How do I test if my RAC configuration can handle the load?

Goal:

Assess impact the SI → RAC conversion for the peak workload & make sure are no negative effects due to the change



Using Database Replay for RAC Testing

- Database Replay handles all RAC related use cases without additional complexity
 - Simple case: Single Instance -> RAC
 - Advanced case: M-Instance RAC → N-Instance RAC
 - Interconnect changes (e.g., Infiniband)
- Database Replay + RAC considerations
 - Workload capture
 - Workload capture location
 - Connection mapping
 - Reporting and performance analysis

RAC Considerations: Workload Capture and Location

- **Workload capture**
 - For 10.2.0.4 and above DB releases: Automatic capture on all instances using MMON infrastructure
 - For 9.2.0.8, 10.2.0.2 and 10.2.0.3: Manually start capture on all instances
- **Workload capture directory location**
 - Both shared and local file system supported
 - Shared FS:
 - One directory across all instances captures entire workload
 - Recommended for ease of use
 - Local FS
 - Every node has separate capture directory
 - Directory name and path should be same
- **Workload replay directory location:**
 - All workload files must be consolidated into **single directory** when capture using shared/local FS



Workload Replay – Connection Remapping

- Database Replay allows connection remapping to test system using any of the following:
 - Explicit 1-1 connection remapping
 - Single connect descriptor
 - Single TNS net service name
- If already using Service Load Balancing (SLB), added node belongs to a service, connection mapping happens transparently
- If workload or application partitioning is used, explicit 1-1 mapping for every connection, e.g.,
 - Instance 1,2,3 :OLTP, Instance 4,5: DSS
- Make sure to remap every connection to test system setting
 - For API check using the following query:

```
SELECT replay_id ,conn_id ,capture_conn ,replay_conn FROM  
DBA_WORKLOAD_CONNECTION_MAP where replay_id = <value> order by 2
```

Workload Replay – Connection Remapping using EM interface

Replay Workload: Customize Options

Database ORCL
Capture Name wr-20091008-092801
Logged In As SYSTEM

Cancel Back Step 2 of 5 Next

Connection Mappings [Replay Parameters](#)

Replay Clients must establish connections to the replay database. Specify connection details to the replay database using either a single connect descriptor or net service name. Optionally, you can map every captured connect descriptor to a separate connect descriptor or net service name for the replay database.

TIP Connections must point to the replay database for a successful replay.

Use a single connect descriptor for all client connections. [Test Connection](#)

```
(DESCRIPTION = (ADDRESS_LIST = (LOAD_BALANCE = ON)(ADDRESS = (PROTOCOL = TCP)(HOST = rac1-vip)(PORT = 1521))  
(ADDRESS = (PROTOCOL = TCP)(HOST = rac2-vip)(PORT = 1521)))(CONNECT_DATA = (SERVICE_NAME = ORCL)))
```

Use a single TNS net service name for all client connections.

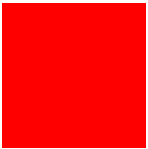
TIP All Replay Clients must be able to resolve the net service name (for example through a local tnsnames.ora file).

Use a separate connect descriptor or net service name for each client connect descriptor captured in the workload.

