
ORACLE®

Safe Harbor Statement

The following 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.

Setting up the Oracle Optimizer for PoCs

Covering Oracle Database 12c Release 1 and Release 2
and Oracle Database 18c
May 1st 2018

Nigel Bayliss

Optimizer Product Manager

@vldbb

http://blogs.oracle.com/optimizer

Format

- I will use this slide deck to cover the topic in general
- Please interrupt
  - Interactive is best
  - Use chat to message everyone - if there's any message I will slow down and wait for you to type full question – Chris will keep an eye out for me
- Anything I can't answer will be noted and I will answer using @vldbb, http://blogs.oracle.com/optimizer or later Office Hours
- The Oracle Optimizer might be *smaller* than you think
  - If a question is out of scope, I will say so and leave it at that

General Principles for PoCs

- Start with a *Keep It Simple Stupid* (KISS) baseline
- Prioritize stability and consistency
- Exercise control over change
  - Make changes for a *reason*
- Keep moving

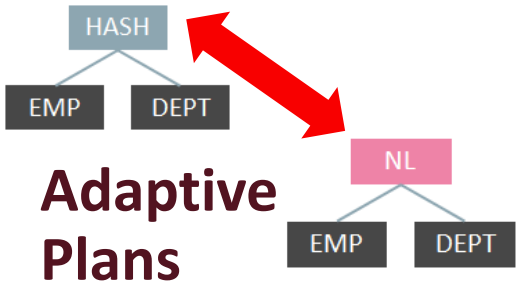
New in Oracle Database 12c Release 1

The Adaptive Optimizer

