# ORACLE DB 升级性能保障利器 SPA最佳实践

丁俊 2016-1

# 个人简介

- 姓名：丁俊 网名：dingjun123

- DBA+社群联合发起人、ITPUB开发版版主、ITPUB社区专家、ChinaUnix BLOG专家，ITPUB 2010-2013连续4届最佳精华获得者、2011-2014连续4届最佳版主

- 电子工业出版社终身荣誉作者，《剑破冰山-Oracle开发艺术》副主编

ORACLE ACE Associate

ACOUG All China Oracle User Group 中国 Oracle 用户组

ZJOUG MEMBER 中国浙江应用中间件与数据库用户组

E-mail:dingjunlove@163.com

Blog:http://blog.chinaunix.net/uid/7655508.html

## 1 升级性能保障利器SPA

- SPA简介
- SPA使用场合
- SPA实施流程

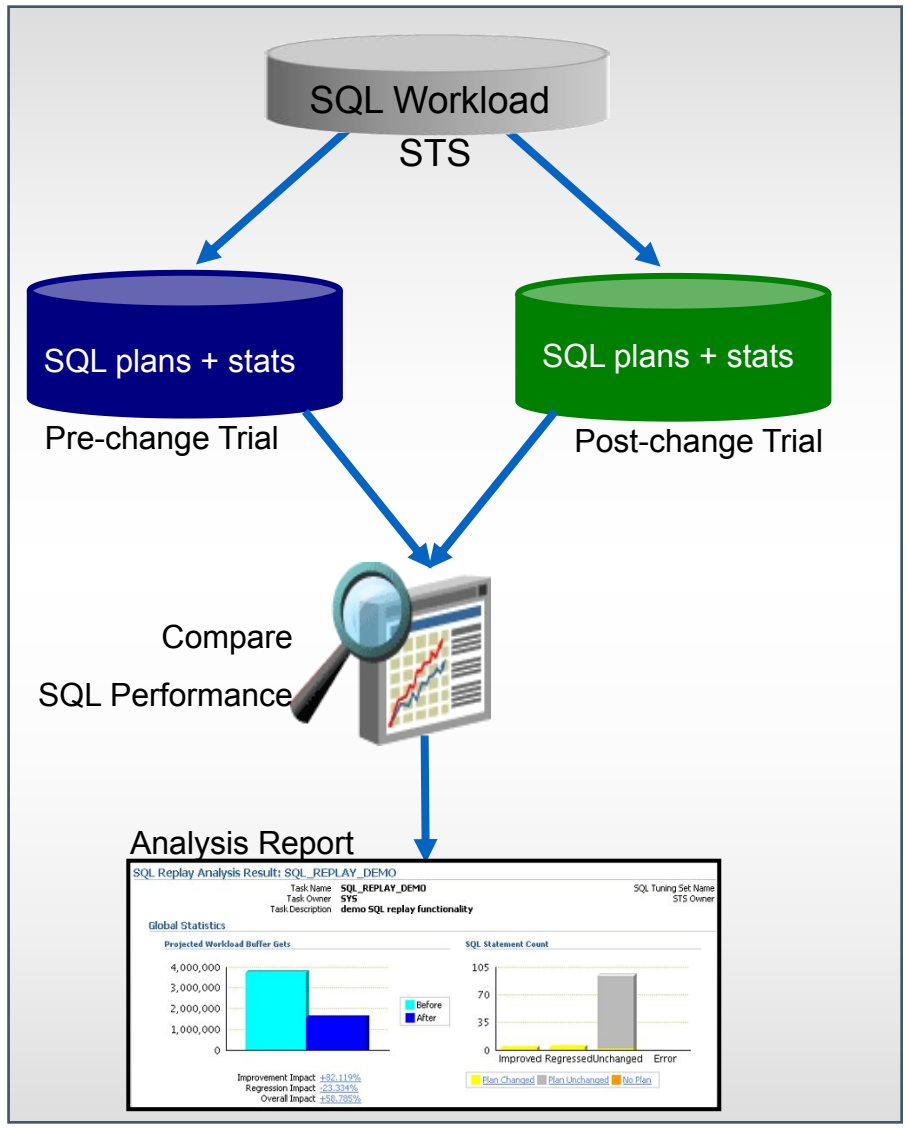
## 2 捕获SQL负载和生成报告

- SPA项目规划
- 高效捕获SQL负载
- 生成报告从10天到1天

## 3 SPA报告分析和优化实施

- SPA报告概览
- 优化性能下降SQL要点
- 优化案例
- 优化实施

# 1.1 升级性能保障利器SPA-简介




- Helps users predict the impact of system changes on SQL workload response time

Low overhead capture of SQL workload to SQL Tuning Set (STS) on production system

