SQL Plan Management on 12c

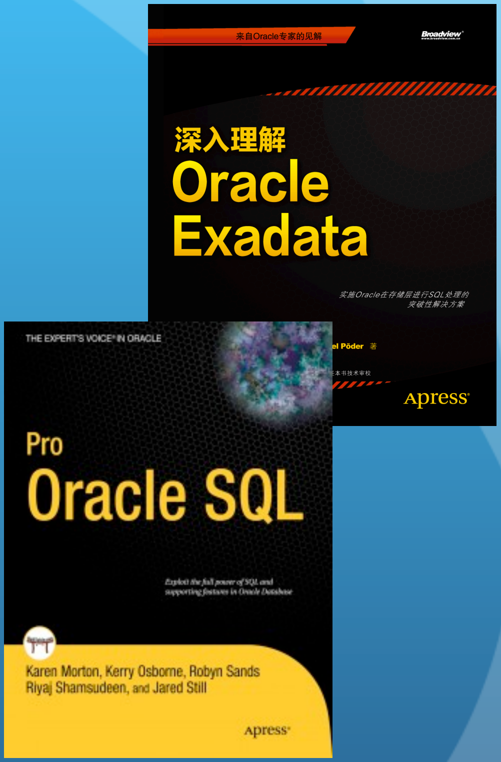
Kerry Osborne

OakTable World, 2013

enkitec

whoami -

Never Worked for Oracle
Worked with Oracle DB Since 1982 (V2)
Working with Exadata since early 2010
Work for Enkitec (www.enkitec.com)
(Enkitec owns several Exadata's – V2/X2/X3)
(And Others BDA, Exalytics, ODA, etc…)
Worked on a couple of books
Hadoop Aficionado
Exadata Fan Boy
12c Novice?


深入理解
Oracle
Exadata
Apress
Pro
Oracle SQL
Karen Morton, Kerry Osborne, Robyn Sands
Riyaj Shamsudeen, and Jared Still
Apress

Blog: kerryosborne.oracle-guy.com
Twitter: @KerryOracleGuy

After Lunch Sessions

SQL Plan Management

Framework
Designed to prevent performance regression
Not heavily adopted in 11g
Uses Baselines
- Signature (based on normalized sql text)
- Hints
- Plan_Hash_Value
- Flags

enkitec

SQL Baselines

Fully Baked (almost)

- Goal was to prevent performance regression
- (Closer to Outlines than to SQL Profiles)
- Enabled by default in 11g (optimizer_use_sql_plan_baselines)
- Capable of applying any valid hints
- * Has associated plan_hash_value
- Invalid hints are NOT silently ignored!
- Provides procedure to import plans (DBMS_SPM.LOAD_PLANS_FROM_CURSOR_CACHE)
- Overridden by Outlines
- Can work with Profiles and Patches (merges hints)
- Can have multiple Baselines per statement
- No Categories
- Preferred Set (fixed=yes)

enkitec

Terminology

SQL Plan Management (SPM)
- Framework

Plan History
- Set of plans generated by the CBO

SQL Plan Baseline
- Set of "accepted" plans for a SQL statement
- Also commonly used for a single plan (a baseline)

Plan Evolution
- Process of adding ACCEPTED plans to the SQL Plan Baseline

SQL Management Base (SMB)
- Part of the Dictionary that stores plans
  - Plans History, SQL Plan Baseline and SQL Profiles

enkitec

Flags

Enabled
Accepted
Fixed
Reproduced
Adaptive – new in 12c

Fixed?

No they are not broken

Fixed=Preferred

Multiple plans can be fixed

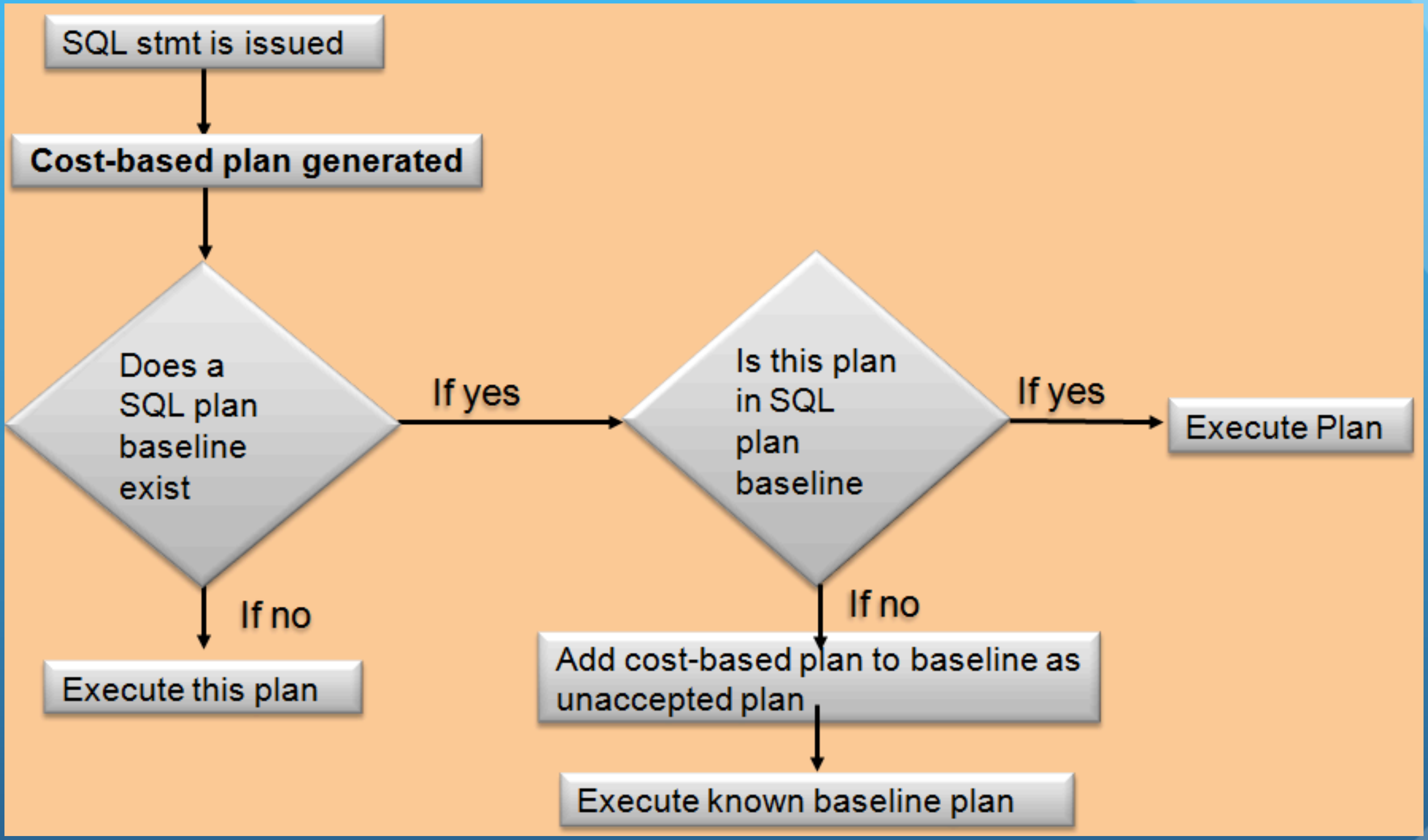
Optimizer will check all fixed plans before non-fixed