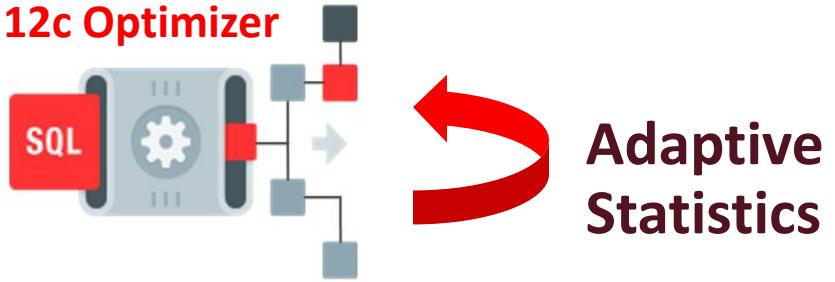
Optimizer Adaptive Features

Change SQL execution plans at runtime

Learn from previous executions and choose better SQL execution plans



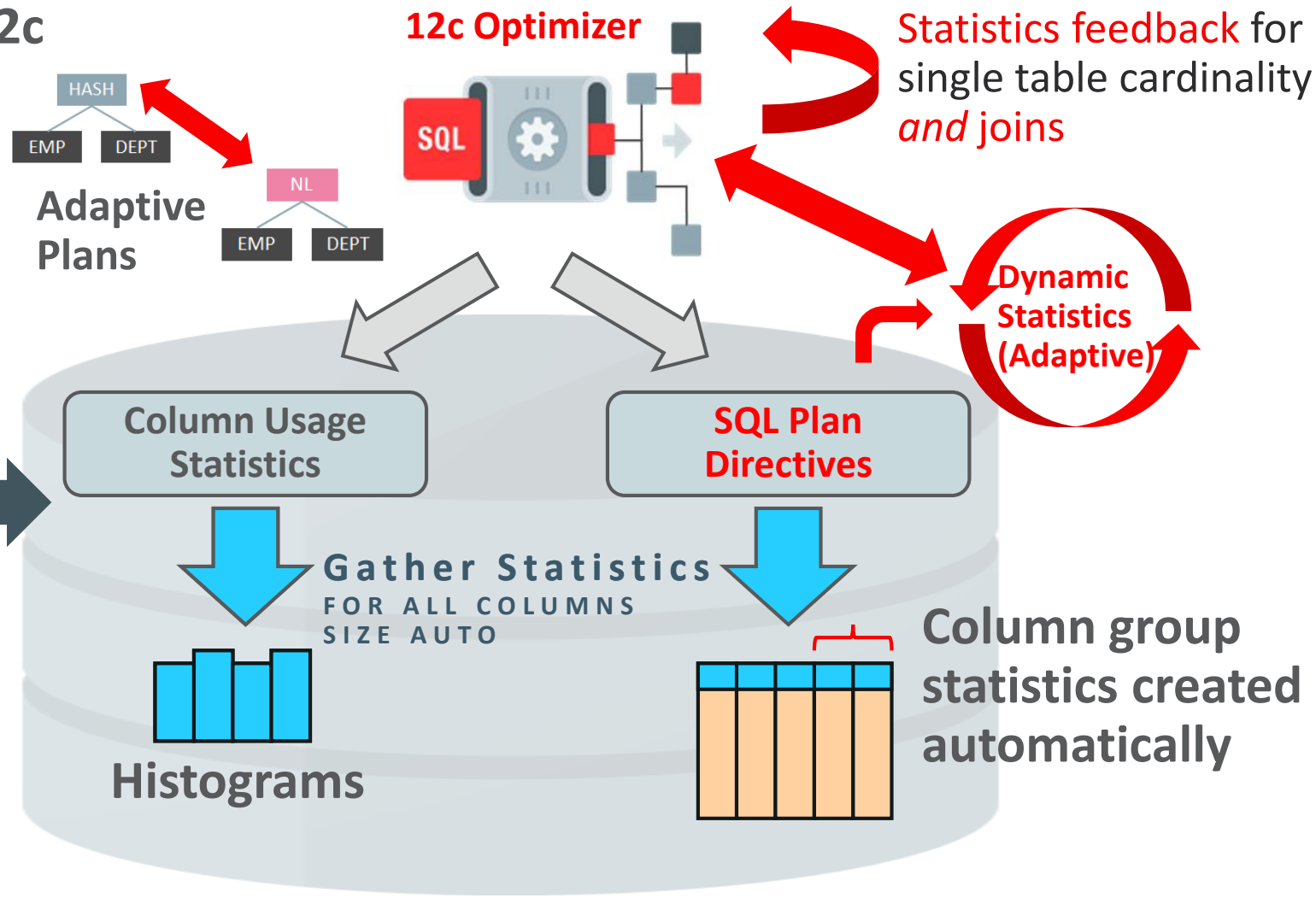
Adaptive Plans



Adaptive Statistics

Oracle Optimizer Adaptive Features

From Oracle Database 12c Release 1



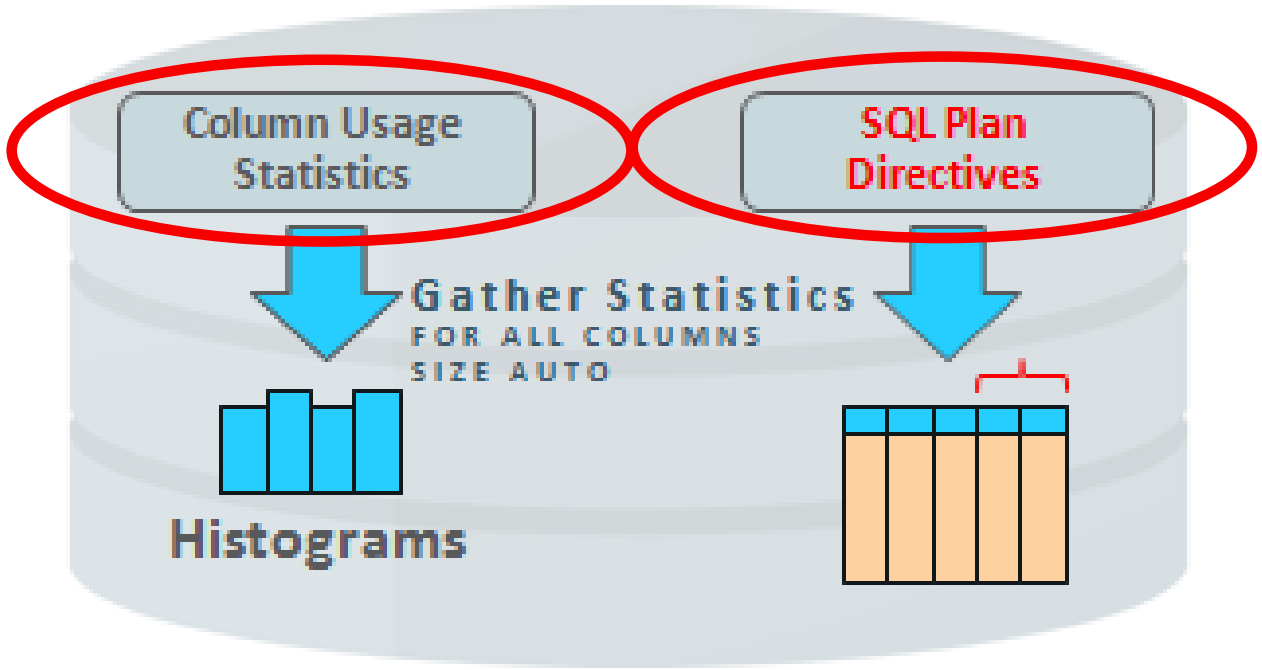
Adaptive Optimizer

Remember…

- The optimizer **adapts over time**
  - SQL execution plans may change over time
  - Performance might not be consistent between runs while it adapts
- Adaptive statistics have an overhead in systems with high hard parse rates
  - High hard parse rates are not best practice, but reality gets in the way
  - More dynamic sampling queries than in Oracle Database 11g
  - Some see library cache contention associated with longer parse times
  - Some see 'RC latch' (result cache) contention in Oracle Database 12c Release 1 (fixed in Oracle Database 12c Release 2)

The Mechanics of Adaption

- **Metadata** is created as workloads are executed
- Metadata affects
  - SQL execution plans
  - Histograms
  - Column group statistics