1

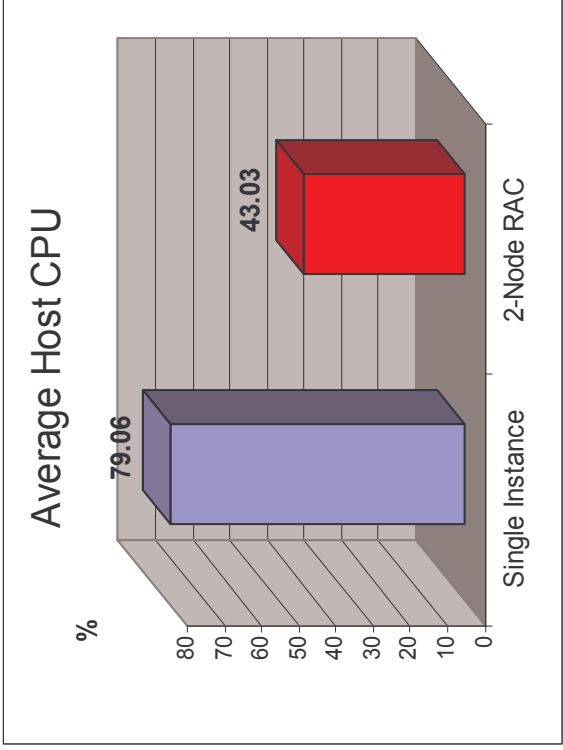
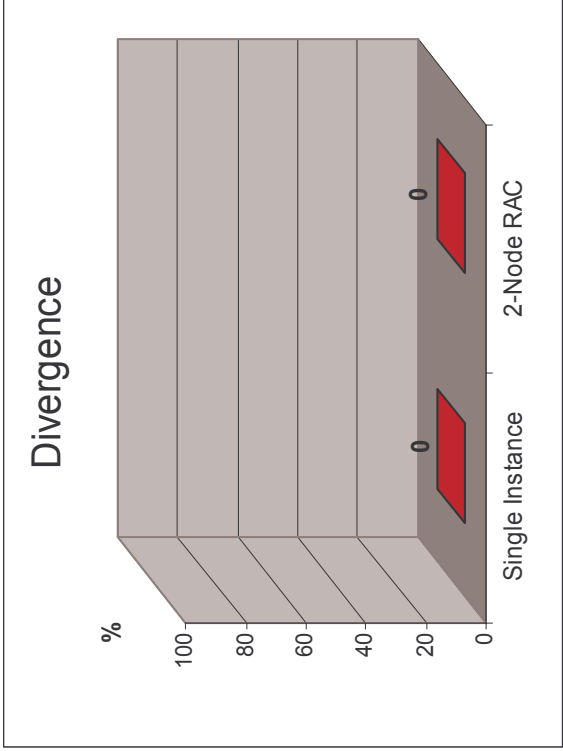
2

3



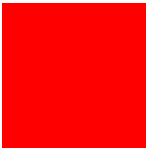
Single Instance (SI) → 2-Node RAC

Replay Baseline on SI Vs Replay on 2-Node RAC



- Average CPU utilization per host is reduced, it is split equally on both nodes
- Compare Period Report





Using Database Replay To Platform Migration Testing



Platform Migration

Scenario:

Problem: As part of our corporate standard push to Linux I want to migrate our 11.1.0.7 Oracle databases to Linux platform. How can I accomplish this?

Goal:

Assess the impact of platform migration from Windows to Linux using the peak workload captured on production system & make sure there are no negative effects due to the change



Platform Migration: Background

- What does “same platform” mean?
 - Software is installed from same Oracle CD/DVD
 - Same platform_id in v\$database
- Migration can be to same /different 32-64 bit, OS, endian
- Migration considerations
 - Uptime/SLAs
 - Datatype support
 - 32/64bit, OS, Little/Big-endian
 - #schema objects, user vs system schemas
 - Other: Expertise level, staging space, EBS, etc.
- Migration Options
 - Data Guard
 - Transportable Tablespace (TTS): 8i: same-endian format
 - Cross-Platform TTS –10gR1: cross-endian
 - Transportable DB (10gR2+): For same endian format
 - Other: Datapump, Golden Gate, etc.

Platform Migration: Determine Options To Use...

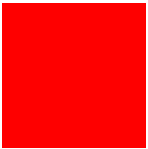
- For a given platform, supported method can be found using:
 - v\$database: platform_id, platform_name columns
 - v\$db_transportable_platform: Same-endian
 - v\$transportable_platform: Both endian
- See “Resources” slide at the end of the presentation
- Let’s assume that the migration strategy has been figured out, how do you test using real-workload?