Also causes interesting side effect

- no new plans are added
- not expected to evolve (i.e. “fixed”)

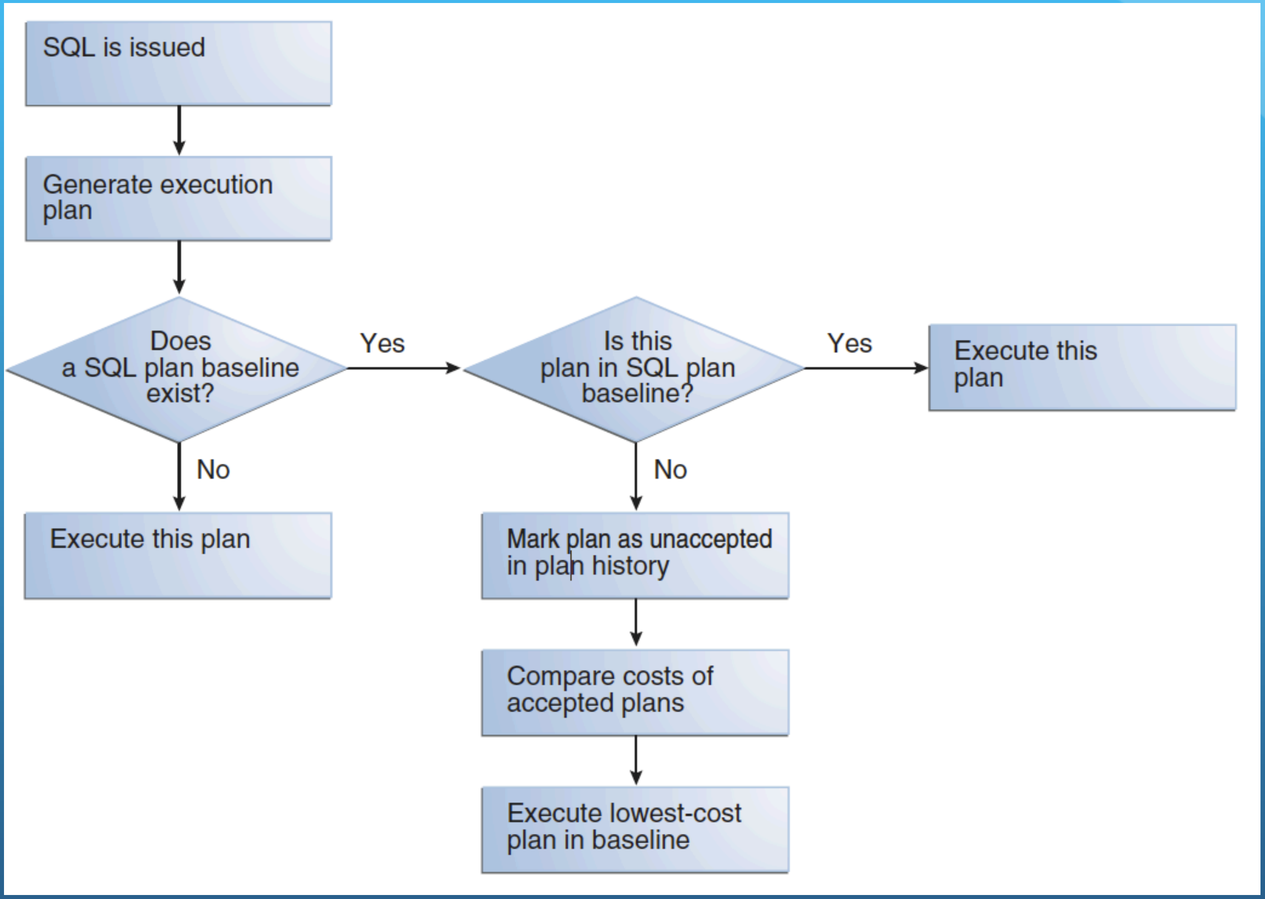
enkitec

Documented SPM Decision Tree


SQL stmt is issued
Cost-based plan generated
Does a SQL plan baseline exist
If yes
Is this plan in SQL plan baseline
If yes
Execute Plan
If no
Execute this plan
If no
Add cost-based plan to baseline as unaccepted plan
Execute known baseline plan

* Assuming no fixed plans

Or Occasionally Like This


SQL is issued
Generate execution plan
Does a SQL plan baseline exist?
Yes
No
Execute this plan
Is this plan in SQL plan baseline?
Yes
Execute this plan
No
Mark plan as unaccepted in plan history
Compare costs of accepted plans
Execute lowest-cost plan in baseline

