Oracle Database 11g
Real Application Testing
Real Application Testing

- Value
  - Rapid technology adoption
  - Higher testing quality
- Business Benefit
  - Lower cost
  - Lower risk

Solution for the Agile Business
Database Replay
The Need for Database Replay

- Businesses want to adopt new technology that adds value
- Extensive testing and validation is expensive in time and cost
- Despite expensive testing success rate low
  - Many issues go undetected
  - System availability and performance negatively impacted
- Cause of low success rate
  - Current tools provide inadequate testing
    - Simulate synthetic workload instead of replaying actual production workload
    - Provide partial workflow coverage

Database Replay makes real-world testing possible
Database Replay

• Replay actual production database workload in test environment
• Identify, analyze and fix potential instabilities before making changes to production

• 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
  • Honor commit ordering

• Analyze & Report
  • Errors
  • Data divergence
  • Performance divergence
Database Replay: Supported Changes

Changes Supported
- Database Upgrades, Patches
  - Schema, Parameters
  - RAC nodes, Interconnect
- OS Platforms, OS Upgrades
  - CPU, Memory
  - Storage
  - Etc.

Recording of External Client Requests
Comparison of LoadRunner & DB Replay Testing e-Business Suite

<table>
<thead>
<tr>
<th>Process</th>
<th>DB Replay</th>
<th>LoadRunner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install &amp; Setup</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>Understand Application</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Usage</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Identify Key Transactions</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>Generate Workload</td>
<td>5</td>
<td>80</td>
</tr>
<tr>
<td>Run Test</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Total Testing Time

- DB Replay: 2 weeks
- LoadRunner: 30 weeks
Why DB Replay?

**From:**
- Artificial workloads
- Partial workflows
- Months of development
- Manual intensive
- High risk

**To:**
- Production workloads
- Complete workflows
- Days of development
- Automated
- Low risk

150 Days → 10 Days
Database Replay Workflow

Production (10.2.0.4)
- Clients
- Mid-Tier
- Storage

Process
- Capture
- Storage

Test (11.1)
- Replay Driver
- Storage

Replay
- Analysis & Reporting
Step 1: Workload Capture

- All external client requests captured in binary files
- System background, internal activity excluded
- Minimal performance overhead for capture
- For RAC, shared and local file system supported
- Specify interesting time period for capture, e.g., peak workload, month-end processing, etc.
- Can capture on 10.2.0.4 and replay on 11g
Capture Options

• Workload can be filtered to customize what is captured
  • Filter Types
    • Inclusion Filters: Specifies which sessions should be captured
    • Exclusion Filters: Specifies which sessions should NOT be captured
  • Filter Attributes: Workload capture can be filtered using any of the following session attributes
    • User
    • Program
    • Module
    • Action
    • Service
    • Session ID
  • Workload capture can be run on-demand or scheduled to run at later time
Step 2: Process Workload Files

- Setup test system
  - Test DB is at same point in time as before production capture
  - Use RMAN to physically restore production db from backup
  - Use Snapshot standby
  - Use imp/exp, Data Pump, etc.
- Processing transforms captured data into replayable format
- Once processed, workload can be replayed many times
- For RAC copy all capture files to single location for processing
Step 3: Replay Workload

- Replays workload preserving timing, concurrency and dependencies of the capture system.

- Replay Driver is a special client program that consumes processed workload and sends requests to the replay system.

- Replay Driver consists of one or more clients. For workloads with high concurrency, it may be necessary to start multiple clients to drive workload.
Replay Options

• Synchronized Replay (Default)
  • Workload is replayed in full synchronized mode
  • Exact same concurrency and timing as production workload
  • Transaction commit order is honored
  • Ensures minimal data divergence

• Unsynchronized Replay
  • Workload can be replayed in unsynchronized mode
  • Useful for load/stress testing
  • High data divergence
  • Three (3) parameters provided to control degree of synchronization
    • Think time synchronization
    • Connect (logon) time synchronization
    • Commit order synchronization
Replay Options

- Unsynchronized Replay Parameters
  - Think time synchronization
    - Controls think time between database calls
    - Auto (Default): Adjusts think time so as to maintain captured request rate
    - Percentage
      - 0% No think time, highest possible request rate
      - <100% Higher request rate
      - 100% Exact think time
      - >100% Lower request rate
  - Connect (logon) time synchronization
    - Controls when sessions are created
    - 0%: All sessions are connected immediately
    - 100% (Default): Sessions connect at same time as in captured system
  - Commit order synchronization
    - Controls commit order between transactions
    - In asynchronous mode, commit order not honored – transactions are committed as soon as commit call is issued
Replay Options

