**How To Stabilize the Execution plan using SQL PLAN Baseline**

Each SQL associated with SQL\_ID and based on the DBMS\_STATS, the Optimizer chooses a execution plan identified with PLAN\_HASH\_VALUE.   Hence, SQL\_ID with multiple PLAN\_HASH\_VALUE often causes inconsistent performance.

**How to fix this?**

To start with, need to identify the optimal PLAN\_HASH\_VALUE for the SQL\_ID.

Below Query give the history of the SQL\_ID with PLAN\_HASH\_VALUE and execution time lines to choose the optimal PLAN\_HASH\_VALUE.  Make a note of the instance number in case of RAC for the optimal PLAN\_HASH\_VALUE.

***set lines 155***

***col execs for 999,999,999***

***col avg\_etime for 999,999.999***

***col avg\_lio for 999,999,999.9***

***col begin\_interval\_time for a30***

***col node for 99999***

***break on plan\_hash\_value on startup\_time skip 1***

***select ss.snap\_id, ss.instance\_number node, begin\_interval\_time, sql\_id, plan\_hash\_value,***

***nvl(executions\_delta,0) execs,***

***(elapsed\_time\_delta/decode(nvl(executions\_delta,0),0,1,executions\_delta))/1000000 avg\_etime,***

***(buffer\_gets\_delta/decode(nvl(buffer\_gets\_delta,0),0,1,executions\_delta)) avg\_lio***

***from DBA\_HIST\_SQLSTAT S, DBA\_HIST\_SNAPSHOT SS***

***where sql\_id = nvl('&sql\_id','XXXXXXXXX')***

***and ss.snap\_id = S.snap\_id***

***and ss.instance\_number = S.instance\_number***

***and executions\_delta > 0***

***order by 1, 2, 3***

***/***

Before creating PLAN\_BASELINE, check if the SQL\_ID with optimal PLAN\_HASH\_VALUE is in cursor cache.

Need to query v$SQL :***select inst\_id ,SQL\_ID,PLAN\_HASH\_VALUE from gv$SQL;***

If the same optimal PLAN\_HASH\_VALUE identified from the above query, you can create Plan Baseline from the Cursor Cache using below Steps:-

**Loading SQL Plan Baseline from CURSOR Cache:**

***DECLARE***

***my\_plans pls\_integer;***

***BEGIN***

***my\_plans := DBMS\_SPM.LOAD\_PLANS\_FROM\_CURSOR\_CACHE(sql\_id => ‘&SQL\_ID’,PLAN\_HASH\_VALUE=>’&PLAN\_HASH\_VALUE’,fixed=>’YES’);***

***END;***

***/***

***commit;***

Need to query ***dba\_sql\_plan\_baselines***, to check if the plan baseline created.

If the Optimal PLAN\_HASH\_VALUE does not exist in cursor cache, we need load it from AWR.

**Loading SQL Plan Baseline from AWR**

**Step1:-** Need to create SQL tuning set name, I am choosing the name as TUNING\_SET1 in this document.

***exec dbms\_sqltune.create\_sqlset(sqlset\_name => 'TUNING\_SET1',description => 'sqlset descriptions');***

**Step2:-** Load the SQL from AWR to above Tuning SET name TUNING\_SET1

***declare***

***baseline\_ref\_cur DBMS\_SQLTUNE.SQLSET\_CURSOR;***

***begin***

***open baseline\_ref\_cur for***

***select VALUE(p) from table(***

***DBMS\_SQLTUNE.SELECT\_WORKLOAD\_REPOSITORY(&begin\_snap\_id, &end\_snap\_id,'sql\_id='||CHR(39)||'&sql\_id'||CHR(39)||'',NULL,NULL,NULL,NULL,NULL,NULL,'ALL')) p;***

***DBMS\_SQLTUNE.LOAD\_SQLSET(' TUNING\_SET1', baseline\_ref\_cur);***

***end;***

***/***

***COMMIT;***

**Note:-** Input the SQL\_ID and snapshotIDs identified from the first step .Run from the appropriate instance , where optimal PLAN\_HASH\_VALUE identified.

**Ensure the SQL\_SET:-**

***SELECT NAME,OWNER,CREATED,STATEMENT\_COUNT FROM DBA\_SQLSET where name='TUNING\_SET1';***

**Note:-** **statement\_count** value in the above query should be 1

**To ensure right PLAN\_HASH\_VALUE is there in Tuning set:-**

***set long 2000000***

***select \* from table(dbms\_xplan.display\_sqlset('TUNING\_SET1','&sql\_id'));***

**Step3 :-** Load SQL Plan baseline from above tuning set.

***declare***

***my\_int pls\_integer;***

***begin***

***my\_int := dbms\_spm.load\_plans\_from\_sqlset (***

***sqlset\_name => 'TUNING\_SET1',***

***sqlset\_owner => 'SYS',***

***fixed => 'YES',***

***enabled => 'YES');***

***DBMS\_OUTPUT.PUT\_line(my\_int);***

***end;***

***/***

***commit;***

Query dba\_sql\_plan\_baselines  , to check if the plan baseline created.