Build different SQL trials (experiments) of SQL statements performance by test execution

Analyzes performance differences

- Offers fine-grained performance analysis on individual SQL

Integrated with STS, SQL Plan Baselines, & SQL Tuning Advisor to form an end-to-end solution

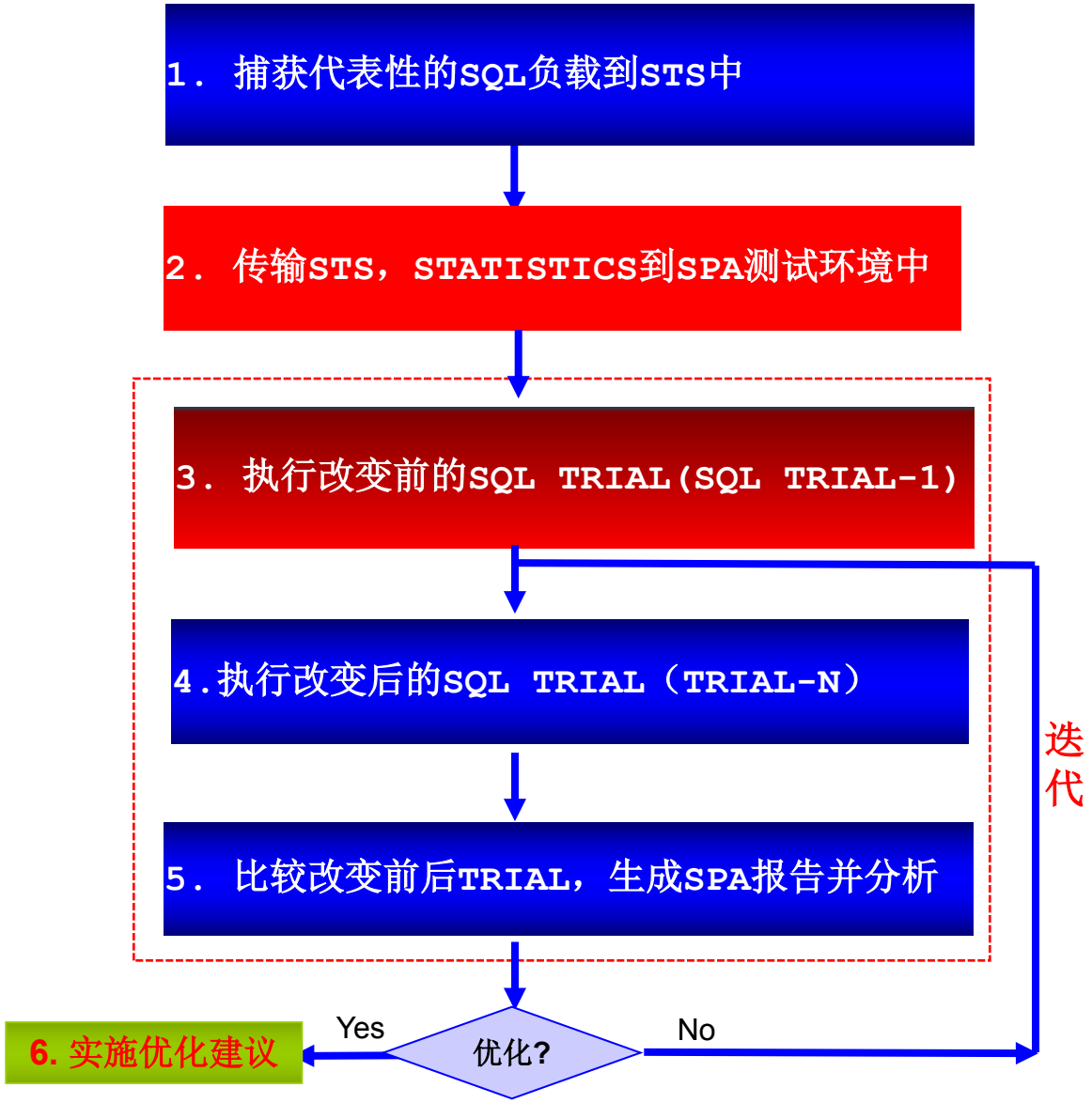
## 1.2 升级性能保障利器SPA-使用场合

- 数据库升级或打补丁
- 改变数据库参数
- 改变Schema
- 收集统计信息
- 实施优化建议
- OS/硬件改变

SPA可以使用在:

- 任何影响SQL执行计划和性能的变更操作中
- 常用于ORACLE大版本升级中，如10g到11g
- 生产和测试环境中

# 1.3 升级性能保障利器-SPA实施流程




- 生产系统捕获SQL负载
- STS和STATISTICS是SPA分析的重要数据基础
- 执行SPA分析任务—改变前(运行快,<1min)
- **执行SPA分析任务—改变后(实际运行,慢)**
- 生成SPA报告，分析，迭代，实施分析结果

# 2.1 捕获SQL负载和生成报告-SPA项目规划

## XX库升级SPA项目规划(10.2.0.4→11.2.0.4,RAC 2节点，A,B库)

### SQL SETS采集

**覆盖所有关键负载：**

- 确定Parsing schema范围
- 日常SQL
- 日帐、出账SQL
- AWR中SQL

**目标：**

- 尽可能多地覆盖业务SQL
- 所有节点和库

**采集周期：**

- 日常SQL负载采集1个月，覆盖生产出账，日帐
- 模拟出账单独采集
- AWR采集（45天）
- 升级前30天停止采集

### 生成SPA报告

**执行升级前后SQL性能对比：**

- 提前安排SPA分析环境
- 传输STS和统计信息
- 执行SPA分析任务并生成报告

**目标：**

- 最迟不超过7天生成全部报告(buffers,cpu time,elapsed time,error,unsupport)
- 尽可能将生成报告时间缩短到2天内

**迭代SPA分析任务：**

- 分析报告过程中，实施影响全局的优化操作，需要迭代SPA分析任务，重新生成报告。

### 分析报告

**报告分析：**

- 重点关注buffer gets对比，之后是cpu time,elapsed time,error,unsupport
- 重点关注执行计划改变并且性能下降的SQL
- 分析性能下降原因

