

ORACLE®



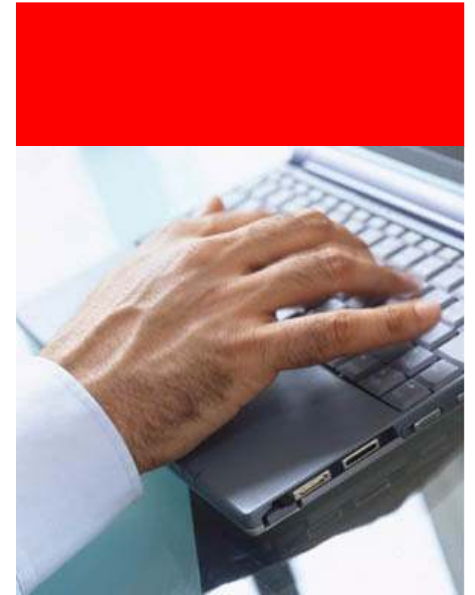
ORACLE®

What to expect from the Optimizer when upgrading from Oracle Database 10g to 11g

Maria Colgan & Mohamed Zait

Agenda

- Changes in behaviour
- SQL Plan Mangement
- Pre-upgrade checklist
- Post-upgrade checklist
- Correcting regressed SQL Statements



ORACLE
OPEN
WORLD

ORACLE
OPEN
WORLD

**Changes in
behavior**

ORACLE®

Init.ora Parameters

Parameter	Function	In 10g	In 11g
Optimizer_mode	Cost-based Optimizer used for all SQL Statements	All_rows	All_rows
Optimizer_Dynamic_Sampling	If no statistics on an object automatically gathered at parse	2	2
Optimizer_Secure_view_merging	Additional security checks before merging a view	True	True
Optimizer_use_invisible_indexes	Allows Optimizer to use an invisible index as access method	N/A	False
Optimizer_use_pending_statistics	Allows Optimizer to use an pending statistics	N/A	False
Optimizer_capture_SQL_plan_baselines	Automatically captures execution plans into SPM	N/A	False
Optimizer_use_SQL_plan_baselines	Optimizer uses any existing SQL Plan Baseline	N/A	True

New DBMS_STATS Subprograms

Subprogram	Function	In 10gR2	In 11g
Gather_System_Stats	Gathers stats on CPU and IO speed of H/W	Yes	Yes
Gather_Dictionary_Stats	Gathers stats on dictionary objects	Yes	Yes
Gather_Fixed_Object_Stats	Gather stats on V\$views	Yes	Yes
Publish_Pending_stats	Pending stats allows stats to be gather but not published immediate	N/A	Yes
Restore_Table_Stats	Revert stats back to what they were before	10.2.0.4	Yes
Diff_Table_Stats	Compare stats for a table from two different sources	10.2.0.4	Yes
Create_Extended_stats	Gathers stats for a user specified column group or an expression	N/A	Yes
Set_Table_Perfs	Sets stats preferences of a table	N/A	Yes



Automatic Statistics Gathering Job

- Introduced in 10g
- Gathers statistics on objects where
 - Statistics are missing
 - Statistics are stale
- In 10g its an Oracle Scheduler job
 - Runs during maintenance window
- In 11g its an Autotask
 - Runs during maintenance window
 - Use `DBMS_AUTO_TASK_ADMIN` package to control job



New Features

- New Optimizations
 - Group-by placement
 - Enhanced join predicate push down
 - Null-aware antijoin
- Adaptive Cursor Sharing (enhanced bind peeking)
- Extended Statistics
 - Multi-column statistics (for correlation)
 - Statistics on expressions
- Pending Statistics

Each new features could potentially change a plan

How can you maintain performance -> stability during upgrade?

SQL Plan Management



SQL Plan Management



SQL Plan Management

Prior to 11g

- Unpredictable changes can happen to an execution plan
- Avoiding plan changes the only method to avoid performance regression
 - Lock Statistics to prevent them from changing
 - Freezing an execution plan with a Stored Outline
- No mechanism for plans to evolve

