

Oracle Real Application Testing:

Testing the SQL Performance Impact of an Oracle Database Release 10.2.0.x to 10.2.0.y upgrade using SQL Performance Analyzer

prabhaker.gongloor@oracle.com

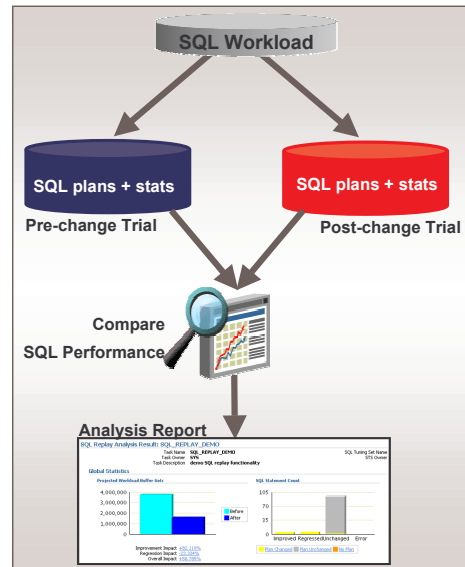
pete.belknap@oracle.com

mark.rivkin@oracle.com

ORACLE

SQL Performance Analyzer: Concepts

- Helps users predict the impact of system changes on SQL workload response time
- Builds different trials (experiment) of SQL workload performance (i.e., SQL execution plans and statistics)
- Analyzes performance differences
- Offers fine-grained performance analysis on individual SQL
- Integrated with SQL tuning set, SQL plan baselines, and SQL tuning advisor to form an End-to-end solution



ORACLE

The SQL Performance Analyzer (SPA) introduces the following key concepts:

- **SQL Tuning Set (STS):** the STS is a database object representing a SQL workload. Oracle contains facilities for populating and manage the STS making it easy to capture your SQL workload and provide it as input to different operations.
- **SPA task:** the analysis task is a container that encapsulates an entire experiment. In the case of an upgrade, we will create a single task to cover the entire upgrade experiment. SPA tasks are stored in the SYSAUX tablespace and available for inspection in the DBA_ADVISOR_XXX views. The primary input of a task is the STS that the experiment will operate upon.
- **SPA trial:** a SPA task contains multiple SPA trials, where each trial is a single version of the workload performance data. SPA can execute SQLs (using test-execute technology to avoid side-effects) and gather execution stats and execution plans or get execution plans only by simply performing an EXPLAIN PLAN on each SQL statement.

SQL Performance Analyzer: Concepts

- Notes Continued

ORACLE

- **SPA comparison:** SPA allows you to compare two trials to see what has changed; this is the performance analysis step when SPA examines each SQL and sees if, and how much, its performance has improved or regressed due to the change. You can perform an analysis using the performance metric of your choice. SPA will recommend running SQL Tuning Advisor or using SQL Plan Baselines to fix regressed SQL statements.
- **SPA report:** The user-consumable output of a SPA comparison is a SPA report, which shows all of the discoveries from the SPA analysis.

An experiment starts off with two SPA trials, one for before the change and one for after the change, followed by a SPA comparison to analyze the change. Further SPA trials and comparisons will be performed as you make additional changes to tune your performance.

Database Upgrade: 10.2.0.x → 10.2.0.y

Scenario:

One of the database I'm managing is on 10.2.0.2. How can I use 11g SQL Performance Analyzer (SPA) functionality to accomplish an 10.2.0.4 patchset upgrade?

Goal:

Assess impact of upgrade on SQL workload performance using SPA so that there are no surprises after upgrade.

ORACLE

Note that the content of this presentation applies to all upgrades with the prior release \geq 10.2.0.2

SPA 10.2.0.x -> 10.2.0.y Workflow Overview

- SPA 11.1.0.6 release allowed for testing any arbitrary change to the database, but only within 11g; upgrade testing relied on simulating previous versions of Oracle.
- To assist 10.2.0.x->10.2.0.y upgrades some Real Application Testing functionality has been backported to 11.1.0.7. The new technology executes SQLs remotely on a 10g system rather than trying to simulate 10g releases within 11g. It allows customers to support any upgrade from a prior release as early as 9i – this presentation focuses on the 10.2.0.x->10.2.0.y use case.
- This presentation discusses using the features in 11.1.0.7. They are also available in 11.1.0.6 with a patch (Metalink Note: 560977.1). Enterprise Manager support for these additional functionalities is not available in 11.1.0.6, but command line APIs are available.

ORACLE

The first release of SQL Performance Analyzer (SPA) was created for Oracle Database 11.1.0.6. It allowed you to take a set of SQL statements, called a SQL Tuning Set (STS), run it before and after making your desired changes and compare the results with a detailed report. This SPA allowed us to evaluate how any database changes will impact SQL performance. SPA suggests using SQL Tuning Advisor or SQL Plan Management to tune any regressed SQL. In the Oracle Database 11.1.0.7 release (and in a one-off patch atop 11.1.0.6, see metalink note 560977.1) we have added to the SPA upgrade testing scenario, allowing you to set up a small “SPA System” with just a SQL Tuning Set but no application schema nor data and use it to drive a test, connecting to a test system running the database release you will be upgrading from/to, executing the SQLs there, and sending the results back to the SPA system where they can be used for analysis. In particular, this enhancement is especially useful because it opens the door to testing upgrades to Oracle Database 10.2 releases.

In this presentation we will discuss only the upgrade from Oracle 10.2.0.x to Oracle 10.2.0.y. You can also use SQL Performance Analyzer to test upgrades from Oracle 9i/10.1 to Oracle 10.2 – these are discussed in our Oracle White Paper, titled “Oracle Real Application Testing: Testing the SQL Performance Impact of an Oracle 9i to Oracle Database 10g Release 2 using SQL Performance Analyzer”, available on OTN and linked at the end of this presentation.