* Assuming no fixed plans

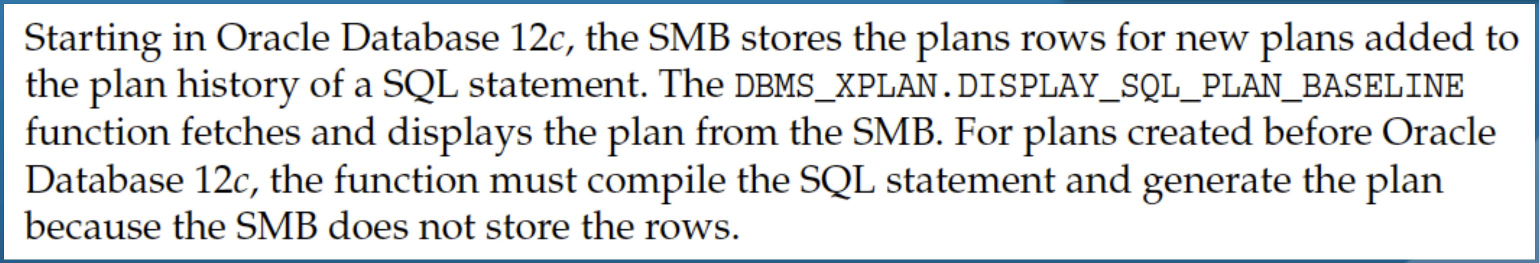
12c Good News / Bad News

Good News

- SPM Now stores the actual plan
- Can see non-reproducible plans

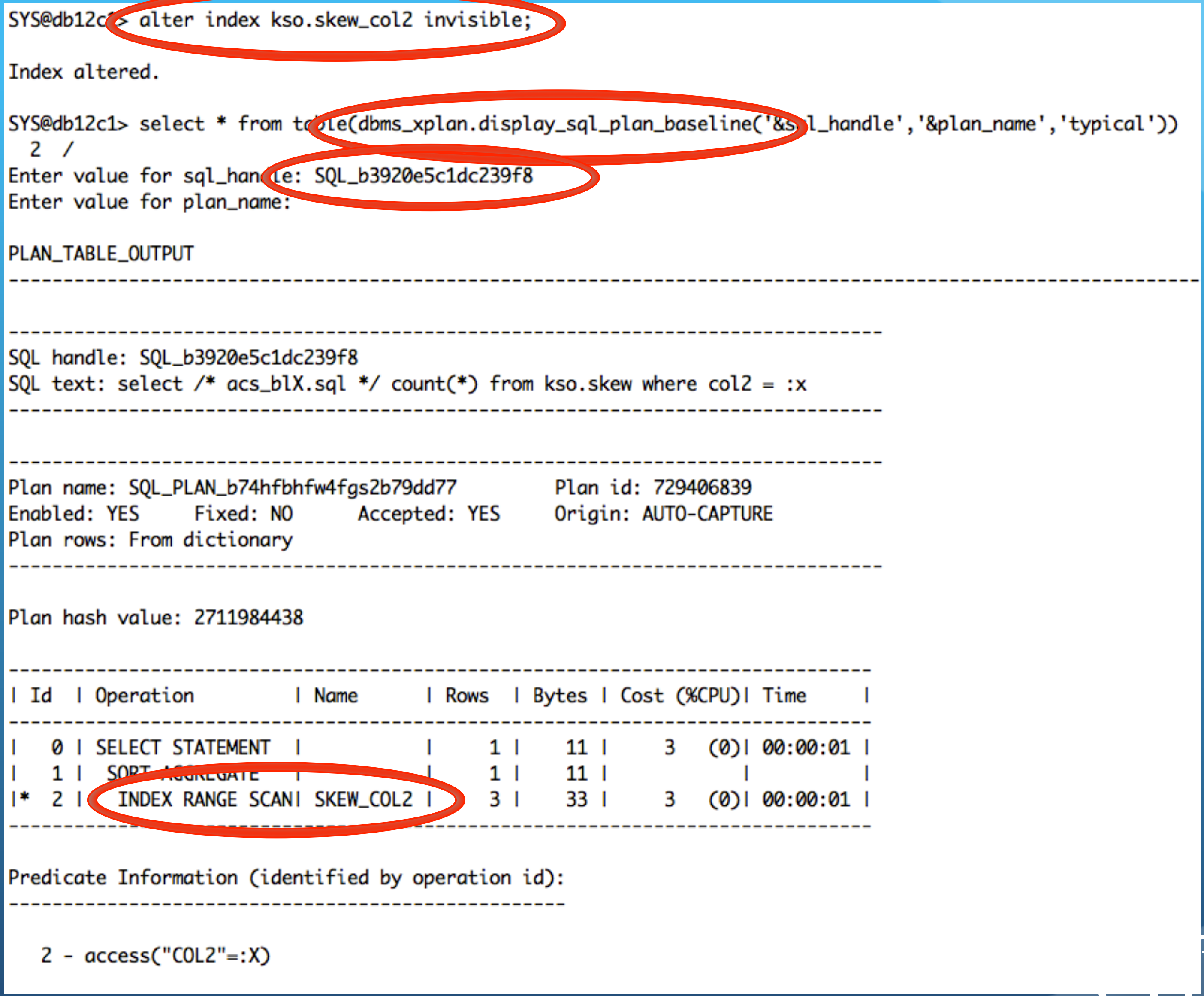
Bad News

- Still Uses Hints



Starting in Oracle Database 12*c*, the SMB stores the plans rows for new plans added to the plan history of a SQL statement. The `DBMS_XPLAN.DISPLAY_SQL_PLAN_BASELINE` function fetches and displays the plan from the SMB. For plans created before Oracle Database 12*c*, the function must compile the SQL statement and generate the plan because the SMB does not store the rows.

Display Non-Reproducible Plan



```
SYS@db12c1> alter index kso.skew_col2 invisible;

Index altered.

SYS@db12c1> select * from table(dbms_xplan.display_sql_plan_baseline('&sql_handle','&plan_name','typical'))
  2  /
Enter value for sql_handle: SQL_b3920e5c1dc239f8
Enter value for plan_name:

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------
SQL handle: SQL_b3920e5c1dc239f8
SQL text: select /* acs_blX.sql */ count(*) from kso.skew where col2 = :x
--------------------------------------------------------------------------------

--------------------------------------------------------------------------------
Plan name: SQL_PLAN_b74hfbhfw4fgs2b79dd77         Plan id: 729406839
Enabled: YES      Fixed: NO       Accepted: YES      Origin: AUTO-CAPTURE
Plan rows: From dictionary
--------------------------------------------------------------------------------

Plan hash value: 2711984438

-------------------------------------------------------------------------------
| Id  | Operation         | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |           |     1 |    11 |     3   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE   |           |     1 |    11 |            |          |
|*  2 |   INDEX RANGE SCAN| SKEW_COL2 |     3 |    33 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - access("COL2"=:X)
```

Automatic Capture?

Upside:
- Improves Stability
- Keeps Repository of Potential Plans
- Makes it Easy to Move Plans Between Systems

Downside:
- Accepts first plan that comes along
    - Can disable some other features (ACS, etc…)
- Don't want to start capture until system is stable
- There is overhead

enkitec

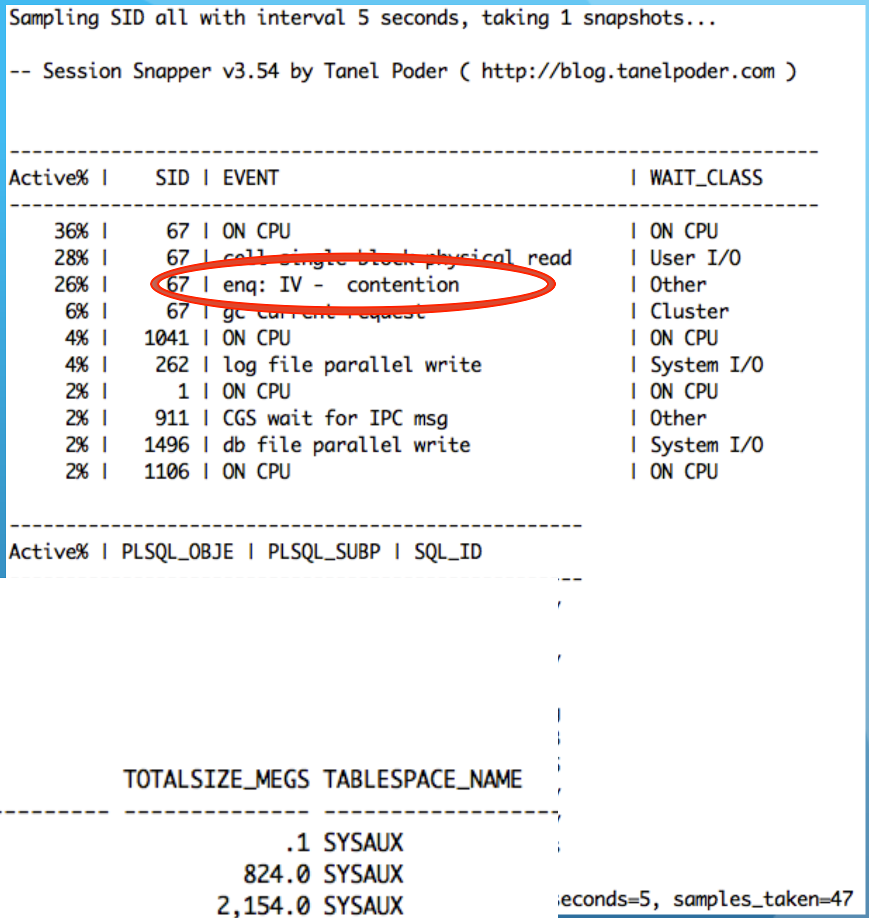
Automatic Capture?