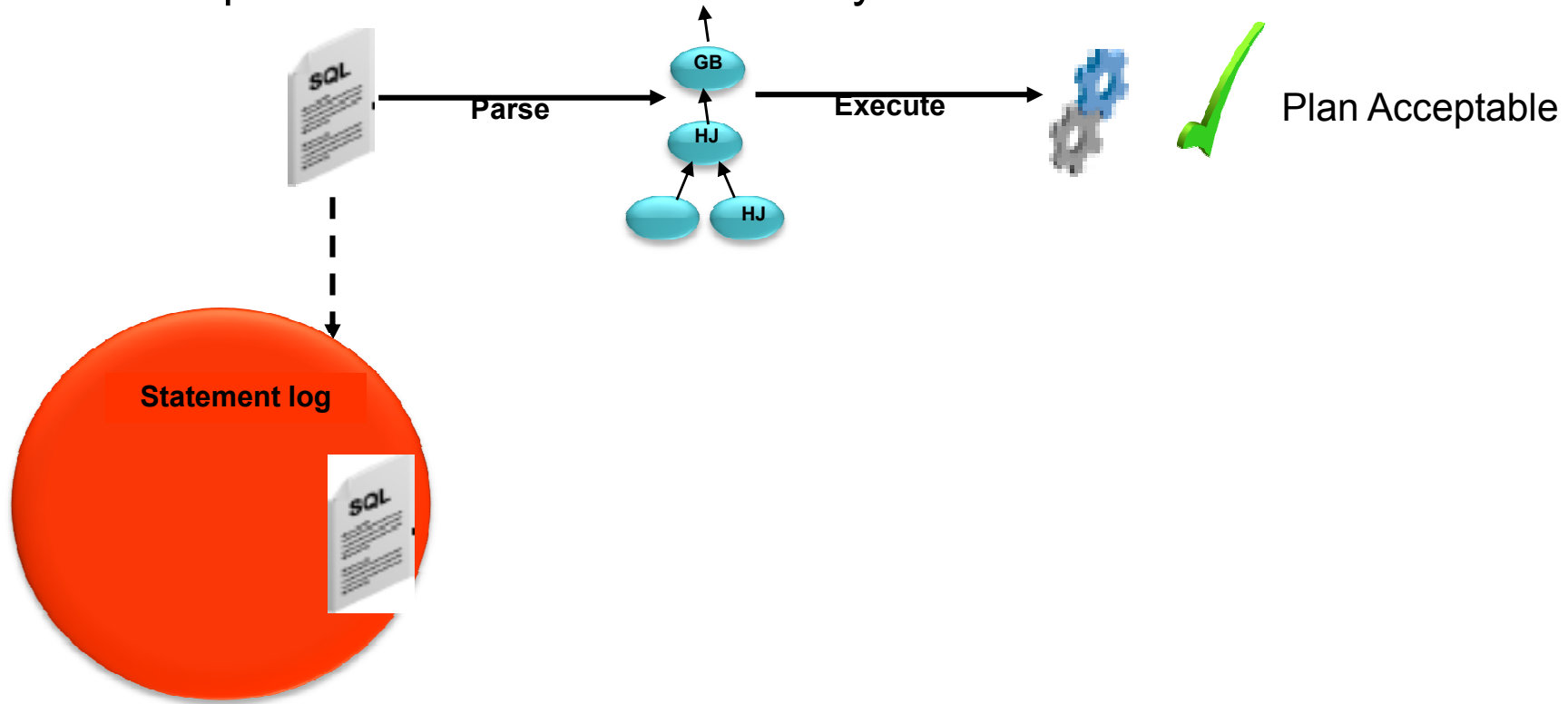
Solution

- Optimizer automatically manages ‘execution plans’
 - Only known and **verified** plans are used
- Plan changes are verified
 - Only comparable or better plans are used going forward

SQL Plan Management is controlled plan performance

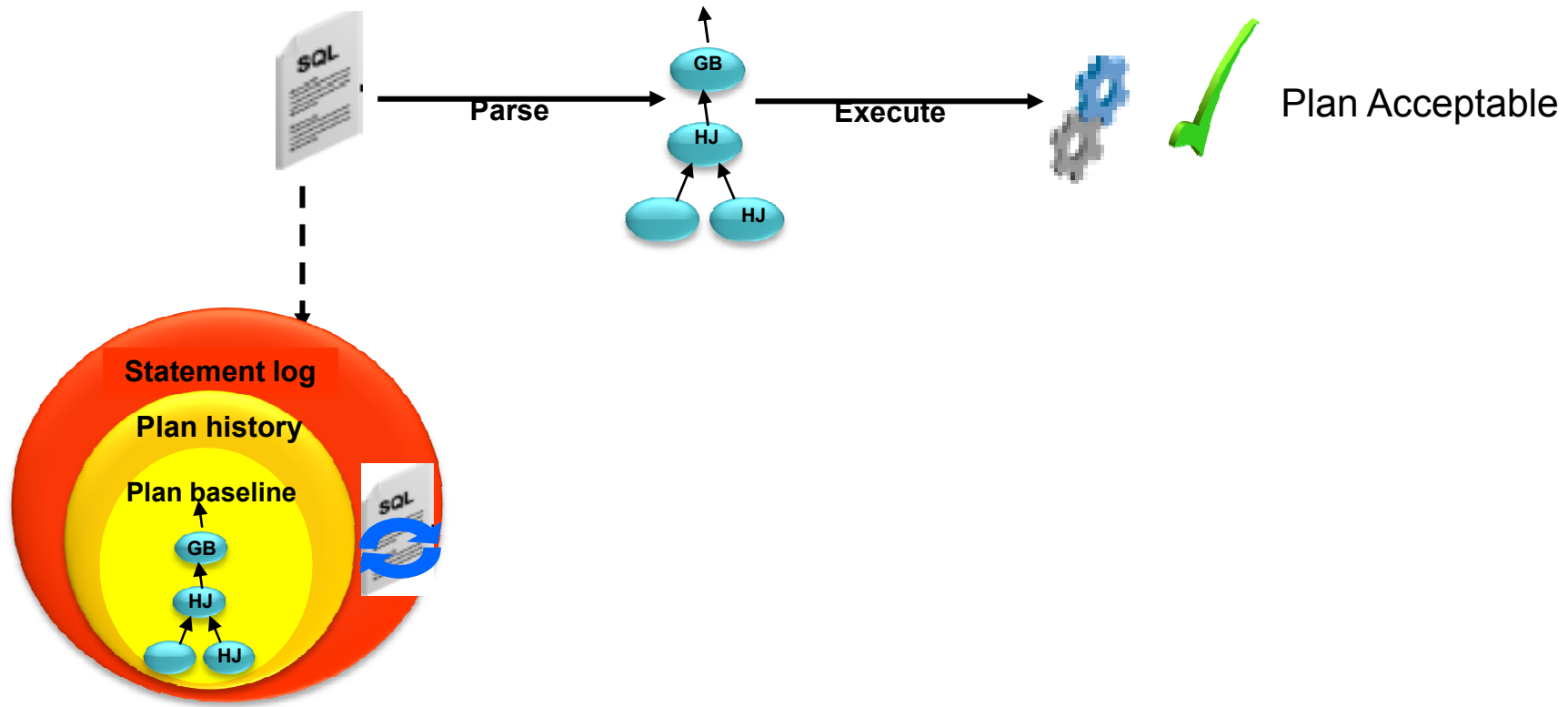
With SQL Plan Management

- SQL statement is parsed for the first time and a plan is generated
- Check the log to see if this is a repeatable SQL statement
- Add SQL statement signature to the log and execute it
- Plan performance is still “verified by execution”