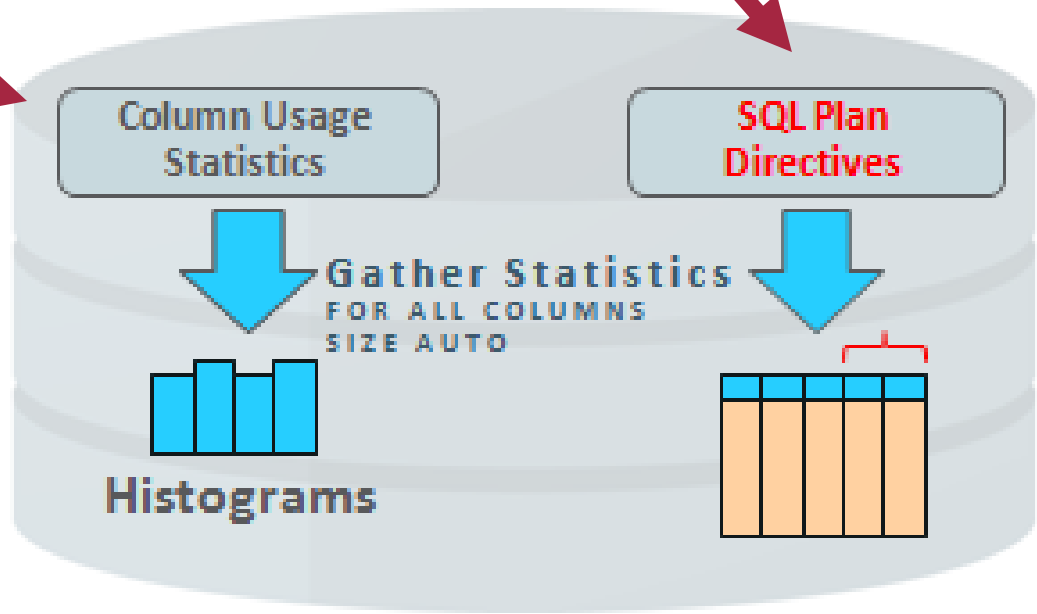


Viewing Optimizer Metadata

SELECT * FROM DBA_SQL_PLAN_DIRECTIVES

```
SELECT DBMS_STATS.REPORT_COL_USAGE('SH,'EMP')
FROM DUAL;
```

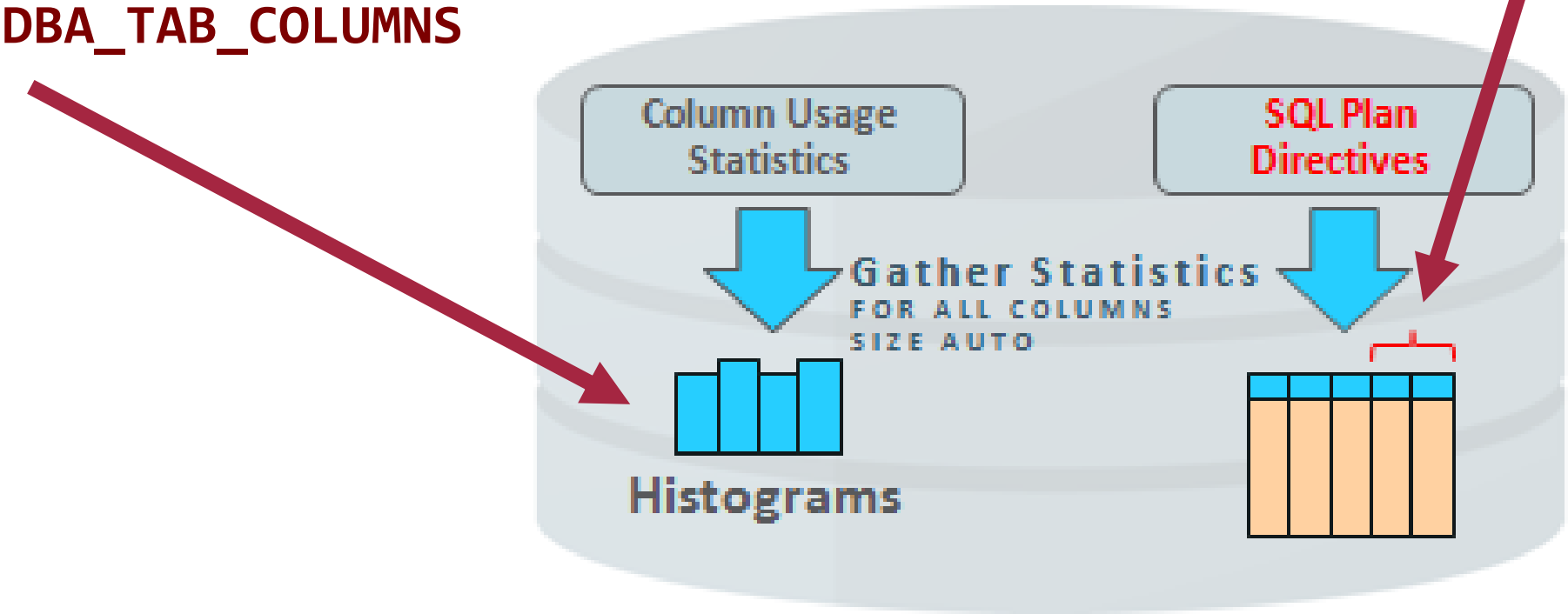
[Also **SYS.COL_USAGE$**]