• Number of Replay Clients
  • Configurable by user
  • Client Calibration Advisor recommends number of replay clients needed for specific workload
  • Replay clients are multithreaded clients that can drive multiple workload sessions each
Analysis & Reporting

• Comprehensive reports are provided for analysis purposes
• There (3) types of divergences are reported
  • Data Divergence: Number of rows returned by each call are compared and divergences reported
  • Error Divergence: For each call error divergence is reported
    • New: Error encountered during replay not seen during capture
    • Not Found: Error encountered during capture not seen during replay
    • Mutated: Different error produced in replay than during capture
  • Performance Divergence
    • Capture and Replay Report: Provides high-level performance information
    • ADDM Report: Provides in-depth performance analysis
    • AWR, ASH Report: Facilitates comparative or skew analysis
Database Replay Summary Report

View Workload Replay: REPLAY3

Summary

- **Replay Name**: REPLAY3
- **Directory Object**: GP1 1D
- **Database Name**: V12
- **DEID**: 1623005897
- **Replay Error Code**: N/A
- **Replay Error Message**: None

**Capture Name**: CAPTURE3
- **Duration (HH:MM:SS)**: 00:00:02
- **Prepare Time**
  - Apr 13, 2007 4:16:46 PM PDT
- **Start Time**
  - Apr 13, 2007 4:46:32 PM PDT
- **End Time**
  - Apr 13, 2007 4:51:34 PM PDT

**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, then 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.
Current Restrictions

- Database Replay does not support the following features in the current release
  - Direct path load, import/export
  - OCI based object navigation (ADTs) and REF binds
  - Streams, non-PL/SQL based AQ
  - Distributed txns, remote describe/commit operations
  - Flashback
  - Shared Server
Best Practices

- **Capture**
  - Provide adequate disk space for captured workload (binary files)
  - Database restart (Optional): Recommended to minimize divergence
  - For RAC, use shared file system
- **Test System Setup**
  - Ensure data in test is identical to production as of capture start time to minimize data divergence during replay
  - Use RMAN backup/restore or Snapshot Standby to setup test system
  - For performance analysis test system capacity should be similar to production
  - Reset system clock to same time as production if application logic involves SYSDATE usage
- **Process Workload**
  - Processing workload has performance overhead and can possibly take a long time
  - Process workload on test system instead of production
- **Replay**
  - Use Client Calibration Advisor to identify number of replay clients needed to replay workload properly
SQL Performance Analyzer (SPA)
The Need for SQL Performance Analyzer (SPA)

- Businesses want systems that are performant and meet SLA’s.
- SQL performance regressions are #1 cause of poor system performance.
- Solution for proactively detecting *all* SQL regressions resulting from changes not available.
- DBA’s use ineffective and time-consuming manual scripts to identify problems.

SPA identifies all changes in SQL performance before impacting users.
Why SQL Performance Analyzer?

From:
- Manual workload creation
- Synthetic workload
- Months of manual analysis
- Partial workload
- High risk

To:
- Automated workload capture
- Production workload
- Automated analysis in minutes
- Complete workload
- Low risk
SQL Performance Analyzer

- Test impact of change on SQL query performance
- Capture SQL workload in production including statistics & bind variables
- Re-execute SQL queries in test environment
- Analyze performance changes – improvements and regressions

Production

Test

Re-execute SQL Queries

Use SQL Tuning Advisor to tune regression
SPA Benefits

- Enables identification of SQL performance regressions before end-users can be impacted

- SPA can help with any change that impacts SQL execution plan
  - DB upgrades
  - Optimizer statistics refresh
  - New indexes, Materialized Views, Partitions, etc.

- Automates SQL performance tracking of hundreds of thousands of SQL statements – impossible to do manually

- Captures SQL workload with low overhead

- Integrated with SQL Tuning Advisor and SQL Plan Baselines for regression remediation
SQL Performance Analyzer Workflow

Production (10.2)

Clients

Mid-Tier

Storage

Test (11.1)

Storage

Capture SQL

Transport SQL

Execute SQL Pre-change

Execute SQL Post-change

Compare Perf

Clients

Mid-Tier

Storage

Capture SQL

Transport SQL

Execute SQL Pre-change

Execute SQL Post-change

Compare Perf

ORACLE
Step 1: Capture SQL Workload

- SQL Tuning Set (STS) used to store SQL workload
- STS includes:
  - SQL Text
  - Bind variables
  - Execution plans
  - Execution statistics
- Incremental capture used to populate STS from cursor cache over a time period
- SQL tuning set’s filtering and ranking capabilities filters out undesirable SQL
- SQL workload captured in 10.2.0.1 and higher can be used for SPA tasks in 11g
Step 2: Move SQL Workload to Test System