OPTIMIZER_CAPTURE_SQL_PLAN_ BASELINES=TRUE
Still not the default in 12c
  - still not the preferred approach?

Note: Once a Baseline is created on a statement
  new plans will be added even if capture is off

Overhead?

There is some
- Extra parsing
- Size of the SQLOBJ$AUXDATA



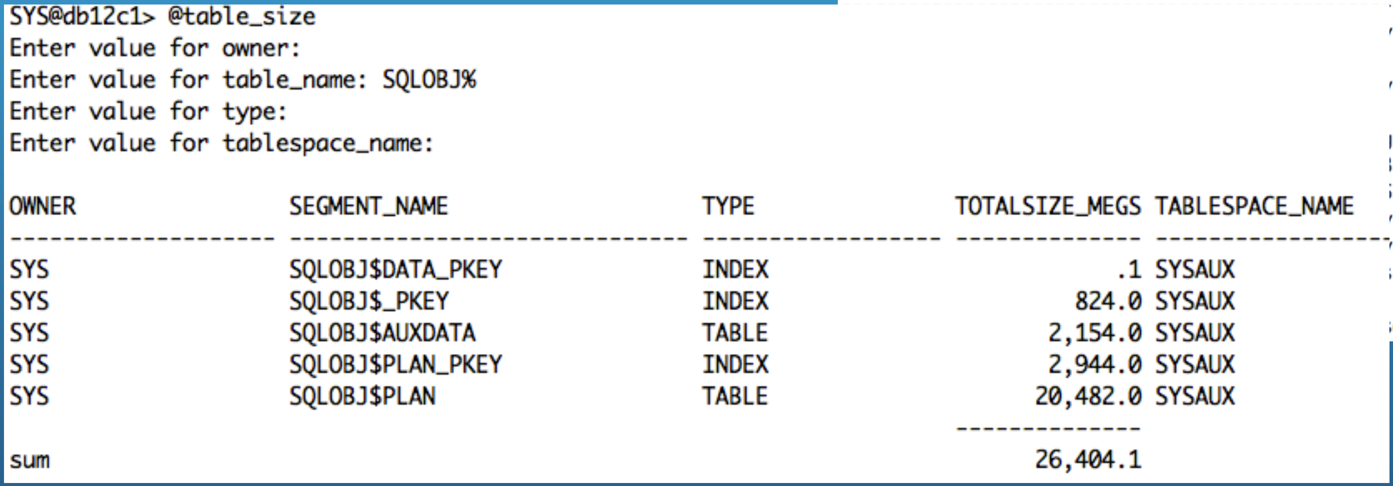
```
Sampling SID all with interval 5 seconds, taking 1 snapshots...

-- Session Snapper v3.54 by Tanel Poder ( http://blog.tanelpoder.com )


---------------------------------------------------------------------------------
Active% |    SID | EVENT                                    | WAIT_CLASS
---------------------------------------------------------------------------------
    36% |     67 | ON CPU                                   | ON CPU
    28% |     67 | cell single block physical read          | User I/O
    26% |     67 | enq: IV -  contention                    | Other
     6% |     67 | gc current request                       | Cluster
     4% |   1041 | ON CPU                                   | ON CPU
     4% |    262 | log file parallel write                  | System I/O
     2% |      1 | ON CPU                                   | ON CPU
     2% |    911 | CGS wait for IPC msg                     | Other
     2% |   1496 | db file parallel write                   | System I/O
     2% |   1106 | ON CPU                                   | ON CPU

----------------------------------------------------------
Active% | PLSQL_OBJE | PLSQL_SUBP | SQL_ID

econds=5, samples_taken=47
```