Seeing the Affects of Optimizer Metadata

```
SELECT *
FROM   DBA_STAT_EXTENSIONS
```

```
SELECT HISTOGRAM
FROM   DBA_TAB_COLUMNS
```



New Histograms and Column Groups

- Column Groups
  - When **gathering statistics** and there are **SQL Plan Directives**
  - Note: this is no longer the default behavior (after 12.1)
- Histograms
  - When **gathering statistics** FOR … COLUMNS **SIZE AUTO** and there is **column usage**
  - This is the default behavior

Adaptive Optimizer Settings

From Oracle Database 12c Release 2

Finer control of adaptive features – new database parameters

~~OPTIMIZER_ADAPTIVE_FEATURES~~ Obsolete

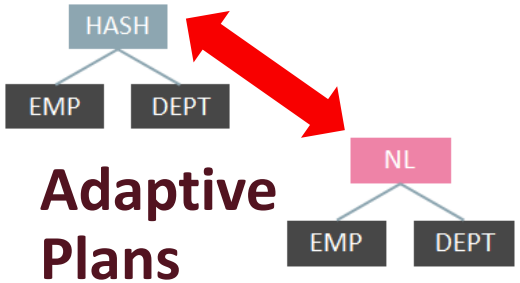
Optimizer Adaptive Features

OPTIMIZER_ADAPTIVE_PLANS

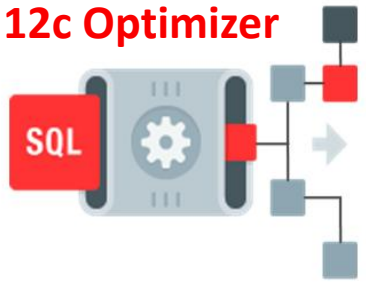
OPTIMIZER_ADAPTIVE_STATISTICS

Change plans at runtime

Learn from previous executions



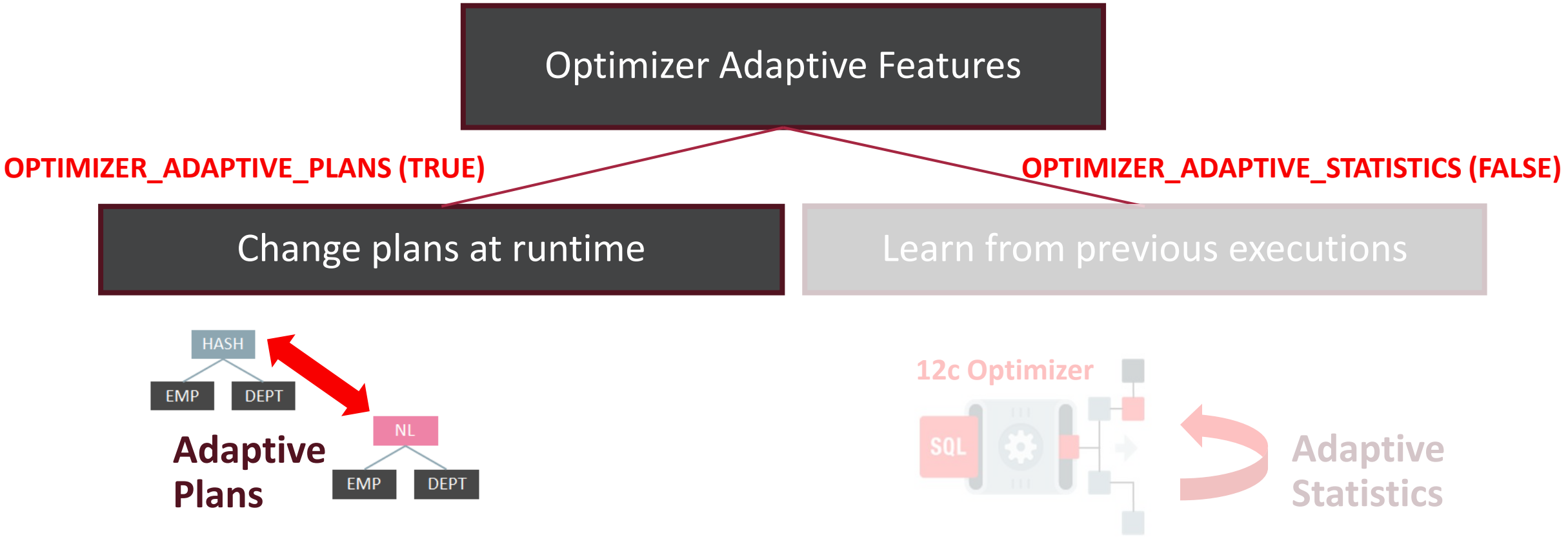
Adaptive Plans



Adaptive Statistics

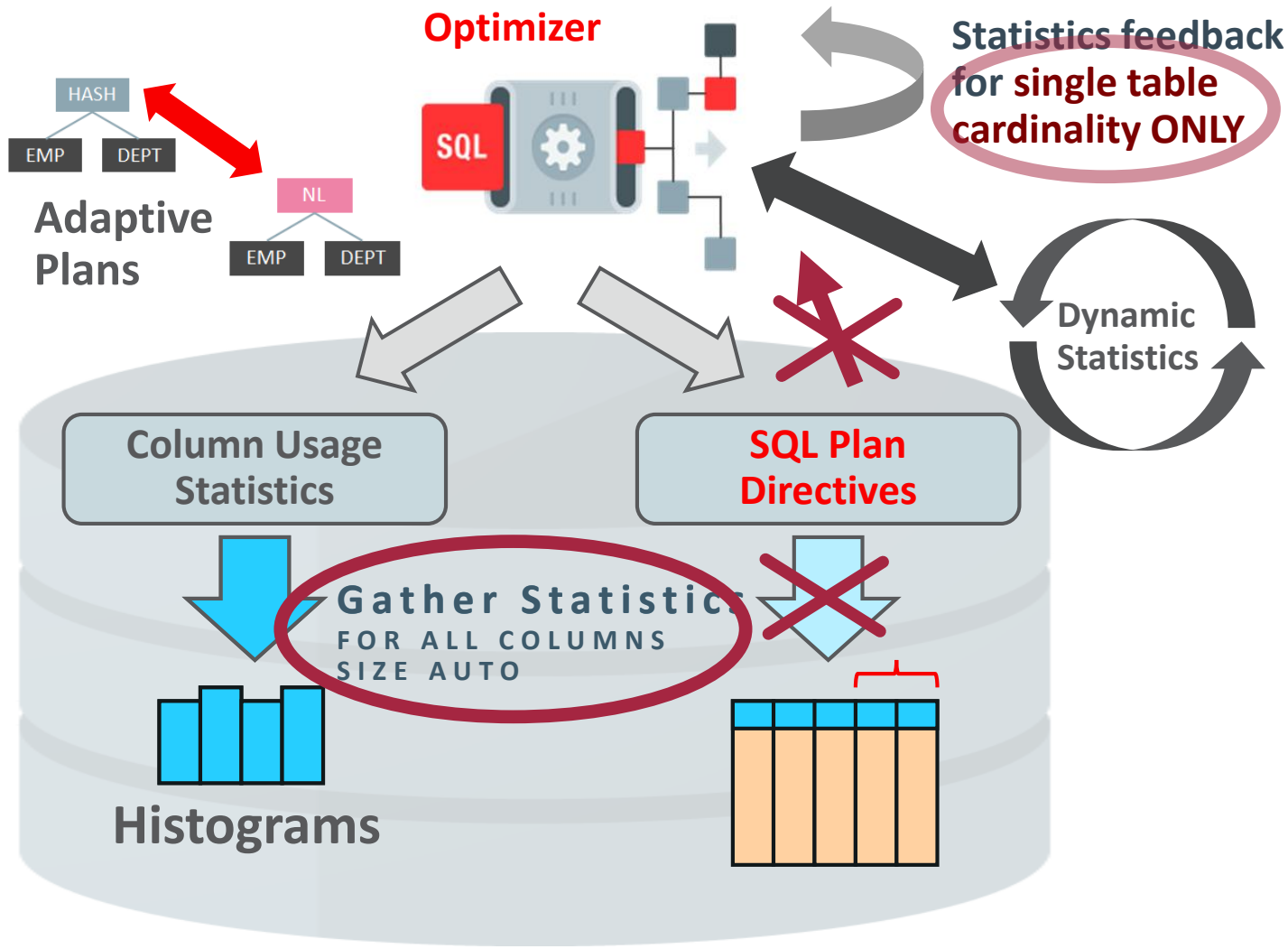
From Oracle Database 12c Release 2

New default behavior



Optimizer Adaptive Features – Recommended Defaults

Oracle Database 12c Release 2 and 18c



The default adaptive feature settings prioritize *consistent* performance

Recommendations for Oracle Database 12c Release 1

Assuming 12.1.0.2

- Install latest **Proactive Bundle Patch** (>=October 2017)
  - e.g. January 2018, Patch 27010930 - Database Proactive Bundle Patch 12.1.0.2.180116
- Recommendations for Adaptive Features in Oracle Database 12c Release 1 (Adaptive Features, Adaptive Statistics and 12c SQL Performance) (**Doc ID 2312911.1**)
  - _fix_control='26664361:7','16732417:1','20243268:1'
- Remove *optimizer_adaptive_features* parameter from pfile/spfile
- Set the following optimizer parameters (these are the defaults in 12cR2 and 18c):
  - **optimizer_adaptive_plans=TRUE**
  - **optimizer_adaptive_statistics=FALSE**
- On the database, execute the following ('OFF' is the default)
  - **EXEC DBMS_STATS.SET_GLOBAL_PREFS('AUTO_STAT_EXTENSIONS','OFF')**