SPA 10.2.0.x -> 10.2.0.y Workflow Overview

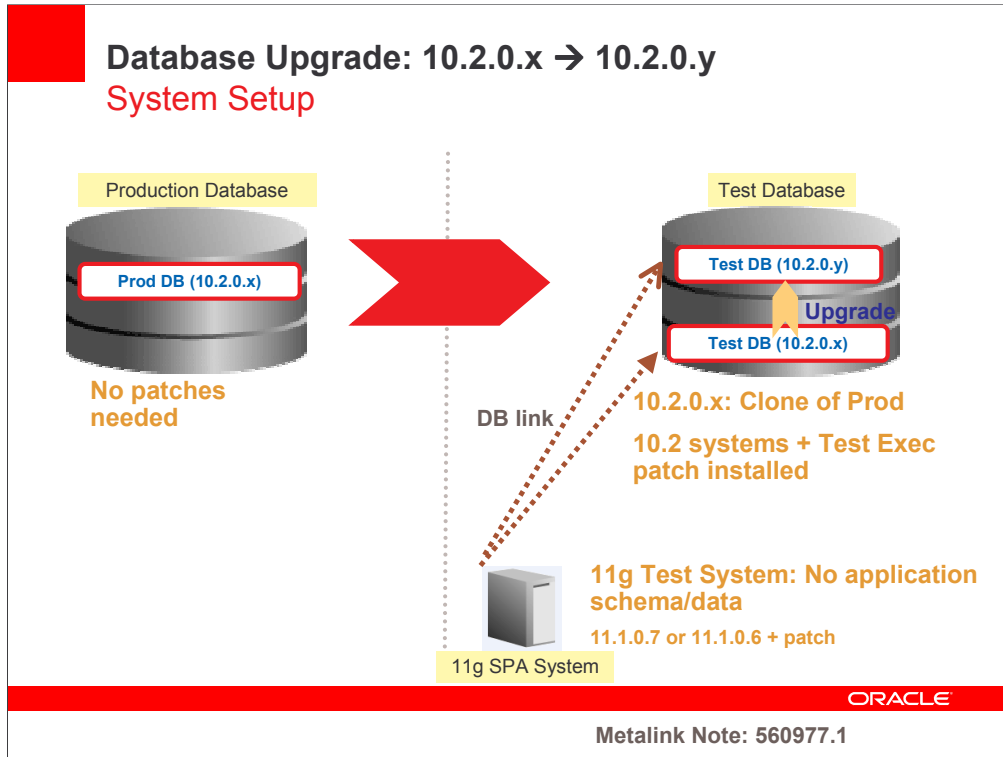
- Notes Continued



ORACLE

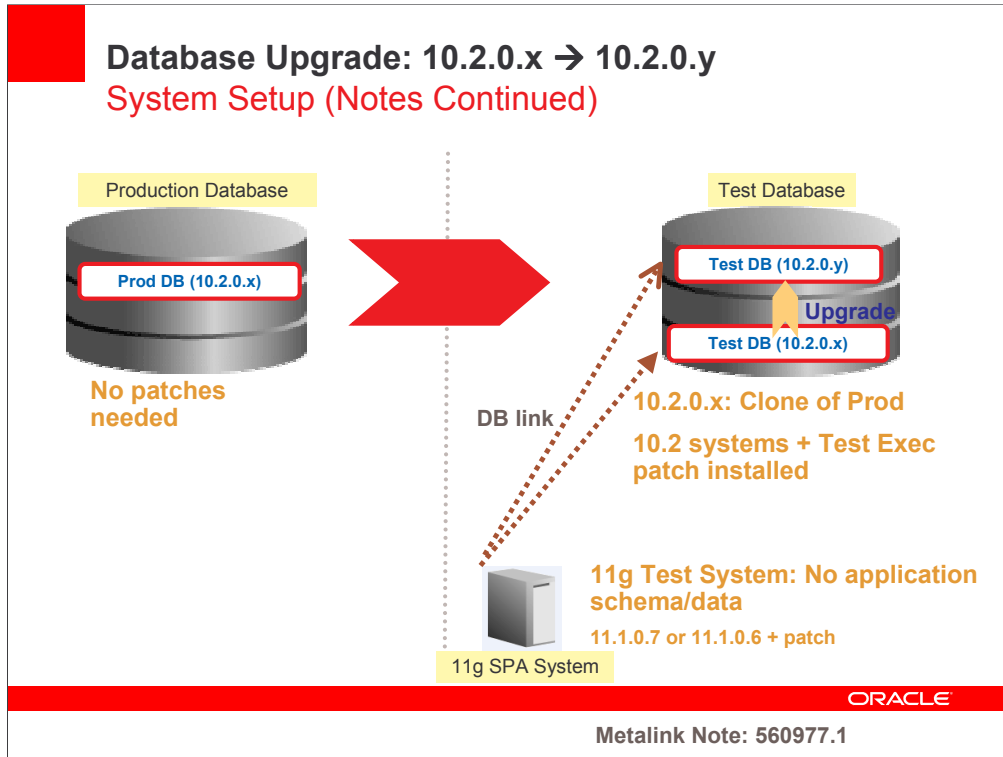
We do recommend reading over the white paper while noting the differences in the 10.2.0.x->10.2.0.y workflow described in this presentation – this workflow is much simpler than the 9i/10.1->10.2 workflow because of the additional manageability infrastructure available in the 10.2 database release to build on top of. Testing an upgrade from 10.2 to 11g can be done with the same flow as described in this presentation.