For Microsoft Windows IA (32-bit),
v\$db_transportable_platform
output of

Platform Name	Platform Id
HP IA Open VMS	19
HP Open VMS	15
HP Tru64 UNIX	5
Linux IA (32-bit)	10
Linux IA (64-bit)	11
Linux x86 64-bit	13
Microsoft Windows IA (32-bit)	7
Microsoft Windows IA (64-bit)	8
Microsoft Windows x86 64-bit	12
Solaris Operating System (x86)	17
Solaris Operating System (x86-64)	20

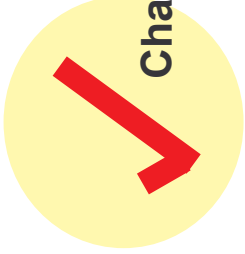
Platform Migration Testing With Database Replay

- Adhere to the “golden” rule of one change at a time
- Use SPA first for testing of SQL response time, followed by Database Replay for load testing
- Database Replay workload files are platform independent
- AWR data is platform-independent
 - Transported as before thru Datapump API
- Apply same performance tuning principles to platform migration testing
 - DB Time – Oracle’s single currency for measuring performance

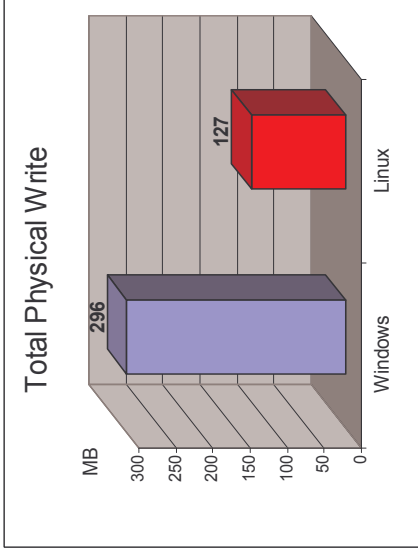
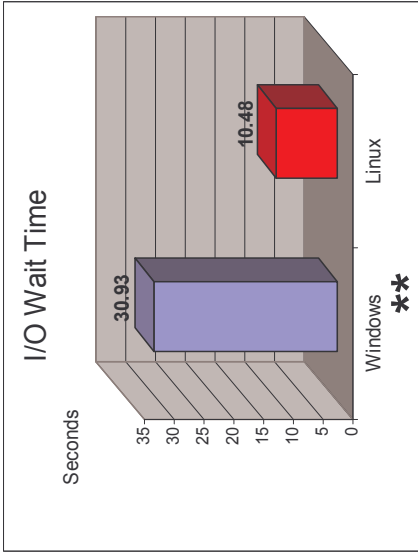
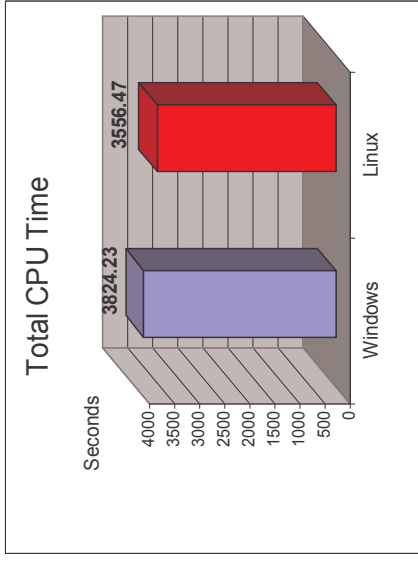
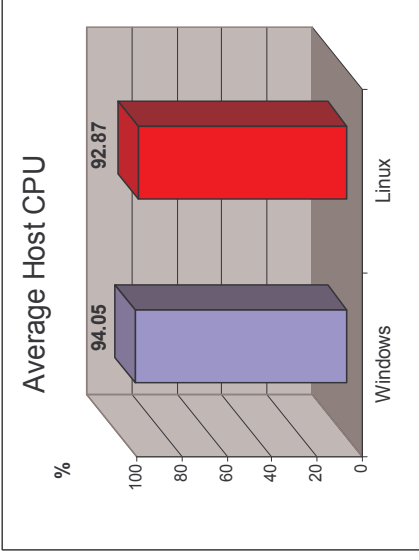
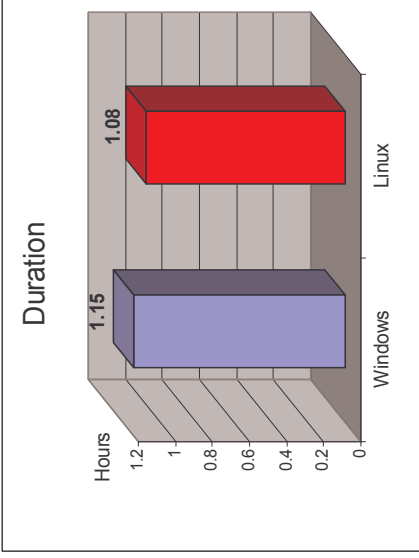
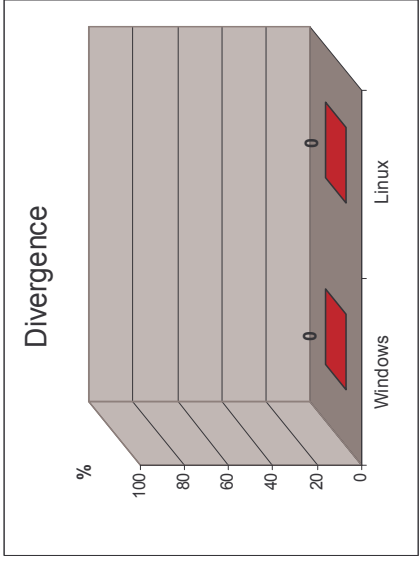