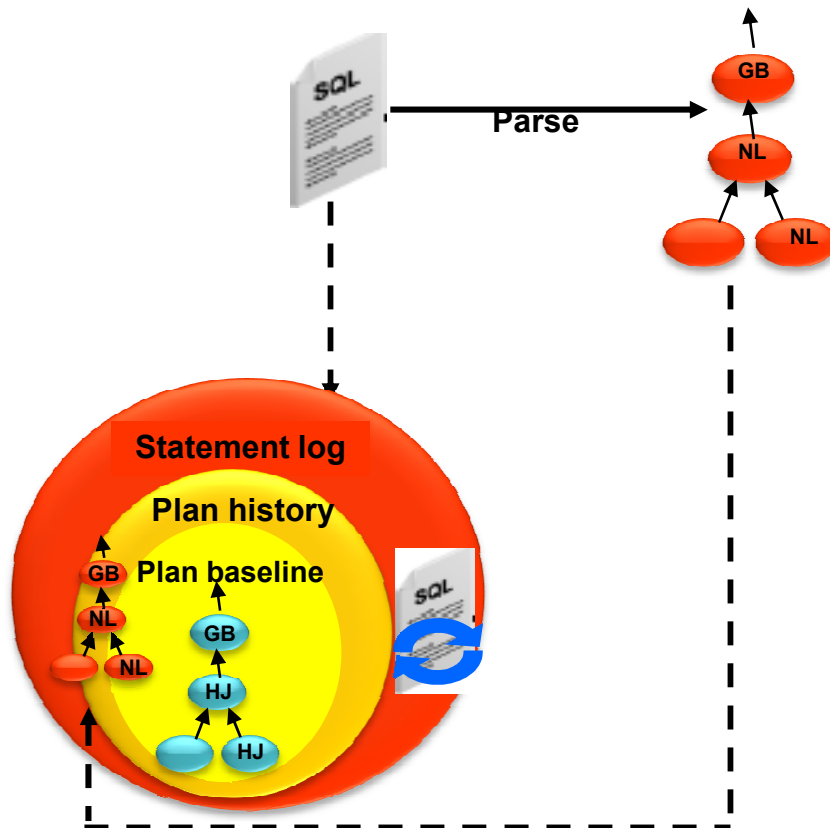
With SQL Plan Management

- SQL statement is parsed again and a plan is generated
- Check log to see if this is a repeatable SQL statement
- Create a Plan history and use current plan as SQL plan baseline
- Plan performance is “verified by execution”



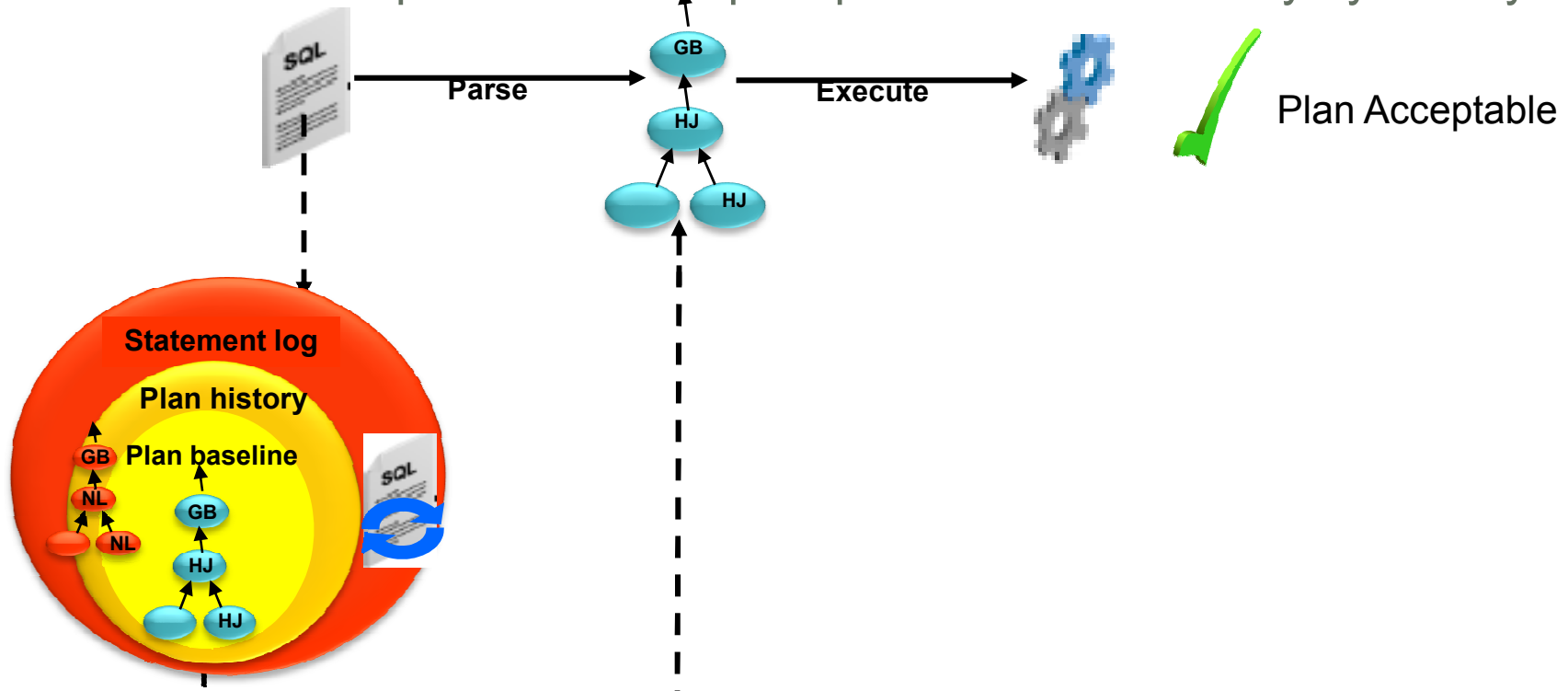
With SQL Plan Management

- Something changes in the environment
- SQL statement is parsed again and a **new plan is generated**
- New plan is not the same as the baseline – **new plan is not executed** but marked for verification