This presentation assumes testing using the 11.1.0.7 database release, as it contains Enterprise Manager support to simplify the workflow. Testing with the 11.1.0.6 one-off patch can be done with the identical workflow, but only from the command line APIs. Here we show only examples using enterprise manager, but the command-line APIs are described in the White Paper (linked at the end of the presentation) and made easier to use by some SQL*Plus scripts we have provided (linked from Metalink note 560977.1).



The basic components of this experiment will be:

- The production system, on release 10.2.0.x: no patch is needed. Here we will capture the SQL workload into a SQL Tuning Set. No actual testing will be run on the production system.
- A test system with a full clone of production schema+data (or as similar as possible) on 10.2.0.x: we will use SPA to test-execute the workload here once on 10.2.0.x, then upgrade the system to 10.2.0.y, and perform a second test-execution of the workload.
- A second test system, which we call the SPA system, on 11.1.0.7; this can be a small system as no application schema nor data is required. This can even be on the same host as the test system with the schema and data where test-execution is performed. The SPA system will store the results of the experiment. It can be convenient to keep these results in a different system than the one we are upgrading as part of the experiment.



Both the test system and the SPA system should license Real Application Testing, while the Production system needs to license the Tuning Pack.

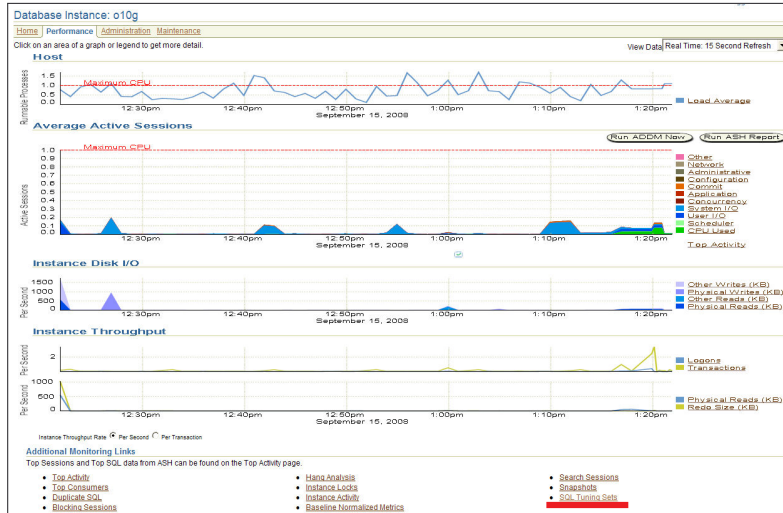
To prepare for the experiment, you will need to complete the following steps:

1. Clone 10.2.0.x production to 10.2.0.x test environment. If a full clone is not available you should create a system that is as similar as possible to production.
2. Apply test execution patch to 10.2.0.x test system (see metalink note #560977.1).
3. Create the 11.1.0.7 SPA system, which does not need any application schema or data and can even be on the same host as the test system.
4. Create a PUBLIC database link from the SPA system to the test system. This will allow the SPA system to connect to the test system in order to drive the experiment.

Database Upgrade: 10.2.0.x -> 10.2.0.y

1. Create STS in test database 10.2

- Use Performance page on 10.2 to create STS



ORACLE

Our first step is to create in the production database a SQL Tuning Set (STS) and capture SQL statements in this STS. To create STS in 10.2 Enterprise Manager use the link “SQL Tuning Sets” in the bottom right-hand corner of the performance page.

Database Upgrade: 10.2.0.x -> 10.2.0.y

1. Create STS in test database 10.2

- Create STS and load SQL statements from cursor cache.

The screenshot displays the Oracle SQL Tuning Sets management interface. At the top, it shows 'Database Instance: o10g > SQL Tuning Sets' and 'Logged in As SYS'. Below this, there's a search bar and a table of existing tuning sets. A red circle highlights the 'Create' button in the top right corner of the table area. A red arrow points from this button to the 'Create SQL Tuning Set: Options' dialog box that is open in the foreground. The dialog box shows the following fields:

- Database: o10g
- SQL Tuning Set Name: HR_WORKLOAD
- Owner: APPS
- Description: STS to test upgrade 11.2.x to 11.2.y

The dialog box is on 'Step 1 of 5' and includes 'Cancel' and 'Next' buttons.

ORACLE

On the STS page push the Create button.

Enter the STS name, STS owner name and STS description.

Database Upgrade: 10.2.0.x -> 10.2.0.y

1. Create STS in test database 10.2

- Use incremental capture to load all active SQLs from cursor cache during peak time

Options Load Methods Filter Options Schedule Review Logged in As SYS

Create SQL Tuning Set: Load Methods

Database: o10g [Finish] [Cancel] [Back] Step 2 of 5 [Next]

Pick one of the load methods to collect and load SQL statements into the SQL tuning set.

Incrementally capture active SQL statements over a period of time from the cursor cache
Specify the duration within which the SQL statements will be collected, and specify frequency over which the active SQL statements from the cursor cache will be collected repeatedly.

Duration: 24 Hours
Frequency: 5 Minutes

Load SQL statements one time only

Data Source: Cursor Cache

ORACLE

You can capture SQL statements from many sources – cursor cache, AWR, exported STS file etc. Since SPA needs to capture your entire workload, we will use the incremental capture facility to capture all SQL active over a period of time from the cursor cache. This example shows a capture job that will run for 24 hours and scan the cursor cache every 5 minutes. Note that the performance overhead for capturing a SQL Tuning Set is very low (<2%) and will not impact system throughput. You can capture all SQL or create filters to capture SQL from concrete modules, for example.