Platform Migration: Windows → Linux

Replay Baseline on Windows Vs Replay on Linux

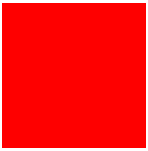


Change Accepted



ORACLE

** Note: I/O Wait Time is insignificant (less than 1% of Total Elapsed Time)



SQL Performance Analyzer (SPA)

SPA Motivation

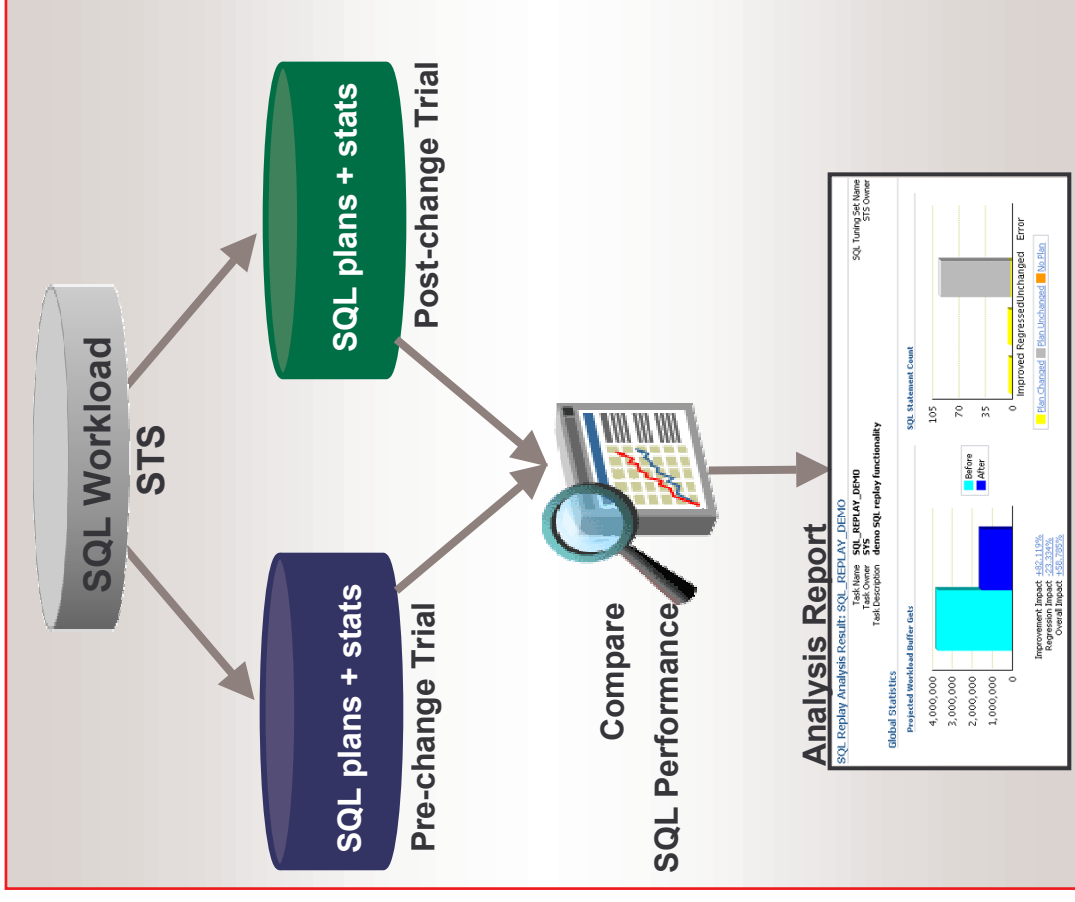
- Businesses need to adapt to changes to stay competitive, compliant and evolve
 - DB upgrades, schema, optimizer statistics refresh
 - SQL performance regressions: #1 cause of poor system perf.
- Current testing landscape and limitations
 - Expensive capture, partial workload, non-production optimizer context, binds
 - Large workloads (**100Ks** SQL stmts are common)
 - Manual and time consuming testing and regression tuning
 - No end-to-end testing solution
 - Test In Production is not too uncommon