Recommendations for Oracle Database 12c Release 2

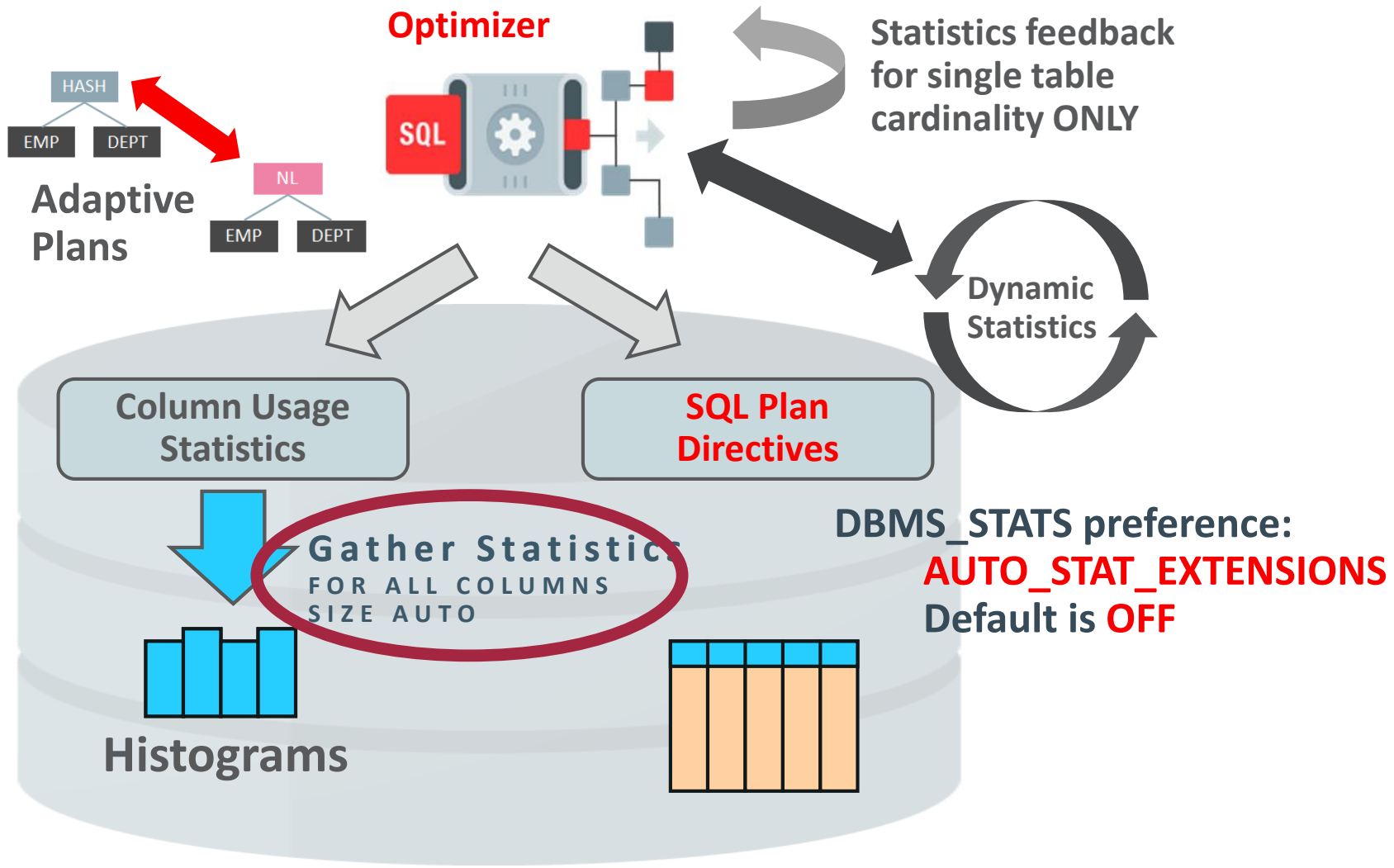
- Do not set the following optimizer parameters - they will take their default values
  - *optimizer_adaptive_plans* [default TRUE]
  - *optimizer_adaptive_statistics* [default FALSE]
- On the database, execute the following (OFF is the default)
  - **EXEC DBMS_STATS.SET_GLOBAL_PREFS('AUTO_STAT_EXTENSIONS','OFF')**

Recommendations for Oracle Database 18c

- Do not set the following optimizer parameters - they will take their default values
  - optimizer_adaptive_plans [default TRUE]
  - optimizer_adaptive_statistics [default FALSE]
- On the database, execute the following (OFF is the default)
  - **EXEC DBMS_STATS.SET_GLOBAL_PREFS('AUTO_STAT_EXTENSIONS','OFF')**

Optimizer Adaptive Features – Recommended Defaults

Oracle Database 12c Release 2 and 18c



Using Adaptive Statistics?