Database Upgrade: 10.2.0.x -> 10.2.0.y

2. Export/Import STS (10.2 -> 11.1) with OEM

- Use SQL Tuning Sets page of OEM to move STS from prod 10.2...

Database Instance: o10g > SQL Tuning Sets Logged in As SYS

SQL Tuning Sets

Page Refreshed Sep 11, 2008 12:28:34 PM [Refresh](#)

A SQL Tuning Set is a collection of SQL Statements that can be used for tuning purposes.

Search [Go](#)
Filter on a name or partial name

[View](#) [Export](#) [Schedule SQL Tuning Advisor](#) [Run SQL Access Advisor](#) [Delete](#) [Create](#)

Select	Name	Schema	Description	Created	Last Modified
☐	MR_TEST	SYS	Automatically generated by Top SQL	Sep 11, 2008 12:04:33 PM	Sep 11, 2008 12:04:33 PM

-To 11g SPA system

SQL Tuning Sets

Page Refreshed Sep 10, 2008 5:21:43 PM PDT [Refresh](#)

A SQL Tuning Set is a collection of SQL Statements that can be used for tuning purposes.

Search [Go](#)
Filter on a name or partial name

[Details](#) [Drop](#) [Export](#) [Schedule SQL Tuning Advisor](#) [Schedule SQL Access Advisor](#) [Create](#) [Import](#)

Select	Name	Schema	Description	SQL Count	Created	Last Modified
☐	TEST1	SYS			7/9/2/08 3:18 PM	9/2/08 3:29 PM
☐	SI_WKLD	SYSTEM			50 8/29/08 6:00 PM	8/29/08 6:00 PM
☐	HR_WORKLOAD	APPS	HR SQL Workload		50 8/26/08 7:34 PM	8/26/08 7:34 PM
☐	TEST	SYS	Automatically generated by Top SQL		3 8/6/08 2:40 PM	8/6/08 2:40 PM

Related Links
[SQL Performance Analyzer](#)

Once the capture completes, you can move the STS to the Oracle 11g SPA system using Oracle Enterprise Manager.

Database Upgrade: 10.2.0.x -> 10.2.0.y

Create Database link on test database 10.2.0.x

- Use Database Link page of OEM to create Database Link on test database
- User that will remote test execute from 11g db needs
 - Grant execute on dbms_sqlpa
 - Grant advisor privileges

Database Instance: v11gk

Home Performance Availability Server **Schema** Data Movement Software and Support

Database Objects **Programs**

Tables Packages
Indexes Package Bodies
Views
Synonyms
Sequences
Database Links
Directory Objects
Reorganize Object

Create Database Link

General Show SQL Cancel OK

Name test.us.oracle.com

Net Service Name TESTDB_1102x

Public - This database link is available to all users.

Connect As

Connected User
 Current User
 Fixed User

Username MASK

Password *****

Confirm Password *****

Show SQL Cancel OK

To run SQL statements from your STS on remote test database 10.2.0.x you have to create a public database link from the 11g SPA database to the test database. For this goal you can use the OEM GUI or the CREATE PUBLIC DATABASE LINK DDL.

The user that the link connects to on the test database needs the following privileges:

- EXECUTE on dbms_sqlpa
- ADVISOR privilege

Create SQL Performance task in 11g SPA

Database Instance: v11gk > Advisor Central > SQL Performance Analyzer > Guided Workflow > Logged in As SYS

Create SQL Performance Analyzer Task

The SQL Performance Analyzer Task is a container for the execution of trial experiments designed to test the effects of changes in execution environment on the SQL performance of an STS.

* Name

Owner: SYS

Description
TIP Use the description to characterize the intended SQL Performance Analyzer investigations.

SQL Tuning Set

The SQL Tuning Set is the basis for SQL Performance Analyzer Task experiments. The STS should represent a coherent set of SQL for the changes being investigated (e.g. full workload for an upgrade test).

* Name TIP You can create a new STS here: [Link to STS Creation Wizard](#)

ORACLE

Next step – create SQL performance task in 11g SPA system using the STS captured on the production database. EM provides step-by-step control over the SPA experiment using the Guided workflow UI. The Guided Workflow will take us through the following steps:

- 1. Create SPA Task – container to store results of STS executions (pre/post-change trials) and results of trial's comparison**
- 2. Replay STS in initial environment**
- 3. Replay STS in changed environment**
- 4. Compare the two trials**
- 5. View trial comparison reports**

Database Upgrade: 10.2.0.x -> 10.2.0.y

Remote STS execution

- Use Remote Test Execution on 10.2 over database link

The screenshot shows the Oracle Performance Analyzer interface. On the left, a 'Guided Workflow' pane lists steps: 1. Create SQL Performance Analyzer Task based on SQL Tuning Set, 2. Replay SQL Tuning Set in Initial Environment, 3. Replay SQL Tuning Set in Changed Environment, 4. Compare Step 2 and Step 3, 5. View Trial Comparison Report. The main area is titled 'Create SQL Trial' and contains the following fields and options:

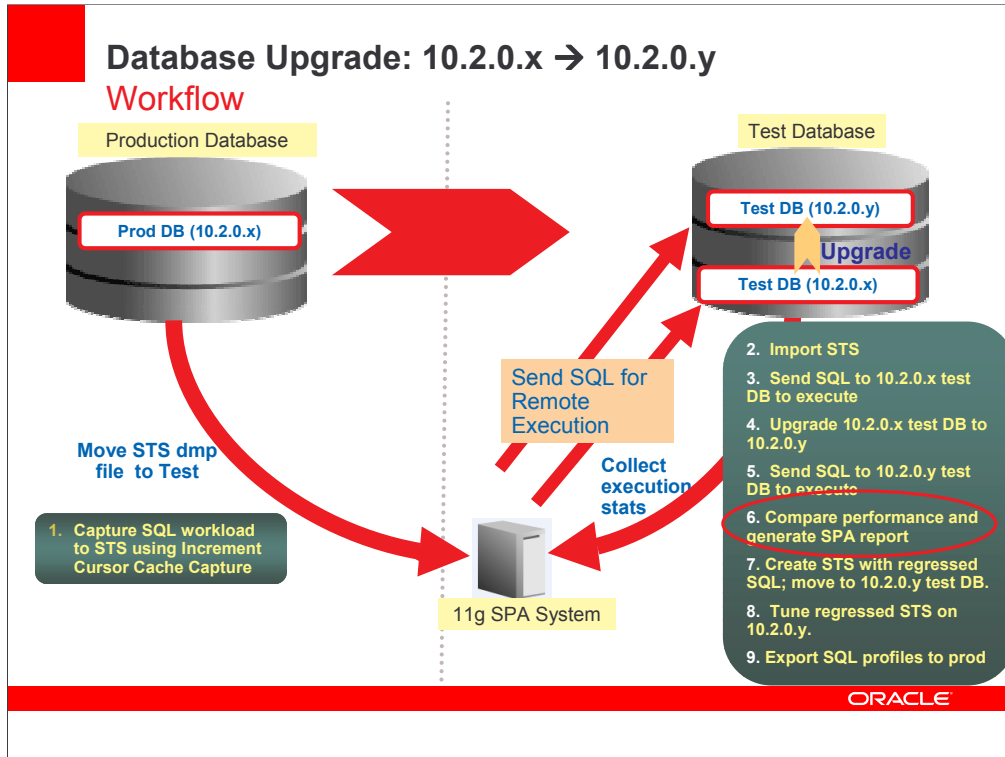
- SQL Performance Analyzer Task: **SYS.Upgrade102x_102y**
- SQL Tuning Set: **APPS.HR_WORKLOAD**
- SQL Trial Name: **AFTER_CHANGE**
- SQL Trial Description: **Remote Exec on 10.2.0.3**
- Creation Method: **Execute SQLs Remotely** (highlighted with a red circle)
- Per-SQL Time Limit: **Unlimited**
- Database Link: **test.us.oracle.com** (highlighted with a red circle)

Additional text in the form includes: 'SQL Trials capture execution performance of the SQL Tuning Set under a given optimizer environment.' and a tip: 'TIP Provide a PUBLIC database link connecting to a remote user with privileges to execute the Tuning Set SQL.'

ORACLE

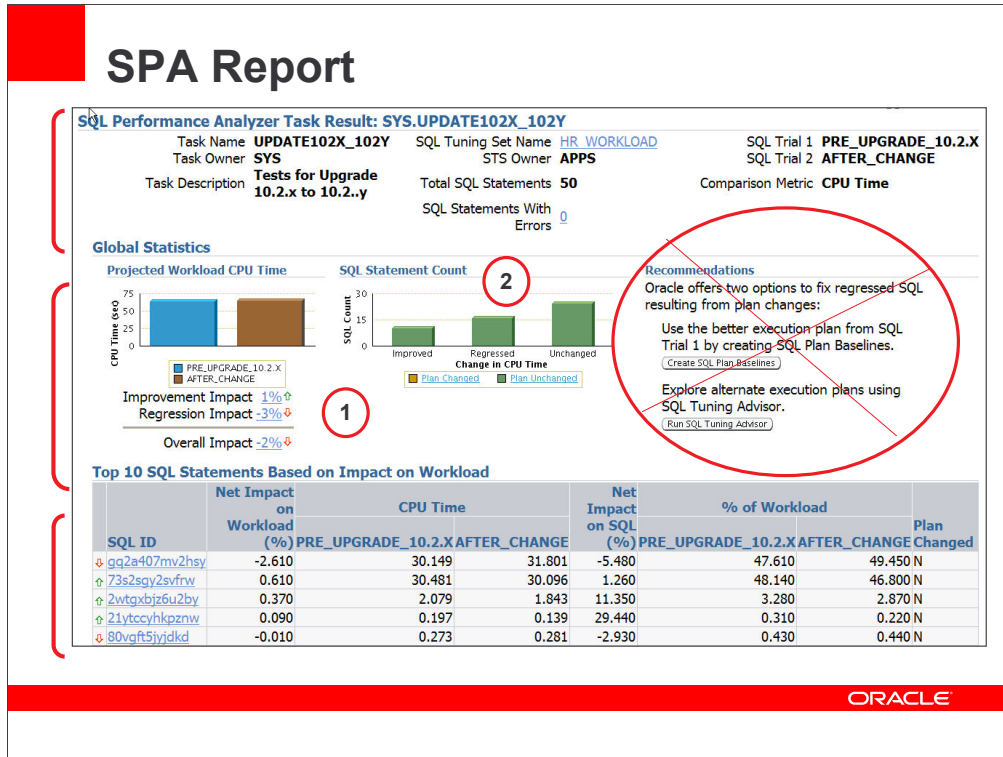
We will test our SQL in the old version of our database (10.2.0.x) and then in the upgraded version of our database (10.2.0.y). To do this we will run SPA from the Oracle 11g SPA system and build pre/post-upgrade trials. The trials will execute using the DB Link, so we call them remote trials. In the Create SQL Trial form we will choose Creation Method = Execute SQLs Remotely and select the DB link name to the test database.

After the trial completes, we can upgrade the test database from release 10.2.0.x to release 10.2.0.y. Then perform the second trial, just as we have here, to execute the SQLs in version 10.2.0.y. All of the performance data for both trials will be stored in the 11g SPA system.



