Oracle Database 11*g*: SQL Plan Management

Harald van Breederode
Oracle University
30-NOV-2009

About Me

- Senior Principal DBA Trainer – Oracle University
- 25 years Unix Experience
- 12 years Oracle DBA Experience
- Oracle8i, 9*i*, *10g* and 11*g* OCP
- Oracle10*g* OCM
- DBA Certification Exam Team Reviewer
- DBA Curriculum Development Reviewer
- Blog: prutser.wordpress.com
- Visually Impaired (Legally Blind)

ORACLE
10*g* Certified Master

Agenda

- Introduction
- Why do execution plans change?
- How to prevent plan changes
- Using optimizer hints
- Using Optimizer Plan Stability
- Using SQL Plan Management
- Demos
- Questions & Answers

Introduction

- SQL statement processing requires an execution plan
- An optimizer generates and selects this plan
- Two optimizer approaches are available:
  - Rule Based Optimization (RBO)
  - Cost Based Optimization (CBO)
- Performance depends on the chosen plan
- One plan can perform better than another
- Occasionally a sub-optimal plan is chosen
  - The question is: How can we prevent the optimizer from choosing the ‘wrong execution plan’?

About Execution Plans

The optimizer selects an execution plan based on:

- SQL statement
- Schema objects

And if CBO is used:

- System parameters
- Session parameters
- Object statistics
- System statistics
- Various other things

Why do execution plans change?

A plan can change when:

- Schema objects change
- Oracle software change

And if CBO is used:

- System parameters change
- Session parameters change
- Object statistics change
- System statistics change

Question

Is a change always welcome?

Answer

As always: It depends!

How to prevent plan changes?

To avoid plans from changing:

- Don’t change your data
- Don’t change statistics
- Don’t change parameters
- Don’t change the Oracle software

Or

- Control the optimizer with
  - Hints
  - Plan Stability
  - SQL Plan Management

Using Optimizer Hints

- Influences the optimizer's decision
- Can not be ignored
- Change your code
- Documented in `V$SQL_HINT`
- Difficult to maintain
- Not *all* hints are evil
- When not used carefully, strange things may happen

Pros and Cons of Hints

Pros:

- Controlled by the developer

Cons:

- Requires access to application code
- Maintenance is a manual task
- Not controlled by the DBA
- Better execution plans remain unnoticed

Advise:

- Use with caution!

Using Optimizer Plan Stability (1/2)

- Plan Stability uses a Stored Outline, a set of hints
- Stored Outlines are organized in categories
- Stored Outlines are created with:
  - `CREATE_STORED_OUTLINES` parameter
  - `CREATE OUTLINE` statement
- Stored in the `OUTLN` schema
  - `DBA_OUTLINES (OL$)`
  - `DBA_OUTLINE_HINTS (OL$HINTS)`

Using Optimizer Plan Stability (2/2)

- **Usage controlled by `USE_STORED_OUTLINES` parameter**
- **Stored Outlines do not affect the use of the following parameters:**
  - **`QUERY_REWRITE_ENABLED`**
  - **`STAR_TRANSFORMATION_ENABLED`**
  - **`OPTIMIZER_FEATURES_ENABLE`**

Maintaining Stored Outlines

- Stored Outlines can be moved to another tablespace using Export and Import
- Stored Outlines can be managed using:
  - `DBMS_OUTLN` package
  - `DBMS_OUTLN_EDIT` package
  - `ALTER OUTLINE` statement

Pros and Cons of Stored Outlines

Pros:

- **Controlled by the DBA**
- **Easy to implement**
- **Easy to maintain**

Cons:

- **Do not control everything**
- **Better plans remain unnoticed**

Advise:

- **Use only on releases prior to Oracle11*g***

About SQL Profiles

- A SQL profile contains extra optimizer information
- Requires the (licensed) SQL Tuning Advisor
- Does NOT “freeze” a particular plan
- Does not prevent SQL performance regression

- Note: An active SQL profile will be of influence during stored outline creation

Using SQL Plan Management (1/2)