**目标：**

- 10到15天完成报告分析
- 升级前10天完成分析
- 所有实施通过迭代SPA分析确认无误

**编写报告：**

- 制定SPA报告分析结果表格、编写脚本用于优化实施

### 实施优化

**实施优化：**

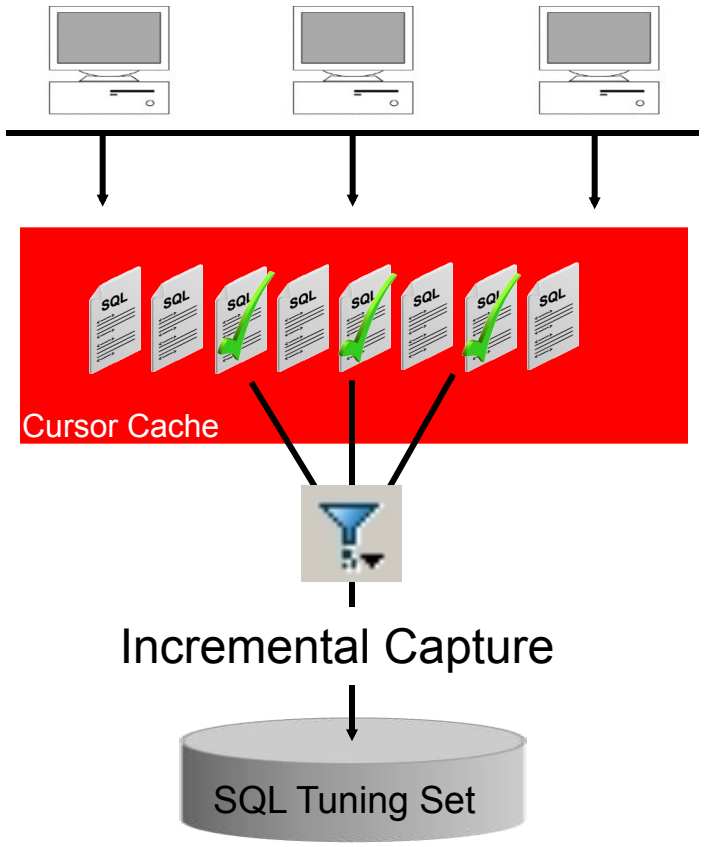
- 根据报告分析结果进行实施：

  如参数修改、收集统计信息，SQL PROFILE等。

**目标：**

- 实施完毕后检查关键SQL性能，确保无问题。
- 编写升级保障方案。
- 升级第二天高峰期收集dictionary和fixed objects统计信息。

# 2.3 捕获SQL负载和生成报告-高效捕获SQL负载(1)




- SQL Tuning Set (STS)用来存储和管理SQL负载。
- STS 包含完整的SQL执行上下文：
  - SQL Text, bind variables, parsing schema
  - Execution plans and run-time statistics
  - Number of executions
- STS 可以来源于Cursor Cache,AWR,sql trace,其它STS
- 需要从Cursor Cache中增量捕获一定时间（30天）
- 通过FILTER和RANK过滤不需要的SQL
- 使用DBMS_SQLTUNE包中的SELECT_CURSOR_CACHE和SELECT_WORKLOAD_REPOSITORY函数采集

- **STS数据字典：**

DBA_SQLSET*,如DBA_SQLSET,DBA_SQLSET_STATEMENTS

# 2.3 捕获SQL负载和生成报告-高效捕获SQL负载(2)

## Cursor Cache中采集中需要考虑的问题

- V$SQL中SQL语句数量大，超过20w条，系统较忙，直接采集慢，每次采集需要4小时以上，可能会漏掉一些SQL。
- 字面量SQL较多，类似select …from t where id=1,select …from t where id=2,直接采集存在冗余问题，会迅速增加STS中SQL的数量，影响后续处理效率，增加表空间使用率。
- 需要提供不间断采集及控制STS采集程序开启和停止的方法，便于管理。

## 解决方案

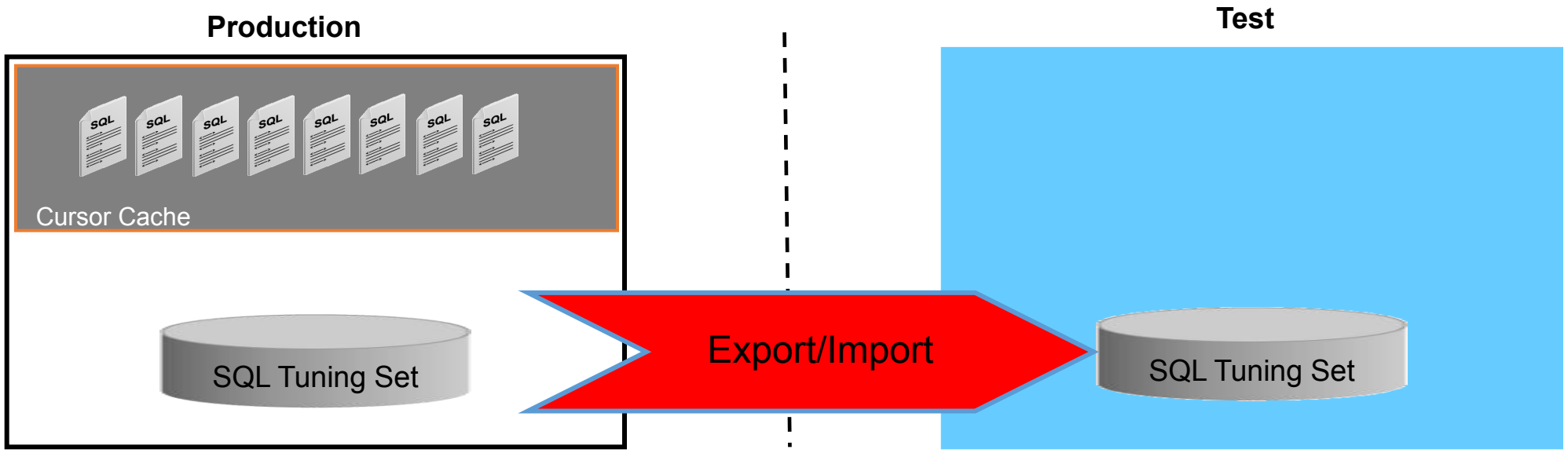
- 增加函数select_cursor_cache的参数basic_filter足够多的条件，如FORCE_MATCHING_SIGNATURE IS NOT NULL and upper(sql_text) not like ‘’%INSERT%INTO%VALUES%‘‘ AND ROWNUM<10000（可以先初始化全量采集，提高效率可以使用buffer_gets,disk_reads等范围条件，之后使用ROWNUM）。
- 根据FORCE_MATCHING_SIGNATURE字段过滤硬解析SQL
- 建立STS采集程序启停控制表，并编写脚本不间断采集。