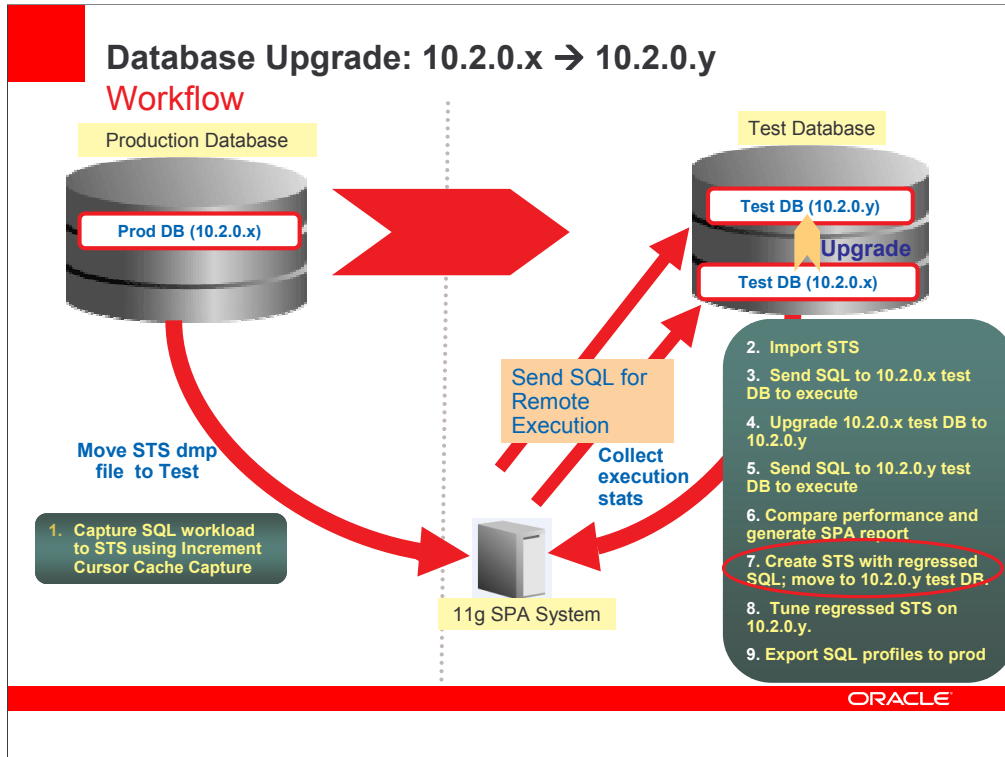
Next step – compare pre- and post-upgrade trials and build a comparison report.

SPA Report



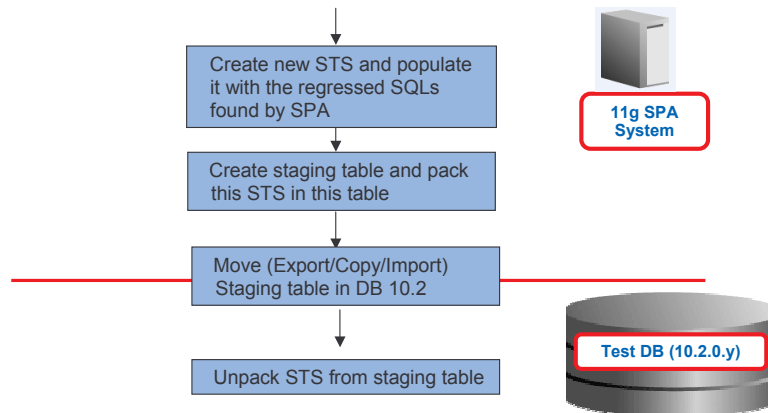
In the SPA report we can see the upgrade’s impact on SQL performance (1) and the number and list of regressed SQL (2). From the summary screen you can drill down to all of the details for each SQL statement, including execution plans and statistics from before and after the change. Look over the report to decide which SQL statements seem to be experiencing significant regressions that you would like to tune.

Our next step will be to tune the regressed SQLs on the 10.2 system. We cannot tune them directly from the SPA UI as as tuning needs to happen on the 10.2.0.y test DB, where the SPA trials are executed and the application schema and data will be available to the SQL Tuning Advisor. Also note that SQL Plan Baselines are a new 11g feature and cannot therefore be used for a 10.2 upgrade.



After reviewing the SPA reports, our next step will be to create a SQL Tuning Set containing only the regressed SQLs we wish to tune using the SQL Tuning Advisor and move it to the test system.