- A SQL Management Base stores execution plans
- The SMB resides in the `SYSAUX` tablespace
  - `DBA_SQL_PLAN_BASELINES`
- 3 Methods for loading SQL plan baselines in the SMB:
  - Auto: `OPTIMIZER_CAPTURE_SQL_PLAN_BASELINES`
  - Manual: `DBMS_SPM` package
  - Running the SQL Tuning Advisor
- A SQL plan baseline in the SMB has 3 important attributes
  - `ENABLED`
  - `ACCEPTED`
  - `FIXED`

Using SQL Plan Management (2/2)

- SQL Plan Management is enabled using:
  - `OPTIMIZER_USE_SQL_PLAN_BASELINES`
- New plans can automatically be stored in the SMB
- New SQL plan baselines have to be evolved before they can be used
- Old SQL plan baselines can be purged automatically
- Note: Only repeatable statements can be auto-captured

Evolving SQL Plan Baselines

- The optimizer only considers accepted SQL plan baselines for execution
- A plan evolves from not accepted to accepted
- Can evolve manually using the `EVOLVE_SQL_PLAN_BASELINE` procedure
- Or, using the SQL Tuning Advisor
  - A SQL profile gets created to enforce the baseline
  - This SQL profile can be accepted automatically
- Note: in Oracle11*g* the SQL Tuning Advisor runs automatically in the maintenance windows

The Plan Selection Process

The optimizer selects a SQL plan baseline based on:

- A plan must be `ENABLED`
- A plan must be `ACCEPTED`
- A plan *can* be `FIXED`
- `FIXED` plans have precedence
- In case of multiple choices:
  - A SQL profile will be taken into account when available
  - The optimizer environment is taken into account
  - The costs are taken into account

- Note: a stored outline takes precedence over a SQL plan baseline

Maintaining SQL Plan Baselines

Using the DBMS_SPM package you can:

- Load new plans into the SMB
  - LOAD_PLANS_FROM_CURSOR_CACHE
  - LOAD_PLANS_FROM_SQLSET
- Maintain existing plans in the SMB
  - EVOLVE_SQL_PLAN_BASELINE
  - ALTER_SQL_PLAN_BASELINE
  - DROP_SQL_PLAN_BASELINE
- Configure the SMB
  - CONFIGURE
- Move plans around
  - CREATE_STGTAB_BASELINE
  - PACK_STGTAB_BASELINE
  - UNPACK_STGTAB_BASELINE

Database Upgrades and SQL Plan Management

Upgrade scenarios:

- **Use current plans in new release**
  - **Create a SQL Tuning Set on the current release**
  - **Upgrade to the new release**
  - **Load plans from the SQL Tuning Set into the SMB**
- **Recreate current plans on new release**
  - **Upgrade to the new release**
  - **Set `OPTIMIZER_FEATURES_ENABLE` to the old release**
  - **Capture plans into the SMB**
  - **Set `OPTIMIZER_FEATURES_ENABLE` to the new release**
- **Migrate existing Stored Outlines to SQL Plan Baselines**
  - **`DBMS_SPM.MIGRATE_STORED_OUTLINE`**

Pros and Cons of SQL Plan Management

Pros:

- SQL regression is prevented
- Controlled by the DBA
- Only improved plans are accepted

Cons:

- Fixed plans prevent new plans from showing up
- Understanding SQL execution becomes more difficult
- SQL plan evolution is a “manual” task

Advise:

- Use it when SQL regression needs to be prevented

Usage Notes and Conclusion

- SQL Plan Management is COOL!
- But, use it only when needed
- And manage the evolution process
- SQL Plan Management supersedes Plan Stability
- Plan Stability will be deprecated in the future
- Migration from Plan Stability is possible
  - Be careful with `FIXED` SQL Plan Baselines!
- Use hints for SQL tuning as a last resort only!

QUESTIONS & ANSWERS

And Finally

Thank you for your kind attention!

For a copy of my demonstration scripts email me at:

Harald.van.Breederode@oracle.com