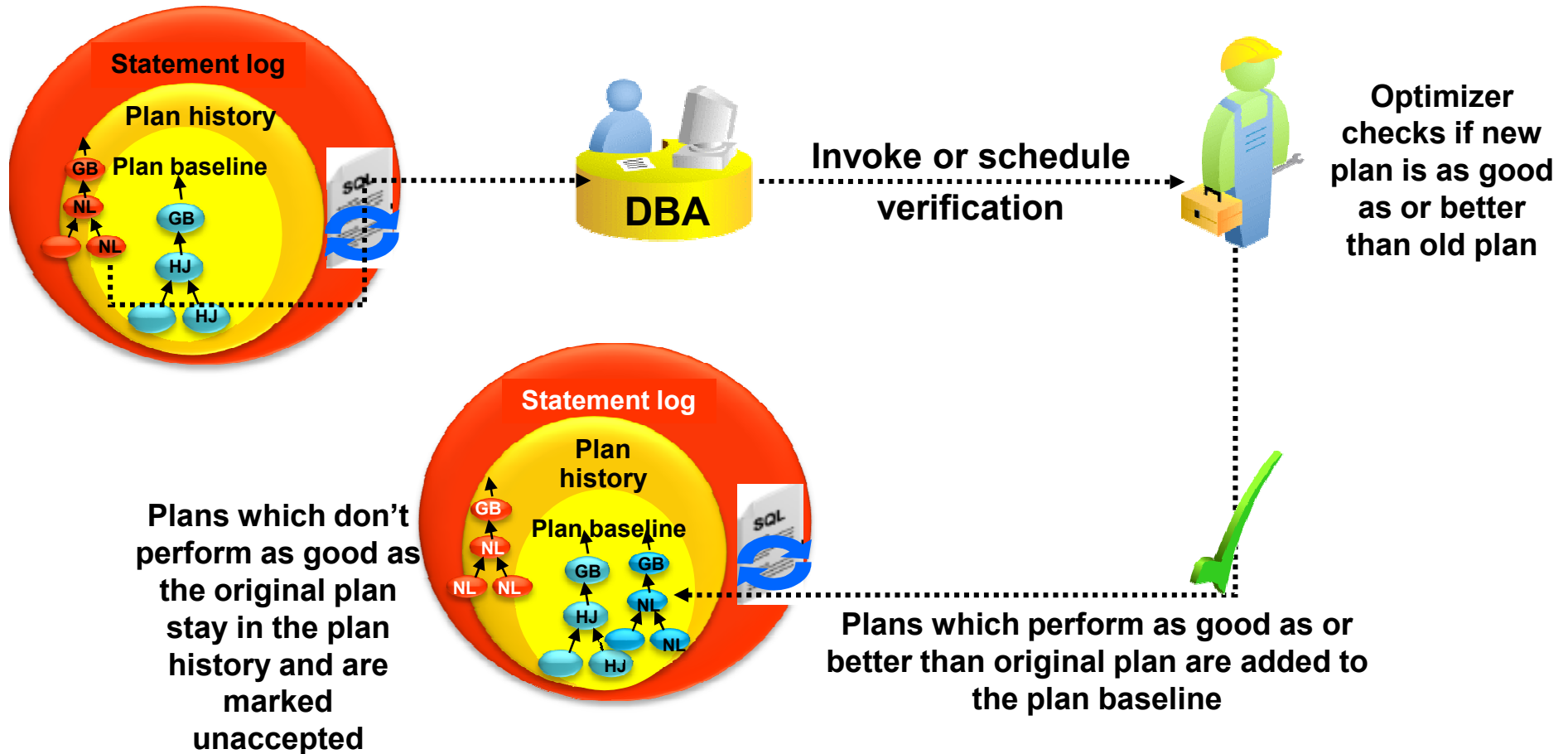
With SQL Plan Management

- Something changes in the environment
- SQL statement is parsed again and a **new plan is generated**
- New plan is not the same as the baseline – **new plan is not executed** but marked for verification
- Execute known plan baseline - plan performance is “verify by history”



Verifying the new plan

- Non-baseline plans will not be used until verified
- DBA can verify plan at any time





SQL Plan Management – the details

- Controlled by two init.ora parameter
 - ***optimizer_capture_sql_plan_baselines***
 - Controls auto-capture of SQL plan baselines for repeatable stmts
 - Set to `FALSE` by default in 11gR1
 - ***optimizer_use_sql_plan_baselines***
 - Controls the use of existing SQL plan baselines by the optimizer
 - Set to `TRUE` by default in 11gR1
- Monitoring SPM
 - Dictionary view `DBA_SQL_PLAN_BASELINE`
 - Via the SQL Plan Control in EM DBControl
- Managing SPM
 - PL/SQL package `DBMS_SPM` or via SQL Plan Control in EM DBControl
 - Requires the ‘administer sql management object’ privilege