Moving regressed SQL to 10.2 for tuning



ORACLE

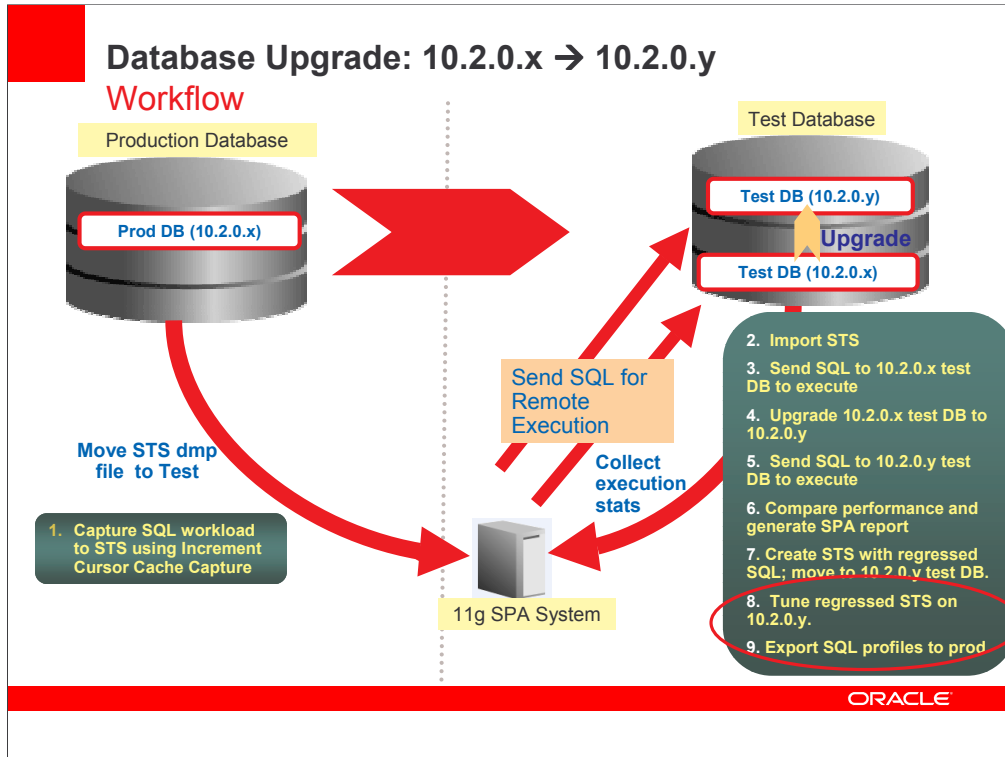
Command-line API usage simplified with scripts available in Metalink Note #560977.1

The next step will be to materialize just the subset of SQLs we wish to tune into a new SQL Tuning Set, move that STS to the test system, and tune the SQLs there.

As we have already shown, Oracle Enterprise Manager supports moving an STS between Oracle Databases using the STS export/import facilities. In this case, however, since we are moving the STS to an older release, we need to complete the steps manually as the APIs do not support moving an STS to an older release. To simplify this for you we have provided some SQL*Plus scripts (see Metalink note #560977.1).

We will do the following:

1. Subset our production STS to contain only regressed SQLs identified by SPA (run script 6_regressions2sts.sql)
2. Create a staging table and pack the STS into the table (run script 7_sts2stgtab.sql)
3. Move the staging table to the 10.2.0.y test system (using the standard export/import or datapump facilities)
4. Import the dump file and unpack the staging table using the DBMS_SQLTUNE.UNPACK_STGTAB_SQLSET API.



In the next step we will use the SQL Tuning Advisor on the 10.2 test system to tune the regressed SQLs. You can create SQL profiles as recommended by the advisor and perform any other tuning actions you deem necessary to fix the SQL regressions detected by SPA. After tuning the system we recommend performing another SPA trial and comparing it to your first trial to make sure the regressions have disappeared prior to upgrading the production system.

Also you should keep track of whatever tuning actions you have taken, as these will need to be re-issued after upgrading the production DB. Any SQL profiles created can be exported from the test DB to the production DB using the SQL Profile export/import facility (described in the following slides).

Database Upgrade: 10.2.0.x -> 10.2.0.y

Tuning regressed SQL in test database 10.2.0.y

- Use STS page in Oracle 10.2.0.y to run SQL Tuning Advisor or SQL Access Advisor and tune STS with regressed SQL

Database Instance: o10g > SQL Tuning Sets Logged in As SYS

SQL Tuning Sets

Page Refreshed Sep 15, 2008 2:50:14 PM

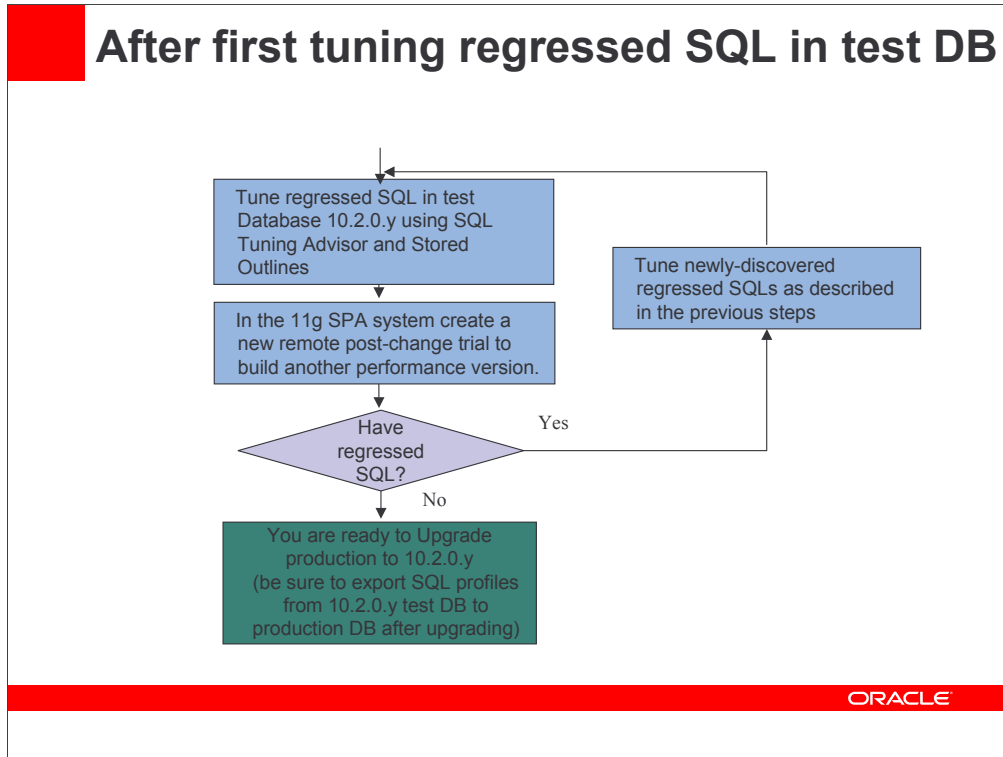
A SQL Tuning Set is a collection of SQL Statements that can be used for tuning purposes.