- Adaptive statistics can be useful for DSS/BI workloads
  - Ad-hoc queries
  - Longer-running queries
  - Low hard parse rates
- Start with defaults (adaptive statistics disabled) and you may want to test benefit
- Multiple runs may be required to achieve best performance
  - SQL plan directives created over time
  - **SQL plan directives are created even if optimizer_adaptive_statistics=FALSE – they are just not used**
  - Gathering statistics is required to generate column group stats (and potentially new histograms)
- Setting optimizer_adaptive_statistics = FALSE can be used to restore behavior once test is complete – and remember you may have new histograms/column group stats

Other Recommendations for PoCs

Other Recommendations for Oracle Database 12c and 18c

Optimizer Statistics

- Gather statistics for you application schemas
- Gather dictionary and fixed object statistics
  - DBMS_STATS.GATHER_DICTIONARY_STATS
  - DBMS_STATS.GATHER_FIXED_OBJECTS_STATS

Other Recommendations for Oracle Database 12c and 18c

System Statistics

- If you do not gather system statistics, then delete them and restart the database
- If you have your own preferred settings, you may want to use them, but it is a **good opportunity to use defaults instead**
  - Prior to baseline runs, delete system statistics (as above) – the default settings are appropriate for most systems
  - You *might* want to include a specific test to evaluate the affect of change
- If you find significant differences with vs without, take the time to find out why

Other Recommendations for Oracle Database 12c and 18c

SQL Plan Management

- SQL plan management **evolution** is automated
  - If you are using SQL plan management (SPM), alternative plans may be auto evolved and accepted
  - You might want to disable the auto tuning job or prevent new plans from being accepted…
    - ```
      exec DBMS_SPM.SET_EVOLVE_TASK_PARAMETER(
         task_name => 'SYS_AUTO_SPM_EVOLVE_TASK'
        ,parameter => 'ACCEPT_PLANS'
        ,value     => 'FALSE')
      ```
    - ```
      exec DBMS_AUTO_TASK_ADMIN.DISABLE (client_name => 'sql tuning advisor')
      ```

Other Recommendations for Oracle Database 12c and 18c

Auto statistics gathering

- STALE stats can be more common than usual in a PoC (e.g. changes in column usage)
- You might acquire new histograms (METHOD_OPT=>'FOR ALL COLUMNS SIZE AUTO')
- Controlling stats gathering
  - As a minimum, use the auto stats job to maintain Oracle-owned objects:

```
DBMS_STATS.SET_GLOBAL_PREFS(pname=>'autostats_target',pvalue=>'oracle')
```

  - If necessary, auto stats can be disabled as follows:

```
DBMS_AUTO_TASK_ADMIN.DISABLE(client_name=>'auto optimizer stats collection')
```

  - Manually gather stats:

```
exec dbms_stats.gather_database_stats
```

  - Auto stats can be initiated manually:

```
exec dbms_stats.gather_database_stats_job_proc
```

Other Recommendations for Oracle Database 12c and 18c

Global Temporary Tables

- GTT statistics are session-local by default
  - Pre-12c workload may assume and depend on statistics being shared across sessions
  - If you need pre-12c behaviour in the PoC then…
  - `exec dbms_stats.set_global_prefs('GLOBAL_TEMP_TABLE_STATS','SHARED')`

Regathering Statistics

- You may need to regather statistics at some point
  - E.g. you might want new histograms to match your workload
- If you experience plan regressions, remember that previous stats can be restored to help isolate issue
  - E.g. – DBMS_STATS RESTORE_SCHEMA_STATS, RESTORE_TABLE_STATS etc
- You can test new statistics using pending stats and publish them when tested
  - DBMS_STATS PUBLISH preference
  - Parameter: optimizer_use_pending_statistics
  - DBMS_STATS.PUBLISH_PENDING_STATS

More General Recommendations

Think Holistically

- Using different features and settings may improve performance in some areas and degrade performance elsewhere
  - Consider the overall picture
  - Strike a balance and keep things simple
- Avoid micro managing individual queries and consider the overall benefit/cost
  - Use **Test Case Builder and/or SQLT** to capture sub-optimal plans and move on!
- Use the *service level agreement* to guide what is important and what is not
  - OK, OK, an SLA never exists ☺
  - Is a query exhibiting a performance regression key to the business?

Think Holistically

- Optimizers are not perfect – they use estimates
  - Avoid being fixated by the need to chase apparent 'bugs'
  - Work around query regressions
- SQL tuning advisor will help find better plans if you have SQL statements with performance regressions
- Use plan control and stability features [a later *Office Hours* topic]
- SQL plan management (SPM) can control SQL execution plans
  - Remember you can even test with optimizer_features_enable=<existing/earlier release> and capture the plans with SPM – then run with OFA=<latest release>
  - You can use SPM fix individual queries