SPM Plan Capture – Bulk

- From SQL Tuning Set (STS)
 - Captures plan details for a (critical) set of SQL Statement in STS
 - Load these plans into SPM as baseline plans
- From Stored Outlines
 - Migrate previously created Stored Outlines to SQL plan baselines
- From Cursor Cache
 - Load plans from the cursor cache into SPM as baseline plans
 - Filters can be specified (SQL_ID, Module name, schema)
- From staging table
 - SQL plan baselines can be captured on another system
 - Exported via a table (similar to statistics) and imported locally
 - Plan are “unpacked” from the table and loaded into SPM



Pre-Upgrade Steps



Pre-Upgrade Step

- Testing on the new Database Release
 - Use hardware identical to product
 - Use a copy of the 'live' data from product
 - Ensure all important queries and reports are tested
 - Capture all necessary performance information during tests
 - Ensure comparable test results are available for your current Oracle release
- Capture current 10g execution plans
 - Using SQL Performance Analyzer
 - Using Stored Outlines
 - Using SQL Tuning Sets
 - Using exported SQL plan baselines

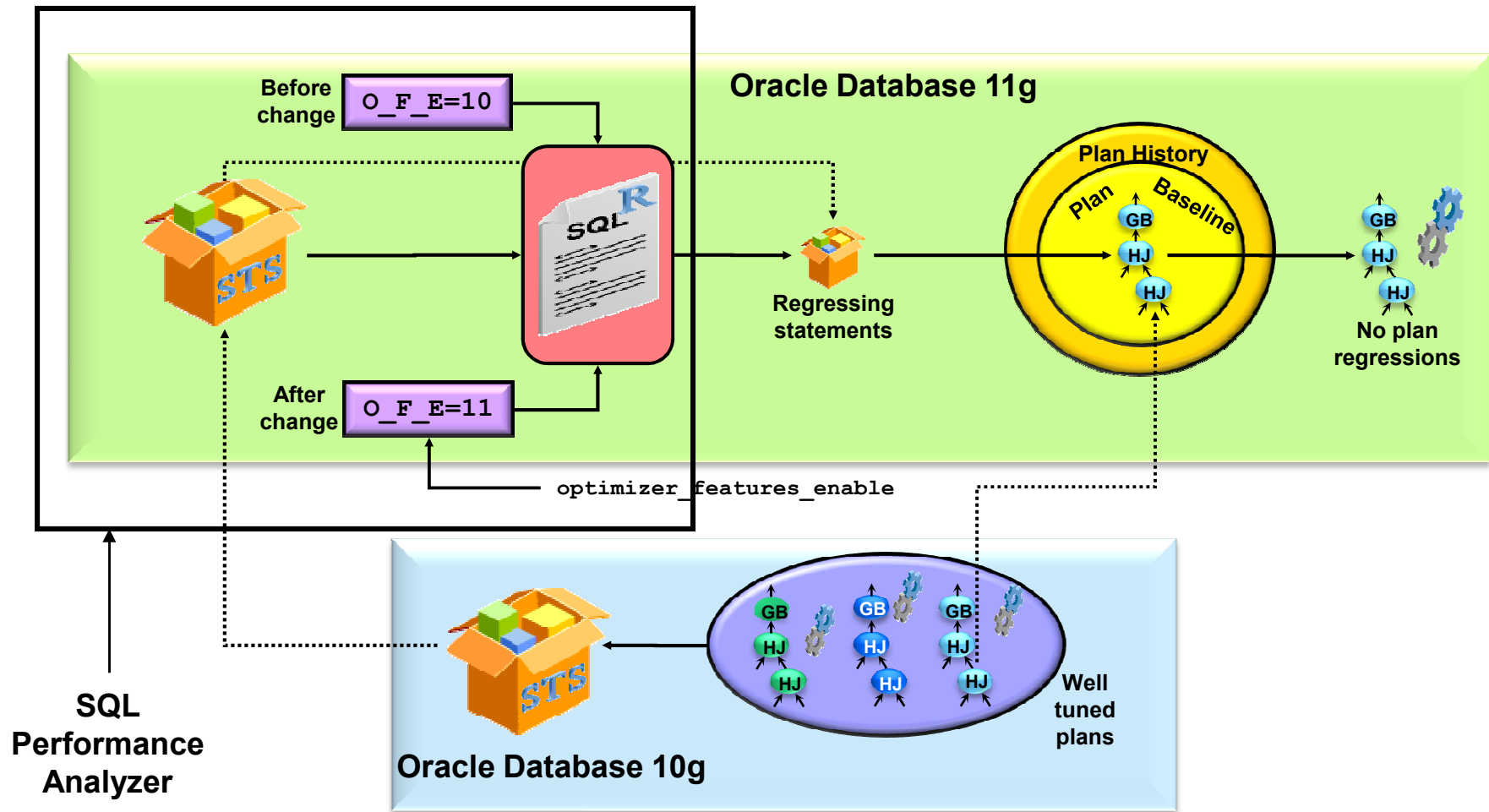


Testing on the new database release

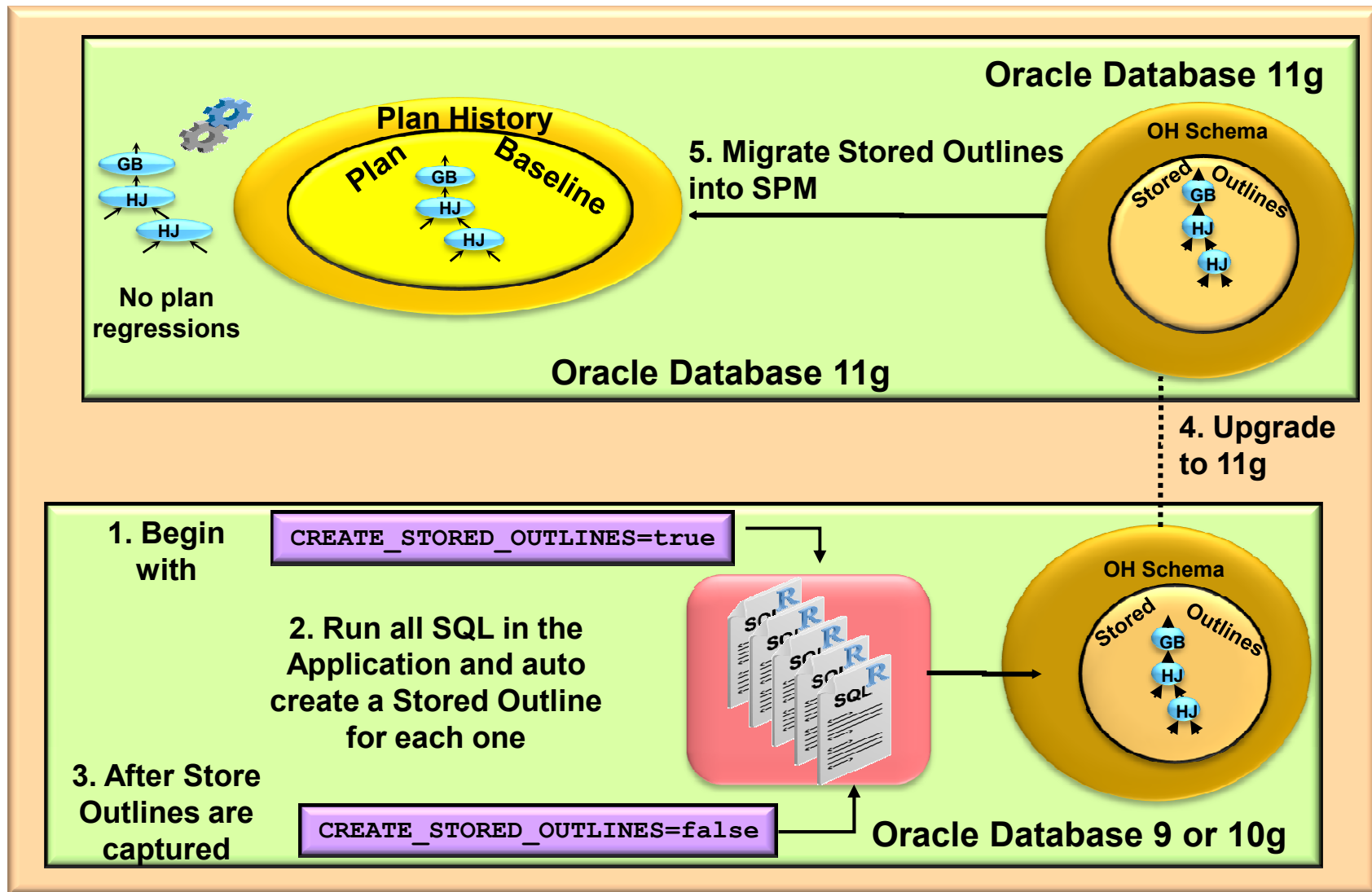
Removing old Optimizer hints

- If there are hints for every aspect of the execution plan the plan won't change between releases (Stored Outline)
- Partial hints that worked in one release may not work in another
- Test all SQL stmts with hints on the new release using the parameter `_optimizer_ignore_hints=TRUE`
 - Chance are the SQL stmts will perform better without any hints