Search
Filter on a name or partial name

Select	Name	Schema	Description	Created	Last Modified
o	REGRESSED_SQL	SYS		Sep 15, 2008 2:50:00 PM	Sep 15, 2008 2:50:08 PM
c	MR_TEST	SYS	Automatically generated by Top SQL	Sep 11, 2008 12:04:33 PM	Sep 11, 2008 12:04:33 PM

ORACLE

To run the SQL Tuning Advisor on the test system use the “Schedule SQL Tuning Advisor” button on STS page in your test database. You can then review the SQL Tuning results and implement changes as you deem necessary.



To tune regressed SQL in test database we can use many features – SQL profiles, indexes, MV, refresh statistics, stored outlines, change database parameters etc. These actions can improve SQL performance but at the same time they may impact some SQL negatively. For this reason we recommend running further SPA trials after performing any tuning action and comparing the performance to the first trial in order to make sure the changes had their desired effect.

After tuning regressed SQLs in 10.2.0.y return to the 11g SPA system and build a new post-upgrade trial in SPA using remote STS execution as in previous cases. Compare this new trial with pre-upgrade trial and analyze the report. Enterprise Manager contains support for performing all of these steps. If you have new regressed SQLs in this report, you should repeat the process iteratively until all performance issues are resolved.

When all issues are resolved, you are ready to upgrade the production system. Don't forget to implement the same tuning actions after upgrading production that you made on the test system. In the case of SQL profiles created on test, we describe an easy way to move them to the production system (similar to how we have moved STSes already) in the following slides.

Database Upgrade: 10.2.0.x -> 10.2.0.y

Export SQL profiles for future production system

- If you create SQL profiles in the 10.2 test system, export them and later import in production system after upgrade
- Use Expdp/Impdp for export/import stage table
- Unpack SQL profiles, imported in new production 10.2.0.y database, using CLI, to re-create them.

```
begin
  dbms_sqltune.create_stgtab_sqlprof(
    table_name => 'SQLPROF_TAB');

  -- pack all sql profiles, for export
  dbms_sqltune.pack_stgtab_sqlprof(
    profile_name => '%',
    staging_table_name => 'SQLPROF_TAB');
end;
/
```

```
begin
  -- pack all sql profiles, for export
  dbms_sqltune.unpack_stgtab_sqlprof(
    replace => TRUE,
    staging_table_name => 'SQLPROF_TAB');
end;
/
```

ORACLE

To export/import SQL profiles in Oracle 10.2 you can use the command line `dbms_sqltune` API as described here.

More information available online

- 11.1.0.7 Real Application Testing Guide (Core DB Doc) gives an overview of SPA and shows EM / API examples of how to use it.
- Oracle White Paper on OTN: “Testing the SQL Performance Impact of an Oracle 9i/10.1 to Oracle 10g Release 2 upgrade with SQL Performance Analyzer” provides in-depth info on the 9i/10.1->10.2 use case.
 - 10.2.0.x->10.2.0.y is a simpler use case (see Notes)
- Metalink note #560977.1 contains all the information about how to use SPA in this and other upgrade scenarios.
- SQL*Plus Scripts available on OTN (under Database, Manageability section, Real Application Testing)

ORACLE

We have written a technical white paper about the 9i/10.1 use case, available at http://www.oracle.com/technology/products/manageability/database/pdf/owp_spa_9i.pdf. When you are reviewing the white paper, please note that the 9i/10.1 case has several key differences from the 10.2.0.x->10.2.0.y case, especially the following:

- **For the 10.2.0.x->10.2.0.y flow we capture the production workload into a SQL Tuning Set rather than enabling SQL trace. This has several implications on the workflow:**
 - No need to capture SQL trace / mapping table; you can just directly capture an STS
 - Execution Frequency will be captured into the STS, so you can depend on SPA to weigh improvements and regressions accordingly without needing your own knowledge of the application. In particular, this means you should not set the `WORKLOAD_IMPACT_THRESHOLD` / `SQL_IMPACT_THRESHOLD` task parameters as described in the white paper, since those values are only appropriate when the execution frequency is not captured.
 - Because capture STS has a negligible performance overhead, you do not need to worry about impacting the performance of the production system when capturing the workload as with sql trace

More information available online

- Notes continued

ORACLE

- **For the 10.2.0.x->10.2.0.y flow we do one remote test-execute on version 10.2.0.x and a second on 10.2.0.y, while the 9i->10.2 scenario built an STS from production stats and did only a single test execute, in 10.2.**
 - This makes for a more scientific test as both sides of the comparison are stats generated by SPA on the test system
 - In the 9i->10.2 case we had to account for the overhead of SQL trace when comparing SQL trace stats to SPA test-execute stats. This is no longer necessary.
 - Because SQL trace so greatly impacts the response time of SQL statements, we could not use the ELAPSED_TIME metric and instead had to count on BUFFER_GETS and CPU_TIME; In the 10.2.0.x->10.2.0.y case ELAPSED_TIME is a sensible performance metric since the sql trace overhead is no longer in effect.
 - The 10.2.0.x -> 10.2.0.y case requires upgrading your test system between running the before and after trials; the alternative is to have two test systems, one on 10.2.0.x and the second on 10.2.0.y. If the hardware resources are available this can be convenient as it keeps both releases available for diagnosing issues after getting the SPA report. But it does require extra diligence to make sure that the 10.2.0.x and 10.2.0.y systems are in fact identical in terms of data, optimizer stats, and configuration.