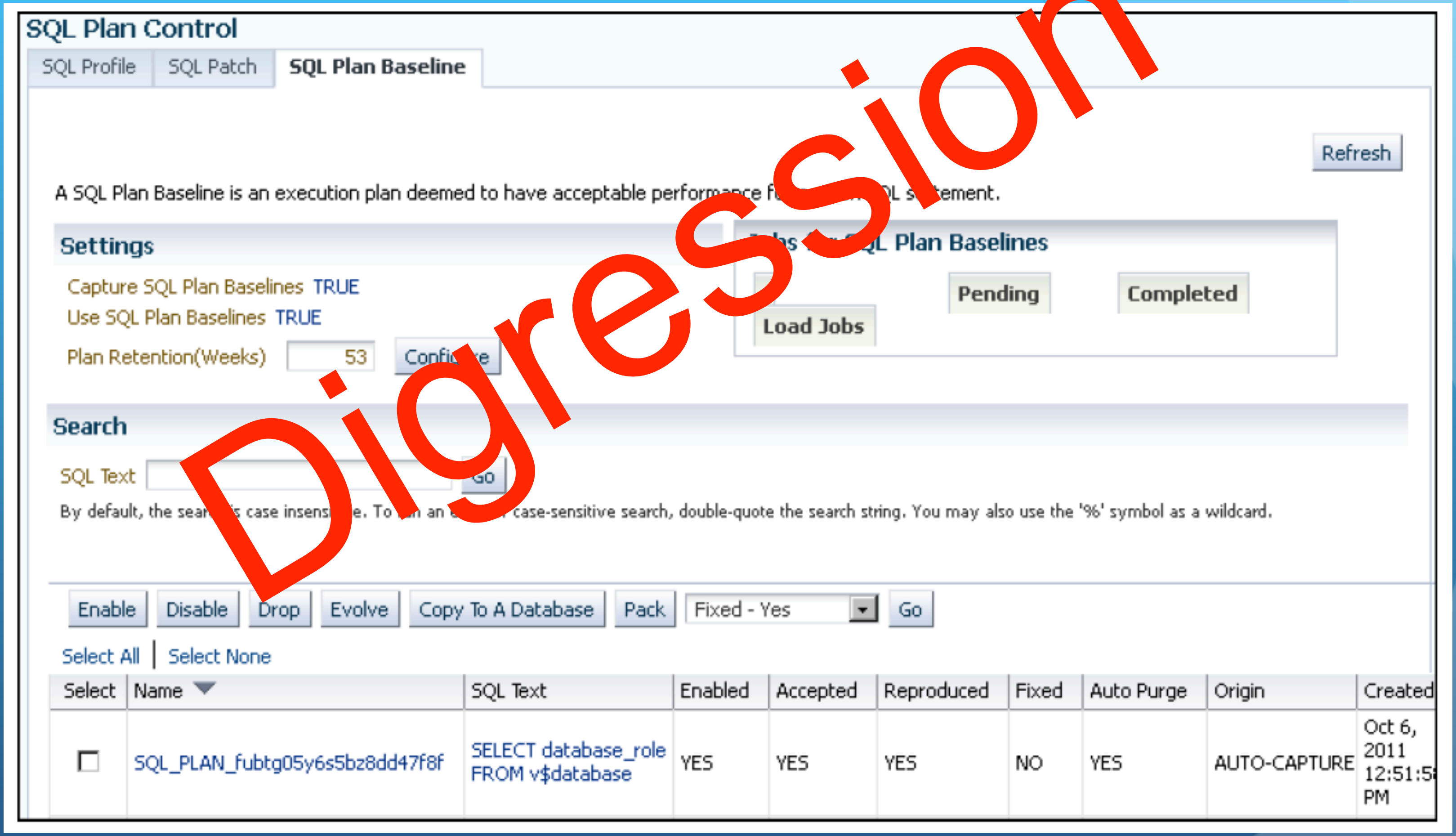


```
SYS@db12c1> @table_size
Enter value for owner:
Enter value for table_name: SQLOBJ%
Enter value for type:
Enter value for tablespace_name:

OWNER                SEGMENT_NAME                   TYPE               TOTALSIZE_MEGS TABLESPACE_NAME
-------------------- ------------------------------ ------------------ -------------- ------------------
SYS                  SQLOBJ$DATA_PKEY               INDEX                          .1 SYSAUX
SYS                  SQLOBJ$_PKEY                   INDEX                       824.0 SYSAUX
SYS                  SQLOBJ$AUXDATA                 TABLE                     2,154.0 SYSAUX
SYS                  SQLOBJ$PLAN_PKEY               INDEX                     2,944.0 SYSAUX
SYS                  SQLOBJ$PLAN                    TABLE                    20,482.0 SYSAUX
                                                                       --------------
sum                                                                          26,404.1
```

enkitec

OEM Interface



SQL Plan Control

SQL Profile | SQL Patch | **SQL Plan Baseline**

Refresh

A SQL Plan Baseline is an execution plan deemed to have acceptable performance f[illegible] SQL statement.

Settings

Capture SQL Plan Baselines	TRUE
Use SQL Plan Baselines	TRUE
Plan Retention(Weeks)	53

[illegible] SQL Plan Baselines

Load Jobs | Pending | Completed

Search

SQL Text

By default, the sear[illegible] is case insens[illegible]e. To [illegible] an [illegible] case-sensitive search, double-quote the search string. You may also use the '%' symbol as a wildcard.

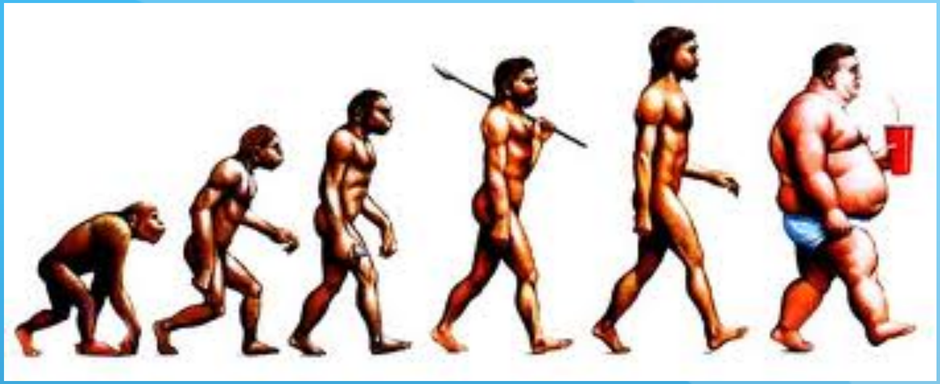
Enable | Disable | Drop | Evolve | Copy To A Database | Pack | Fixed - Yes | Go

Select All | Select None

Select	Name	SQL Text	Enabled	Accepted	Reproduced	Fixed	Auto Purge	Origin	Created
☐	SQL_PLAN_fubtg05y6s5bz8dd47f8f	SELECT database_role FROM v$database	YES	YES	YES	NO	YES	AUTO-CAPTURE	Oct 6, 2011 12:51:5 PM

enkitec

Plan Evolution



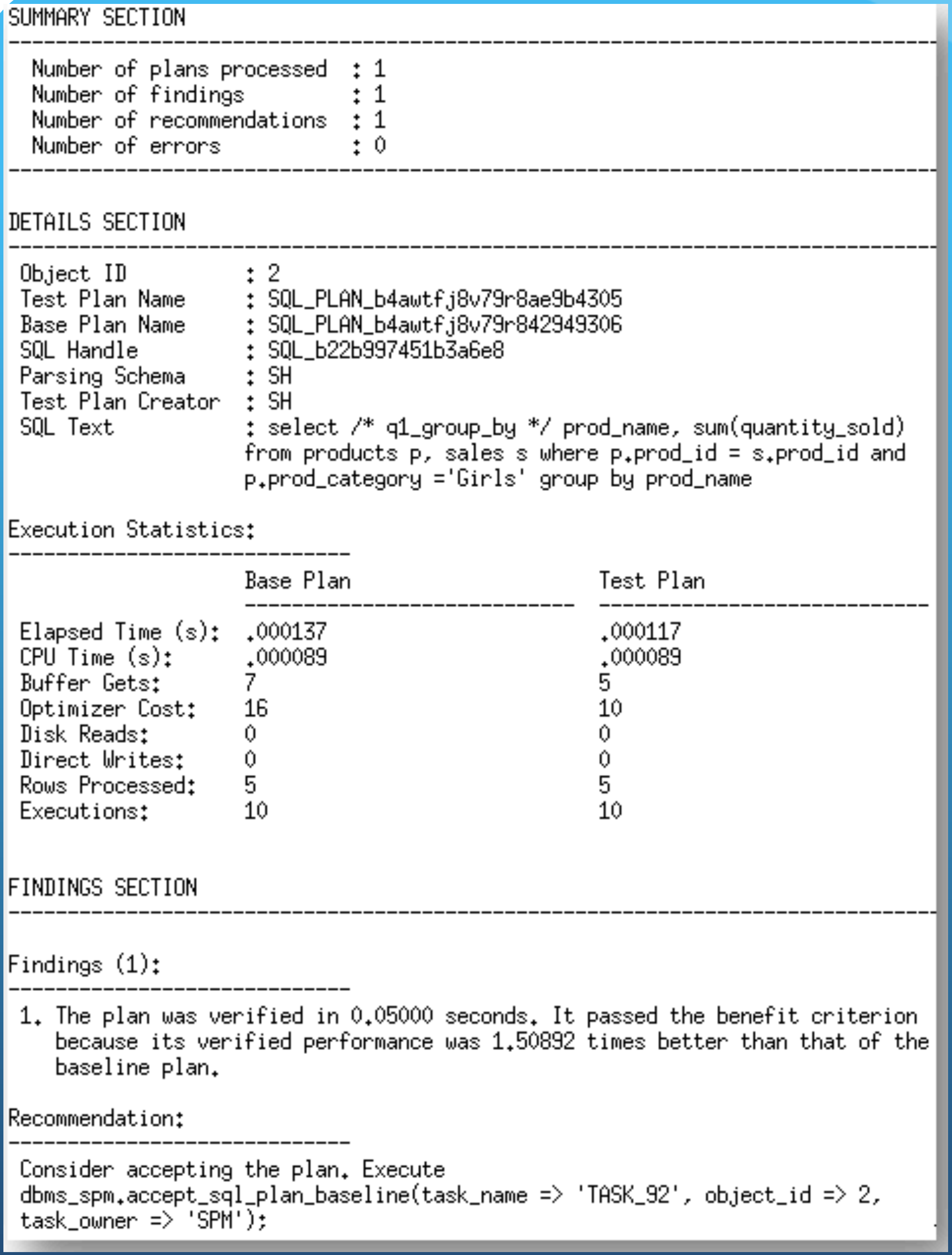
Step One – Collect Plans
Step Two – Evaluate Plans
Step Three – Accept Better Plans