SQL Performance Analyzer (SPA)

- Proactively detects ALL SQL regressions, BEFORE actual change is deployed
- Integrated comprehensive solution for end-to-end SQL workload testing

SPA Overview

- Helps users predict the impact of system changes on SQL workload response time
- Low overhead capture of SQL workload to SQL Tuning Set (STS) on production system
- Build different SQL trials (experiments) of SQL statements performance by test execution
- Analyzes performance differences
- Offers fine-grained performance analysis on individual SQL
- Integrated with STS, SQL Plan Baselines, & SQL Tuning Advisor to form an end-to-end solution





SPA: Common Usage Scenarios

- Database upgrades and patch-set releases
 - 9.2/10.1 → 10.2 or 11g releases
 - 10.2.0.x → 10.2.0.y or 11g releases
- Optimizer statistics refresh
- Database parameter changes
- Database schema changes (e.g., add/drop indexes)
- Implementation of tuning recommendations
- I/O subsystem changes (e.g., ASM, Exadata)

SPA can be used for:

- any change that affects SQL execution plan & performance
- in production as well as test environments

SPA: Enterprise Manager Interface

- Rich GUI through Enterprise Manager – New workflows added!
- DBMS_SQLPA package PL/SQL API



Database Instance: orclj.us.oracle.com

[Home](#) [Performance](#) [Availability](#) [Server](#) [Schema](#) [Data Movement](#) [Software and Support](#)

Software

[Configuration](#)

[Search](#)

[Last Collected Configuration](#)

[Collection Status](#)

[Clone Oracle Home](#)

[Host Configuration](#)

[Oracle Home Inventory](#)

[Real Application Testing](#)

[Database Replay](#)

[SQL Performance Analyzer](#)



SQL Performance Analyzer

Page Refreshed **Aug 27, 2009 3:55:26 PM PDT** [Refresh](#) [View Data](#) [Real Time: 15 Second Refresh](#)

SQL Performance Analyzer allows you to test and to analyze the effects of changes on the execution performance of SQL contained in a SQL Tuning Set.

SQL Performance Analyzer Workflows

Create and execute SQL Performance Analyzer Task experiments of different types using the following links.

[Upgrade from 9i or 10.1](#)

Test and analyze the effects of database upgrade from 9i or 10.1 on SQL Tuning Set performance.

[Upgrade from 10.2 or 11g](#)

Test and analyze the effects of database upgrade from 10.2 or 11g on SQL Tuning Set performance.

[Parameter Change](#)

Test and compare an initialization parameter change on SQL Tuning Set performance.

[Exadata Simulation](#)

Simulate the effects of a Exadata Storage Server installation on SQL Tuning Set performance.