Think Holistically

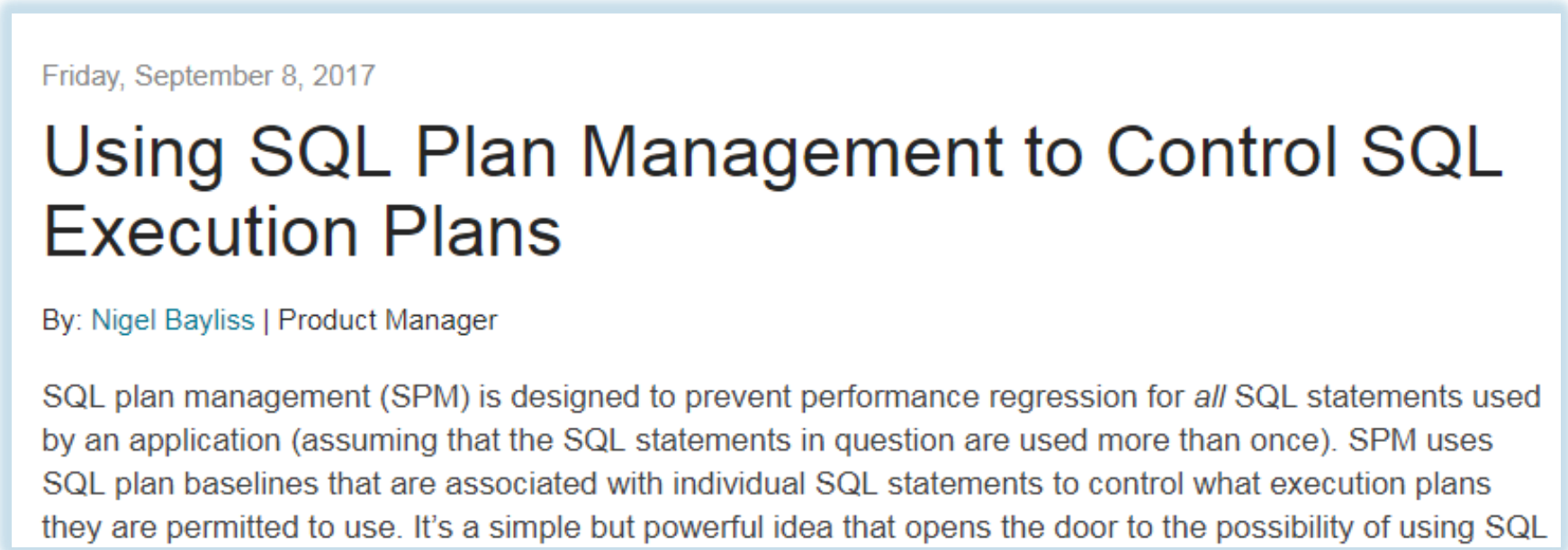
- Over-hinted SQL?
  - _optimizer_ignore_hints=TRUE
  - SQL Patch [see later]

Summary

- Start with a KISS baseline
  - Latest BP + Defaults
  - Gather statistics at the start of the PoC to gain a solid initial baseline
  - Remember you can restore statistics and use pending statistics if necessary
- Prioritize stability and consistency
  - Use the most recent optimizer adaptive feature defaults
  - Remember: the optimizer adapts over time if you use non-defaults
- Exercise control over change
  - Be aware of auto jobs and disable where appropriate
- Keep Moving
  - Work around problems, save for later
  - Plan stability features (e.g. SPM, SQL Patch), SQL Tuning Advisor, TCB, SQLT

More

- http://blogs.oracle.com/optimizer
- SPM
  https://blogs.oracle.com/optimizer/using-sql-plan-management-to-control-sql-execution-plans



Friday, September 8, 2017

Using SQL Plan Management to Control SQL Execution Plans

By: Nigel Bayliss | Product Manager

SQL plan management (SPM) is designed to prevent performance regression for *all* SQL statements used by an application (assuming that the SQL statements in question are used more than once). SPM uses SQL plan baselines that are associated with individual SQL statements to control what execution plans they are permitted to use. It's a simple but powerful idea that opens the door to the possibility of using SQL

More

- Test Case Builder to capture a test case for later diagnosis
  https://docs.oracle.com/en/database/oracle/oracle-database/18/tgsql/sql-test-case-builder.html

More

- SQL Patch to patch individual SQL statements
  https://blogs.oracle.com/optimizer/adding-and-disabling-hints-using-sql-patch

Monday, June 12, 2017

Adding and Disabling Hints Using SQL Patch

By: Nigel Bayliss | Product Manager

If you're a DBA, it's likely that you've encountered systems where a lot of SQL statements have been hinted almost as a matter of policy. Perhaps you'd like to figure out if these hints are actually helping. You might like to demonstrate to a development team that they should probably dial down their enthusiasm for micro-managing the Oracle Optimizer. Sometimes, you might want to apply a hints on-the-fly.

A while ago, Maria Colgan wrote a couple of posts (here and here) on SQL Patch and how you can add hints to SQL in a packaged application. In other words you can apply hints to SQL statements without having to change any application code. From Oracle Database 12c Release 2, the interface to SQL Patch is greatly improved and easier to use. In particular, it's now part of the public API and the hint text is a CLOB (because VARCHAR2 can be too limiting if you want to specify a complete query outline). The API includes a new SQL_ID parameter too. Check out the documentation for the details, but here's an example of the new look:

More

- SQLT to capture query information for later diagnosis
  https://blogs.oracle.com/performancediagnosis/sqlt-version-122171004

Safe Harbor Statement

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.

Integrated Cloud

Applications & Platform Services


ORACLE®