- Copy SQL tuning set to staging table ("pack")
- Transport staging table to test system (datapump, db link, etc.)
- Copy SQL tuning set from staging table ("unpack")
Step 3: Execute SQL Before Making Change

- Establishes SQL workload performance baseline
- SQL execution plan and statistics captured
- SQL executed serially (no concurrency)
- Each SQL executed only once
- DDL/DML skipped
- Option to do Explain Plan only analysis
Step 4: Execute SQL After Making Change

- **Manually implement the planned change**
  - Database upgrade, patches
  - Optimizer statistics refresh
  - Schema changes
  - Database parameter changes
  - Tuning actions, e.g., SQL Profile creation

- **Re-execute SQL after change**
  - Gathers new SQL execution plans and statistics
Step 5: Compare & Analyze Performance

- Compare performance using different metrics, e.g.,
  - Elapsed Time
  - CPU Time
  - Optimizer Cost
  - Buffer Gets

- SPA Report shows impact of change for each SQL
  - Improved SQL
  - Regressed SQL
  - Unchanged SQL

- Fix regressed SQL using SQL Tuning Advisor or SQL Plan Baselines
# SPA Report

**SQL Performance Analyzer Task Result: SYS.SYSTEMCHANGES1**

<table>
<thead>
<tr>
<th>Task Name</th>
<th>SQL Tuning Set Name</th>
<th>Total SQL Statements</th>
<th>SQL Statements With Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEMCHANGES1</td>
<td>HR_WORKLOAD</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>Task Owner</td>
<td>STS Owner</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS</td>
<td>APPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task Description</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Global Statistics**

- **Projected Workload Execute Elapsed Time**
  - Before
  - After

- **SQL Statement Count**
  - Improved
  - Regressed
  - Unchanged

- **Change in Execute Elapsed Time**
  - **Plan Changed**
  - **Plan Unchanged**

**Recommendations**

- Run SQL Tuning Advisor to tune regressed SQL statements.
- Schedule SQL Tuning Advisor

**Improvement Impact**

- 69%↑

**Regression Impact**

- 5%↓

**Overall Impact**

- 85%↑
SPA Report – Regressed SQL Statements

Regressed Plan Changed SQL Statements

<table>
<thead>
<tr>
<th>SQL_ID</th>
<th>Executions</th>
<th>Net Impact on Workload (%)</th>
<th>Execute Elapsed Time</th>
<th>Net Impact on SQL (%)</th>
<th>% of Workload</th>
</tr>
</thead>
<tbody>
<tr>
<td>2wtgxbjz6u2by</td>
<td>1</td>
<td>-4.390</td>
<td>2.004</td>
<td>4.697</td>
<td>-134.380</td>
</tr>
<tr>
<td>fbp9za0h42km</td>
<td>1</td>
<td>-0.020</td>
<td>0.000</td>
<td>0.015</td>
<td>-1,500,000</td>
</tr>
<tr>
<td>654x8xs5w12</td>
<td>1</td>
<td>-0.020</td>
<td>0.000</td>
<td>0.013</td>
<td>-1,300,000</td>
</tr>
</tbody>
</table>

SQL Details: 2wtgxbjz6u2by

Execution Frequency: 1

Problem Findings:
The performance of this SQL has regressed.

Symptom Findings:
The structure of the SQL execution plan has changed.

Information Findings:

Plan Comparison:
Before_change
Plan Hash Value 1511732424

After_change
Plan Hash Value 35170111486

Oracle
Best Practices

• Workload Capture
  • Use incremental STS capture
  • Peak or representative workloads can be more accurately captured with this mechanism

• Test system
  • Run SPA on test system instead of production as analysis can be resource intensive
  • Ensure test system has similar configuration and comparable optimizer statistics as production

• Performance comparison
  • Use several different metrics, e.g., elapsed time, CPU time, etc., to compare pre- and post-change performance for reliable results

• Regression remediation
  • Use SQL Tuning Advisor and SQL Plan Baselines to fix regressions
Real Application Testing Summary

• Provides cost-effective and easy-to-use solution of assessing impact of change on production systems
  • Holistic and actual workload testing results in lower risk
  • Brings down testing cycle from months down to days
  • Lowers hardware costs by eliminating need for mid-tier and application setup on test system
  • Maximizes ROI by leveraging Diagnostics and Tuning Pack to remediate regressions

• With Real Application Testing businesses can
  • Stay competitive
  • Improve profitability
  • Be compliant