[Guided Workflow](#)

Create a SQL Performance Analyzer Task and execute custom experiments using manually created SQL trials.

SQL Performance Analyzer Tasks

[Delete](#) [View Latest Report](#)

[Previous](#) [Next 1](#)

Select Name	Owner	Last Modified	Current Step Name	Type	Status	SQLs Processed	Steps Completed
54_SPA	SYS	Aug 27, 2009	COMPARE_1251402338324	Compare	Completed	1 of 1	1 of 1

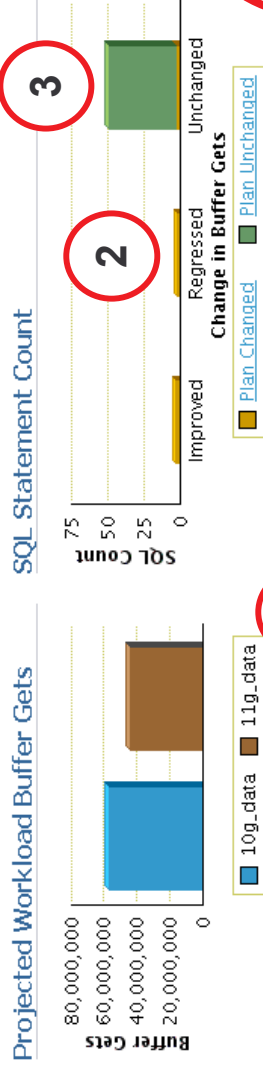
SPA Report

SQL Performance Analyzer Task Result: [SYS.UPGRADE_10G11G](#)

Task Name [UPGRADE_10G11G](#) SQL Tuning Set Name [OOW_54G](#)
 Task Owner [SYS](#) STS Owner [SYS](#)
 Task Description [test upgrade to 11g](#) Total SQL Statements [54](#) Comparison Metric [Buffer Gets](#)
 SQL Statements With Errors [0](#)

Replay Trial 1 [10g_data](#)
 Replay Trial 2 [11g_data](#)

Global Statistics



Improvement Impact [24%](#) ↑
 Regression Impact [-2%](#) ↓

Overall Impact [22%](#) ↑

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](#)

Top 10 SQL Statements Based on Impact on Workload

SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	Plan
		10g_data	11g_data		
g4dzf4ak4rus2	12.000	20,318,458.000	13,502,097.000	33.550	30.670 Y
gfacm5jr3rz9j	11.990	6,990,541.000	180,401.000	97.420	0.410 Y
2ny751aat2vd9	-0.820	12,973,052.000	13,440,825.000	-3.610	30.530 Y
c2fb0ug5p7d4p	-0.750	12,740,524.000	13,165,998.000	-3.340	29.910 Y
2wtgxbjz6u2by	0.050	244,678.000	218,533.000	10.690	0.500 Y

SPA Report

Regressed SQL Statements

Regressed SQL Statements

SQL ID	Net Impact on Workload (%)		Buffer Gets		Net Impact on SQL (%)		% of Workload		Plan Changed
	10g_data	11g_data	10g_data	11g_data	10g_data	11g_data	10g_data	11g_data	
2ny751aat2vd9	-0.820	12,973,052.000	13,440,825.000		-3.610	22.850	30.530	Y	
c2fb0u...	-0.750	12,...							

SQL Details: 2ny751aat2vd9
 Parsing Schema: DWH_TEST Execution Frequency: 1
 (Schedule SQL Tuning Advisor)

►SQL Text

Single Execution Statistics

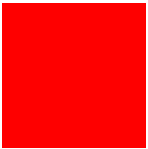
Execution Statistic Name	Net Impact on Workload (%)	Execution Statistic Collected		Net Impact on SQL (%)	
		10g_data	11g_data	10g_data	11g_data
Elapsed Time	-4.340	70.518	89.593	-27.050	16.060
Parse Time	-13.830	0.207	0.312	-50.720	27.270
CPU Time	-5.700	64.704	85.188	-31.660	18.010
Buffer Gets	-0.820	12,973,052.000	13,440,825.000	-3.610	22.850
Optimizer Cost	0.170	982.000	658.000	32.990	0.530
Disk Reads	10.800	7,011.000	5.000	99.930	10.810
Direct Writes	10.950	6,968.000	0.000	100.000	10.950
Rows Processed	0.000	111.000	111.000	0.000	0.000