实施结果：每次采集效率从4小时降低到5分钟，采集20w条1小时30分钟左右，对系统影响更小，漏掉需要采集SQL的几率减少，可过滤大部分硬解析SQL，更容易控制STS采集。

# 2.3 捕获SQL负载和生成报告-高效捕获SQL负载(3)

```
DECLARE
  v_status       NUMBER;--启停SPA标志
  v_cnt          NUMBER; --每个STS采集超过20w条自动放入到下一个STS中
  v_sqlset_name VARCHAR2(100) := 'sqlseta1_tab22'; --初始STS
BEGIN
  LOOP
    SELECT nvl(MAX(status), 0)
    INTO   v_status
    FROM   spa.sqlseta1_control;           --SPA启动控制表
    IF v_status = 0
    THEN
      EXIT;
    END IF;
    EXECUTE IMMEDIATE 'truncate  table spa.spaqc_a1';   --过滤字面量SQL表
    INSERT /*+append*/ INTO spa.spaqc_a1 SELECT DISTINCT FORCE_MATCHING_SIGNATURE FROM   DBA_SQLSET_STATEMENTS;
    SELECT statement_count INTO   v_cntFROM   dba_sqlset WHERE  NAME = 'sqlseta1_tab22';
    IF v_cnt > 200000
    THEN
      v_sqlset_name := 'sqlseta1_tab23';
      。。。
      END IF;
    END IF;
    DECLARE
      mycur DBMS_SQLTUNE.SQLSET_CURSOR;
    BEGIN
      OPEN mycur FOR
        SELECT VALUE(P)
        FROM   TABLE(dbms_sqltune.select_cursor_cache('parsing_schema_name in ('XXX'') and  rownum<10000
      and FORCE_MATCHING_SIGNATURE not in (select FORCE_MATCHING_SIGNATURE from spa.spaqc_a1)
      and FORCE_MATCHING_SIGNATURE IS NOT NULL
      and upper(sql_text) not like ''%INSERT%INTO%VALUES%''',NULL,NULL,NULL,NULL,1,NULL,'ALL')) p;
      dbms_sqltune.load_sqlset(sqlset_name=> v_sqlset_name,sqlset_owner=> 'SPA',populate_cursor => mycur,load_option=>
'MERGE');
      CLOSE mycur;
      dbms_output.put_line('step 1:' || to_char(SYSDATE, 'yyyymmdd hh24:mi:ss'));
    END;
    sys.dbms_lock.sleep(300);
  END LOOP;
END;
```

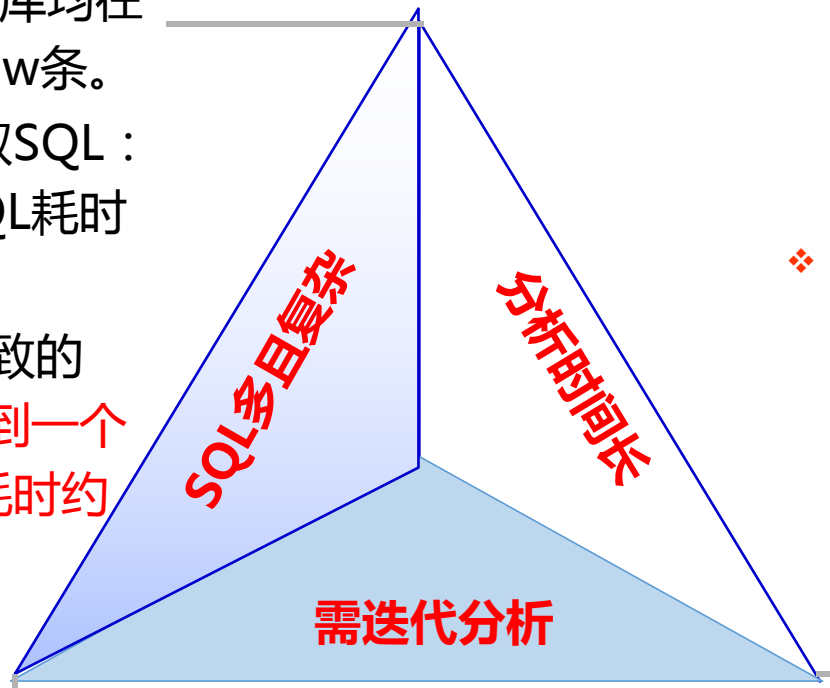
# 2.4 生成SPA报告从10天到1天-导入STS到测试环境中




- SPA测试环境应该与11g准生产环境相似（版本、参数等），测试环境的统计信息应该与升级前10.2.0.4保持一致（统计信息Export/Import,**导入统计信息之前注意先清空原有统计信息,导入后可以对STS进一步做剔除操作**)
- Pack SQL tuning set 到staging table中
- 传输staging table 到测试系统(datapump, db link, etc.)
- 在测试环境中unpack staging table到SQL tuning set中