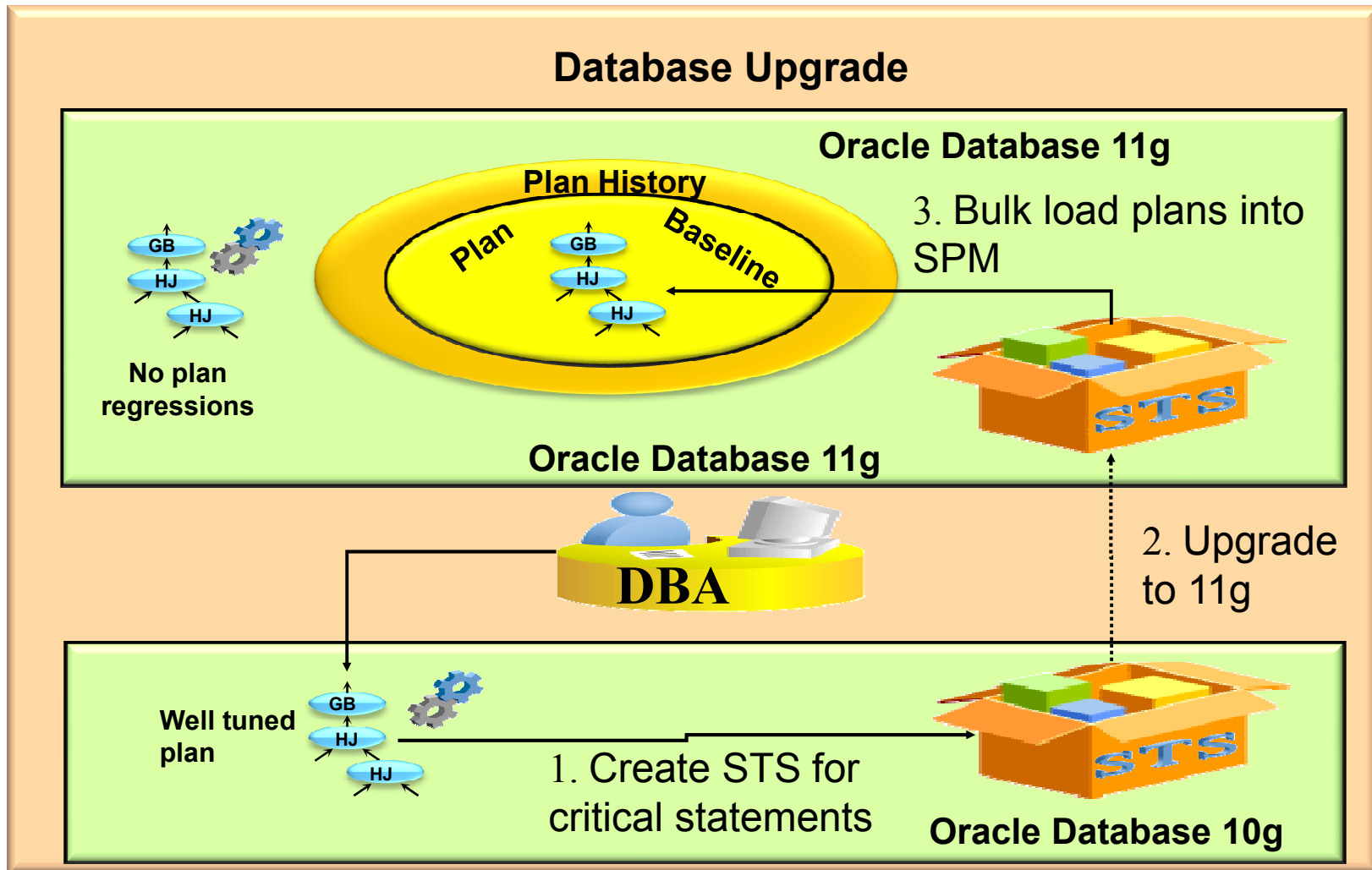
Capturing Plans using SPA



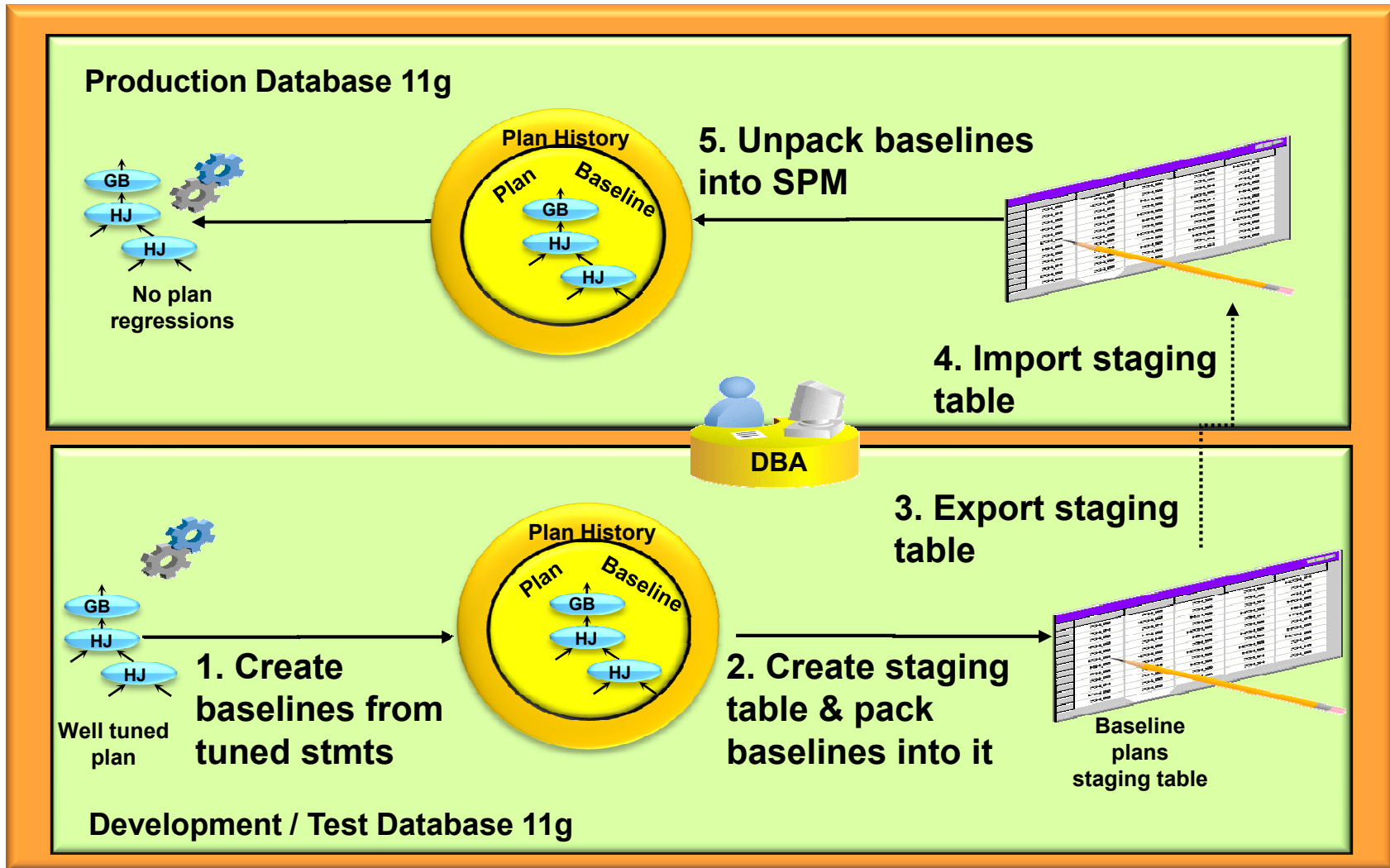
Capturing Plans using Stored outlines



Capturing Plans using SQL Tuning Set



Capturing Plans Using an 11g test environment





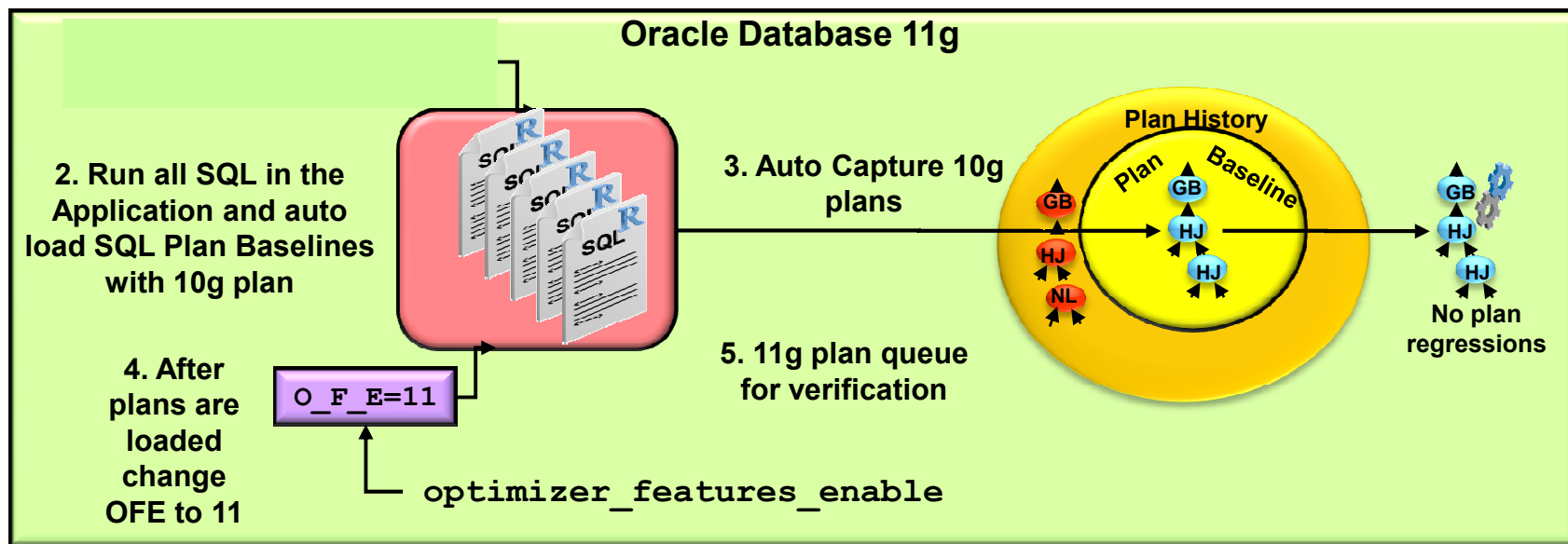
Post-Upgrade Steps



Post-upgrade Steps

- Load SPM with 10g plans
 - From a STS create in Oracle Database 10gR2
 - From Stored Outlines
 - From SQL Tuning Set
 - From a staging table
 - From the Cursor Cache
- Manage Optimizer Statistics

SQL Plan Management - general upgrade strategy

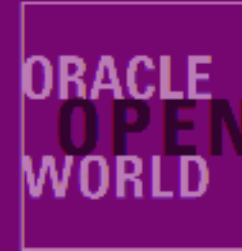


- Seeding the SQL Plan Baselines with 10g plans No plan change on upgrade
- After all SQL Plan Baselines are populated switch Optimizer_Features_Enable to 11
 - new 11g plans will only be used after they have been verified



What to do with statistics after upgrade

- Use last known 10g stats until system is stable
- Switch on incremental statistics for partitioned tables
 - `DBMS_STATS.SET_GLOBAL_PREFS('INCREMENTAL','TRUE');`
- Temporarily switch on pending statistics
 - `DBMS_STATS.SET_GLOBAL_PREFS('PENDING','TRUE');`
- Gather 11g statistics