Plan Evolution – 12c

New SPM Evolve Advisor
New AutoTask - SYS_AUTO_SPM_EVOLVE_TASK
Enabled By Default
Attempts to Verify New Plans Every Night
Can auto evolve (1.5x improvement)
 - Improvement (based on etime, lio, cpu time)
Non-Accepted not tried again for 30 days
Produces Report

Plan Evolution – 12c

DBMS_SPM.REPORT_AUTO_EVOLVE_TASK



```
SUMMARY SECTION
---------------------------------------------------------------------------------------------
  Number of plans processed  : 1
  Number of findings         : 1
  Number of recommendations  : 1
  Number of errors           : 0
---------------------------------------------------------------------------------------------

DETAILS SECTION
---------------------------------------------------------------------------------------------
 Object ID          : 2
 Test Plan Name     : SQL_PLAN_b4awtfj8v79r8ae9b4305
 Base Plan Name     : SQL_PLAN_b4awtfj8v79r842949306
 SQL Handle         : SQL_b22b997451b3a6e8
 Parsing Schema     : SH
 Test Plan Creator  : SH
 SQL Text           : select /* q1_group_by */ prod_name, sum(quantity_sold)
                    from products p, sales s where p.prod_id = s.prod_id and
                    p.prod_category ='Girls' group by prod_name

Execution Statistics:
-----------------------------
                    Base Plan                      Test Plan
                    ----------------------------   ----------------------------
 Elapsed Time (s):  .000137                        .000117
 CPU Time (s):      .000089                        .000089
 Buffer Gets:       7                              5
 Optimizer Cost:    16                             10
 Disk Reads:        0                              0
 Direct Writes:     0                              0
 Rows Processed:    5                              5
 Executions:        10                             10


FINDINGS SECTION
---------------------------------------------------------------------------------------------

Findings (1):
-----------------------------
 1. The plan was verified in 0.05000 seconds. It passed the benefit criterion
    because its verified performance was 1.50892 times better than that of the
    baseline plan.

Recommendation:
-----------------------------
 Consider accepting the plan. Execute
 dbms_spm.accept_sql_plan_baseline(task_name => 'TASK_92', object_id => 2,
 task_owner => 'SPM');
```

Manual Plan Evolution – 12c

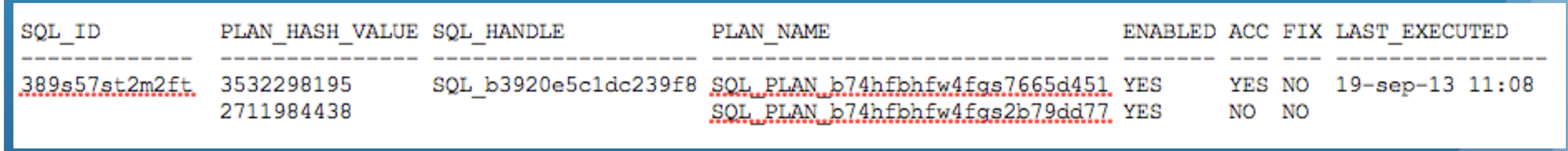
New Task Oriented Approach
Much Like SQL Tuning Advisor
- Create Evolve Task (dbms_spm.create_evolve_task)
- Execute Evolve Task (dbms_spm.execute_evolve_task)
- Report Evolve Task (dbms_spm.report_evolve_task)
- Accept Recommendation (dbms_spm.accept_sql_plan_baseline)