Plan Comparison

10g_data
 Plan Hash Value: 393503022

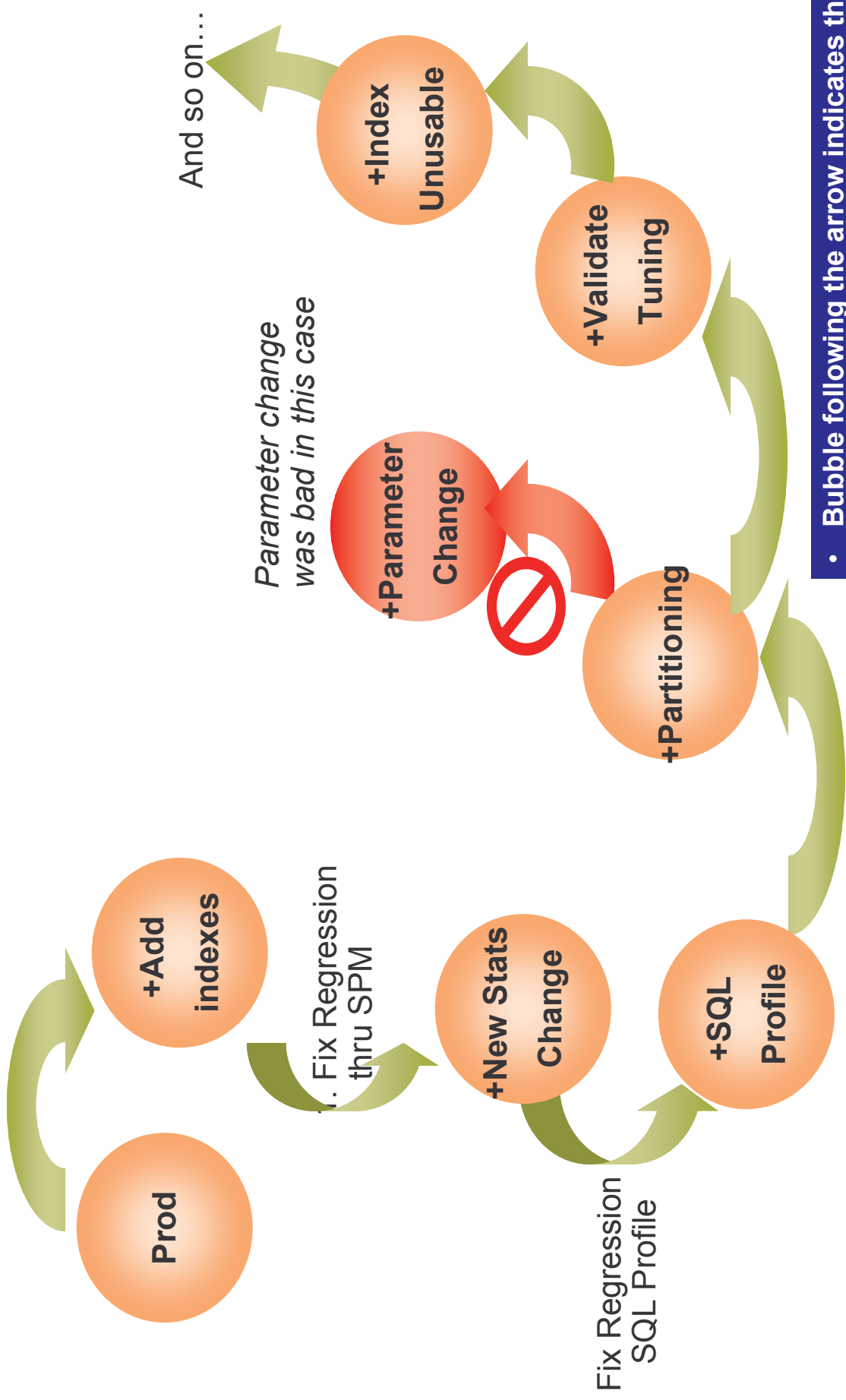
Expand All | Collapse All

Operation	Line ID	Object	Rows	Cost
SELECT STATEMENT	0		1	967
HASH	1		1	967
TABLE ACCESS	2	FACT_PD_OUT_ITM_293	1	966
NESTED LOOPS	3		1	966
MERGE JOIN	4		1	320
SORT	5		90	315
TABLE ACCESS	6	ADM_PG_FEATUREVALUE	1	2
NESTED LOOPS	7		00	314



Usage Scenario: Evaluating Changes On Production

Using SPA For Changes in Production: Example



- Bubble following the arrow indicates the delta change on Production
- SPA is used for testing every change

How to Minimize Impact on Production?

- Generate Plan Vs Test Execute
 - Use Generate Plan Trial Method to subset SQL with plan changes
 - Only test execute SQL with plan changes
- Limit testing scope to **private** session or schema where possible
 - Use alter session set `<parameter> = <value>`; (Vs system)
 - Example usage for SQL Profiles:
 - alter session set sqtune_category= 'TEST';
 - *Implement SQL Profiles and test Only sessions with "TEST" sqtune_category see these Profiles - private scope!!*
 - alter session set sqtune_category= 'DEFAULT'; -- Now SQL Profiles visible **globally to all sessions**
 - Similarly for Invisible Indexes, Pending Stats
- Use SPA time limit to control resource usage
- Test during maintenance window or non-peak activity when spare resources are available



Using SPA on Production: Evaluating Optimizer Statistics Refresh and other changes

Scenario:

Can I use SPA to check if any SQL statements regressed due to optimizer statistics refresh on my 10.2/11g production databases? If so, how?

Goal:

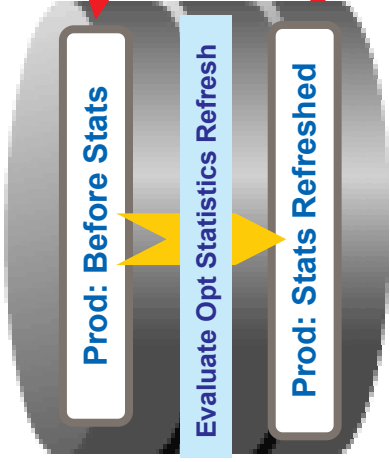
Assess impact of optimizer statistics gathering on SQL workload performance on production system using SPA & make sure are no negative effects of the change