 - `DBMS_STATS.GATHER_TABLE_STATS('sh','SALES');`
- Test your critical SQL statement with the pending stats
 - `Alter session set optimizer_use_pending_statistics=TRUE;`
- When proven publish the 11g statistics
 - `DBMS_STATS.PUBLISH_PENDING_STATS();`



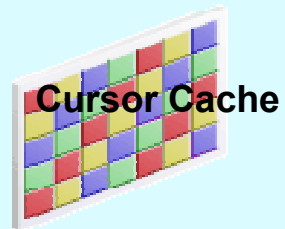
Correcting Regressed SQL Statements

Correcting Regressed SQL Statement

Load plans
from a SQL
Tuning Set



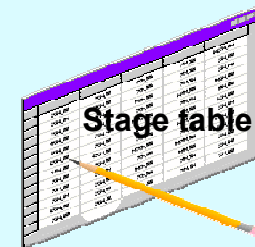
Load plans
from the
Cursor
Cache



Load plans
from Stored
Outlines



Load plans
from a
staging
table





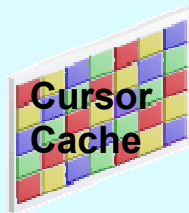
Upgrade Demo

Correcting Regressed SQL Statement

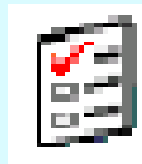
Load plans
from a SQL
Tuning Set



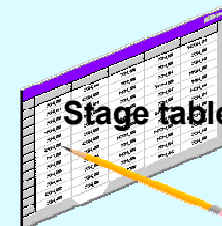
Load plans
from the
Cursor
Cache



Load plans
from Stored
Outlines



Load plans
from a
staging
table



Load a
hinted
execution
plan

Hints



For More Information


search.oracle.com

Upgrading Optimizer



or

http://www.oracle.com/technology/products/bi/db/11g/pdf/twp_upgrading_10g_to_11g_what_to_expect_from_optimizer.pdf



The preceding 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.