Note: DBMS_SPM.EVOLVE_SQL_PLAN_BASELINE deprecated

enkitec

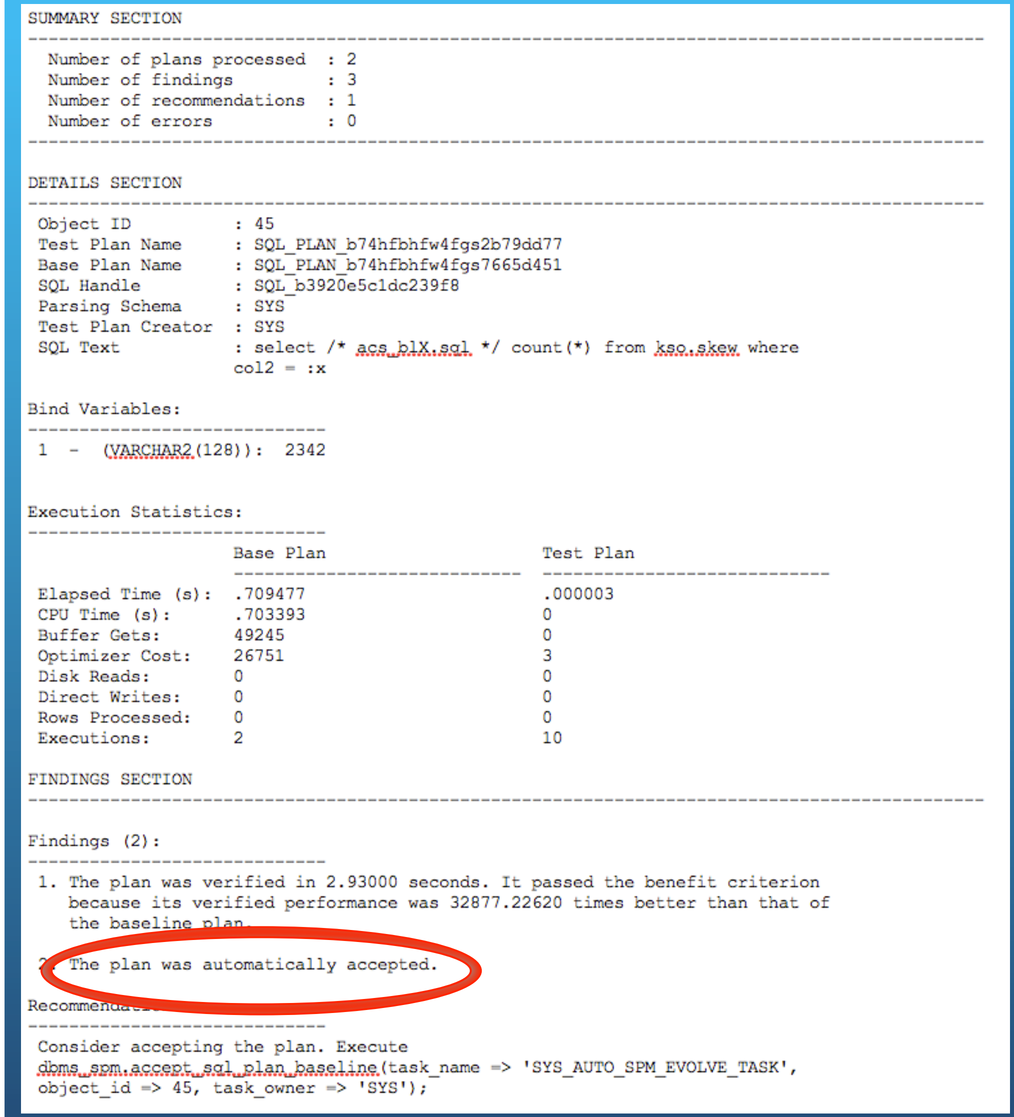
Interaction with Adaptive Cursor Sharing (ACS)

ACS – “fix” for bind variable peeking
ACS – allows multiple execution plans per statement
Capture automatically accepts first plan
But Baselines allow multiple plans per statement as well
This means Plans must be evolved in order to work well with ACS
Otherwise you end up with this



```
SQL_ID        PLAN_HASH_VALUE SQL_HANDLE           PLAN_NAME                      ENABLED ACC FIX LAST_EXECUTED
------------- --------------- -------------------- ------------------------------ ------- --- --- ---------------
389s57st2m2ft 3532298195      SQL_b3920e5c1dc239f8 SQL_PLAN_b74hfbhfw4fgs7665d451 YES     YES NO  19-sep-13 11:08
              2711984438                           SQL_PLAN_b74hfbhfw4fgs2b79dd77 YES     NO  NO
```

Interaction with Adaptive Cursor Sharing (ACS)



```
SUMMARY SECTION
---------------------------------------------------------------------------------------------
  Number of plans processed  : 2
  Number of findings         : 3
  Number of recommendations  : 1
  Number of errors           : 0
---------------------------------------------------------------------------------------------

DETAILS SECTION
---------------------------------------------------------------------------------------------
 Object ID          : 45
 Test Plan Name     : SQL_PLAN_b74hfbhfw4fgs2b79dd77
 Base Plan Name     : SQL_PLAN_b74hfbhfw4fgs7665d451
 SQL Handle         : SQL_b3920e5c1dc239f8
 Parsing Schema     : SYS
 Test Plan Creator  : SYS
 SQL Text           : select /* acs_blX.sql */ count(*) from kso.skew where
                      col2 = :x

Bind Variables:
-----------------------------
 1  -  (VARCHAR2(128)):  2342


Execution Statistics:
-----------------------------
                    Base Plan                     Test Plan
                    ----------------------------  ----------------------------
 Elapsed Time (s):  .709477                       .000003
 CPU Time (s):      .703393                       0
 Buffer Gets:       49245                         0
 Optimizer Cost:    26751                         3
 Disk Reads:        0                             0
 Direct Writes:     0                             0
 Rows Processed:    0                             0
 Executions:        2                             10

FINDINGS SECTION
---------------------------------------------------------------------------------------------

Findings (2):
-----------------------------
 1. The plan was verified in 2.93000 seconds. It passed the benefit criterion
    because its verified performance was 32877.22620 times better than that of
    the baseline plan.

 2. The plan was automatically accepted.

Recommendation:
-----------------------------
 Consider accepting the plan. Execute
 dbms_spm.accept_sql_plan_baseline(task_name => 'SYS_AUTO_SPM_EVOLVE_TASK',
 object_id => 45, task_owner => 'SYS');
```

12c auto evolve task

Interaction with Adaptive Optimization

Digression

What's adaptive optimization?

What’s the Point?

(of Adaptive Optimization)

Sometimes the Optimizer Makes Mistakes
It’s Often Pretty Easy to Spot the Mistakes
Why Not Let the DB Fix the Mistakes on the Fly?

enkitec

How Does the Optimizer Mess Up?