Evaluating Optimizer Statistics Refresh

- Assumptions
 - Statistics refreshed periodically through custom jobs
 - No test system available, so evaluation is done on production
 - Should impact end-users minimally
 - Test using optimizer pending statistics feature
- Use SPA remote trials capability (generate plan and test execute) to evaluate statistics refresh on 10.2/11g production database
- Analyze SPA report and take appropriate action
 - Overall improvement but few SQL regressions
 - Solution: Use SQL Profiles or Plan Management
 - No improvement and many regressions
 - Solution: Revert to old statistics: Use optimizer statistics retention/history feature. Alternatively, configure optimizer statistics appropriately*
 - For Oracle Database 11g, “publish pending” statistics after evaluation of statistics

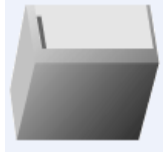
Using SPA on Production: Evaluating Optimizer Statistics Refresh

Production Database (10.2/11g)



No Test Database Used!!

11g SPA System



- No app schema/data necessary
- Not Mandatory for 11g db
- Repository for many tests!

Remotely Build SQL Trials

1. Capture SQL workload to STS

2. Import STS

3. Gather Pending Stats

4. Use SPA to detect performance changes

5. Publish Pending Stats and Remediate Regressions



Conclusion

- Real Application Testing provides comprehensive and easy-to-use solution for testing database infrastructure
- Real Application Testing can be used for testing many changes both test and production environments
- Helps adopt technology faster by significantly cutting down testing costs and production deployment risk
- Helps businesses can be stay competitive and improve profitability