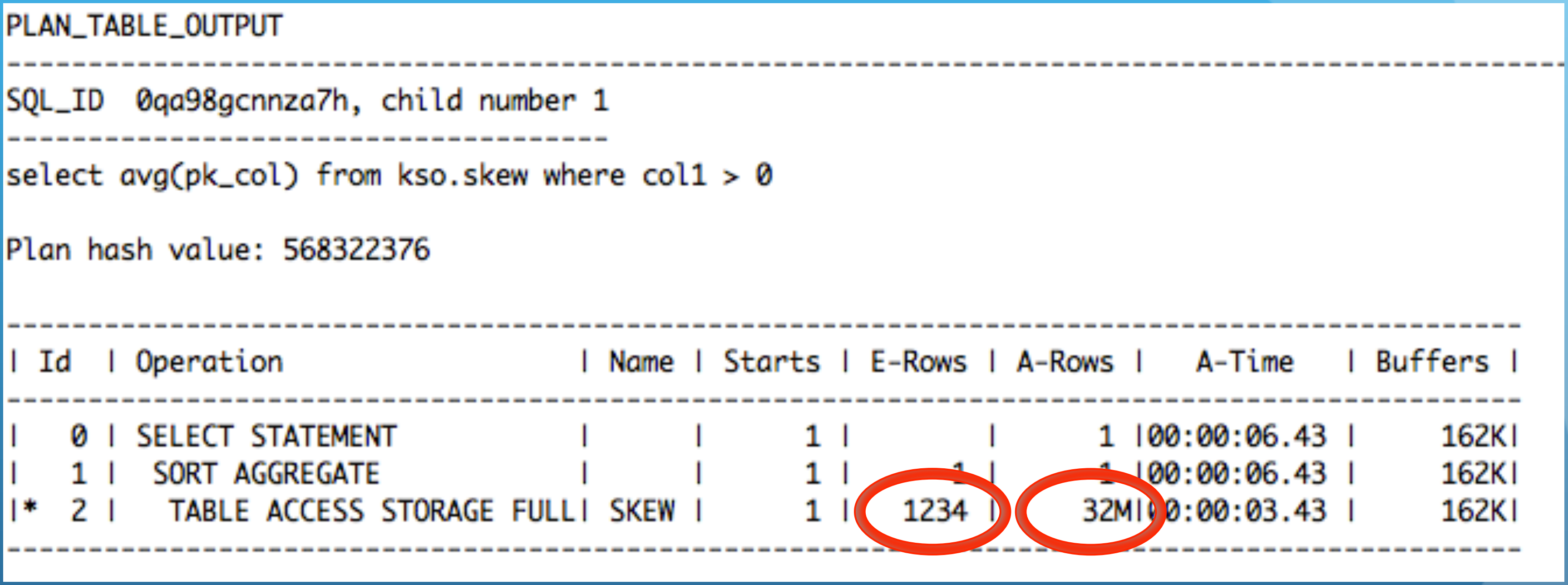
Cardinality – Misunderestimate

mostly …
and it’s pretty easy to recognize …

Estimated Rows ≠ Actual Rows

enkitec

Cardinality – Misunderestimate



```
PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------------
SQL_ID  0qa98gcnnza7h, child number 1
-------------------------------------
select avg(pk_col) from kso.skew where col1 > 0

Plan hash value: 568322376

----------------------------------------------------------------------------------------------------
| Id  | Operation                 | Name | Starts | E-Rows | A-Rows |   A-Time   | Buffers |
----------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT          |      |      1 |        |      1 |00:00:06.43 |    162K|
|   1 |  SORT AGGREGATE           |      |      1 |      1 |      1 |00:00:06.43 |    162K|
|*  2 |   TABLE ACCESS STORAGE FULL| SKEW |      1 |   1234 |    32M|00:00:03.43 |    162K|
----------------------------------------------------------------------------------------------------
```

Adaptive Execution Plans

How Does it Work?

- Optimizer Can Change It’s Mind in Mid-Execution
  - But Don’t Panic!
  - Easy to See
  - Only kicks in when it recognizes a mistake

enkitec

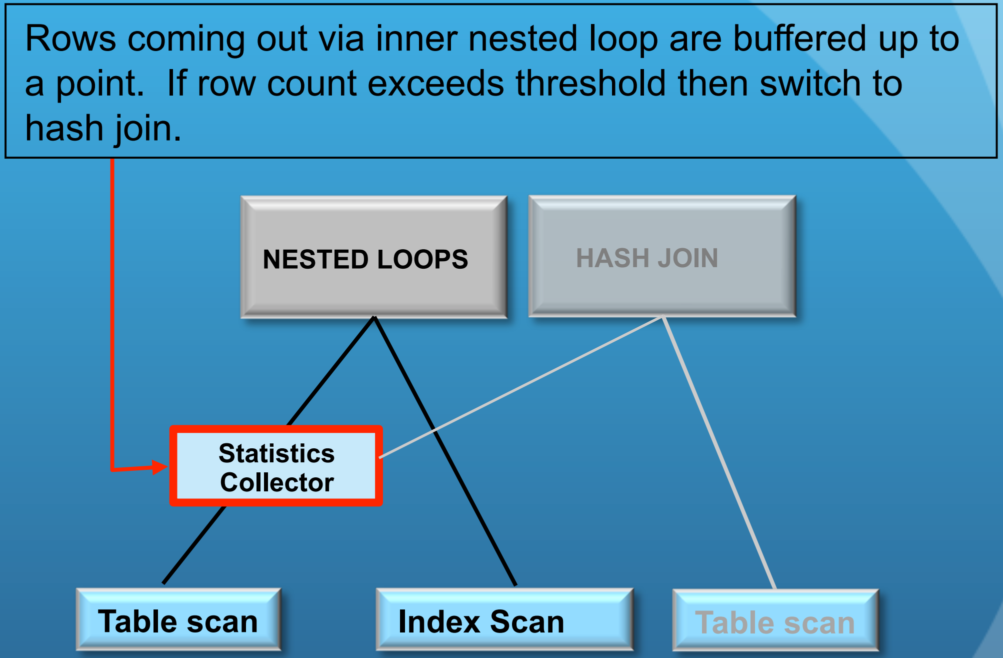
Adaptive Execution Plans

Alternative sub-plans are pre-computed

Sub-plans stored in the cursor

Stats collector inserted before join

Rows buffered until final decision is made


Rows coming out via inner nested loop are buffered up to a point. If row count exceeds threshold then switch to hash join.
NESTED LOOPS
HASH JOIN
Statistics Collector
Table scan
Index Scan
Table scan

Adaptive Execution Plans

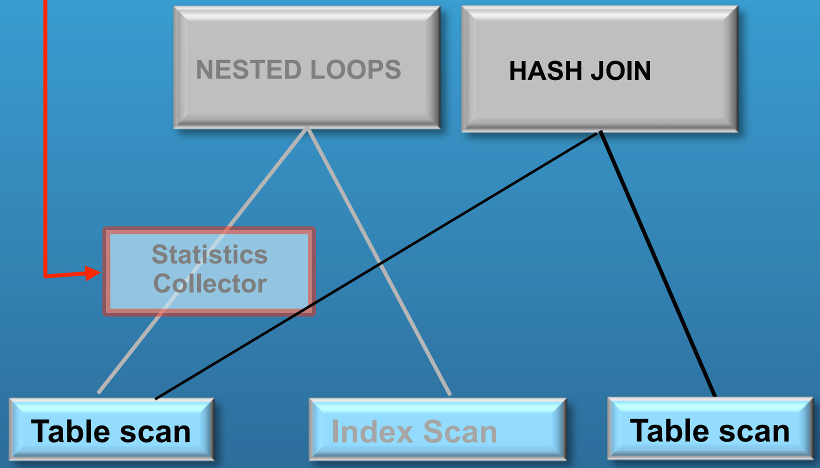
Number of rows seen in statistics collector exceeds threshold

Plan switches to hash join

Statistics collector disabled

Plan resolved on first execution & remains the same for subsequent executions

Statistics collector disabled after decision is made and becomes a pass through operation.



Final Plan is a hash join

Interaction with Adaptive Plans

Adaptive Plans Can Be Captured

- if no baseline exists (The Final Plan)
- if baseline exists, add default plan (mark as adaptive)

Once Accepted – No Longer Marked Adaptive

Wrap Up

Change of Heart
Capture Is Viable Now – not enabled by default in 12c
Evolve is Required Though – enabled by default in 12c
12c Stores Plans – so easier to diagnose reproducibility issues
There are big companies using it in a big way
There are companies that are misusing it

Questions?

Contact Information : Kerry Osborne
kerry.osborne@enkitec.com
kerryosborne.oracle-guy.com
www.enkitec.com