# 2.4 生成SPA报告从10天到1天-451w条SQL分析的效率问题(1)

- STS中存储SQL多：A,B库均在200w条以上，共约451w条。
- A,B均含有上千条BI抽取SQL：性能低下，一般单条SQL耗时3-10分钟不等。
- 还有因执行计划变差导致的SQL性能下降，如果放到一个STS中执行SPA分析，耗时约10天。




- 大规模SPA分析任务，容易出问题，如ORA-01555，会导致SPA分析任务非正常完成，跑出的性能对比报告将不全，重跑返工成本高。
- 大规模SPA分析任务，如果SQL复杂，优化性能下降的SQL，进行影响全局的修改（如修改CBO参数），则需要进行迭代SPA分析（多次SPA分析），需要占用额外的工作量。

**解决方案：分割STS，执行并行SPA分析任务，提高效率**

# 2.4 生成SPA报告从10天到1天-效率提升关键之并行SPA分析(1)

- 从staging table中按SQL语句类型分割

```
1.将SQL均分，UPDATE+DELETE 1份，PL/SQL EXECUTE+CALL METHOD 份，INSERT单独1份，SELECT均分为10份，共13个SQLSET
select b.command_type,b.command_name,count(*)
from STS_TAB_B a,v$sqlcommand b
where a.command_type=b.command_type
group by b.command_type,b.command_name
;
COMMAND_TYPE COMMAND_NAME                                                    COUNT(*)
------------ ---------------------------------------------------------------- ----------
           3 SELECT                                                          2074179
          47 PL/SQL EXECUTE                                                    27255
           2 INSERT                                                          265051
           7 DELETE                                                            2886
           6 UPDATE                                                          201039
         170 CALL METHOD                                                          8

DECLARE
  v_tail   NUMBER   :=0;
BEGIN
  --SELECT 0..9尾号 改为SQLSET_RUN，每个STS大约存储20w条SQL
  FOR X IN (SELECT SQL_ID FROM STS_TAB_B where command_type =3 ORDER BY ELAPSED_TIME/EXECUTIONS) LOOP
    UPDATE STS_TAB_B SET NAME='SQLSET_RUN_'|| v_tail WHERE SQL_ID = X.SQL_ID;
    v_tail := MOD(v_tail + 1, 10);
  END LOOP;
END;
/

--UPDATE+DELETE SQLSET_RUN_10 PL/SQL,INSERT等类似UPDATE
 UPDATE STS_TAB_B SET NAME='SQLSET_RUN_10' where command_type in (6,7);
```

# 2.4 生成SPA报告从10天到1天-效率提升关键之并行SPA分析(2)

- unpack staging table

```
i=0
while [ "$i" -le 12 ]
do
cat > ./exec_SPA_RUN_$i.sh <<EOFSCRIPT
sqlplus spa/spa <<EOF
DECLARE
X NUMBER :=$i;
BEGIN
    DBMS_SQLTUNE.UNPACK_STGTAB_SQLSET (
                SQLSET_NAME          => 'SQLSET_RUN_'||X,
                SQLSET_OWNER         => 'SPA',
                REPLACE              => TRUE,
                STAGING_TABLE_NAME   => 'STS_TAB_B',
                STAGING_SCHEMA_OWNER => 'SPA');
END;
/
exit
EOF
EOFSCRIPT
chmod u+x exec_SPA_RUN_$i.sh
nohup ./exec_SPA_RUN_$i.sh > exec_SPA_RUN_$i.log 2>&1 &
i=$((i+1))
done
```

同时执行过多，可能引起**ORA-01555**错误

# 2.4 生成SPA报告从10天到1天-效率提升关键之并行SPA分析(3)

- 创建SPA分析任务 **跑SPA分析Trial需要导入老库统计信息到新库（统计信息如何处理要考虑清楚）**

瞬间即可完成

```
DECLARE
  L_SPA_TASK_NAME  VARCHAR2(64);
BEGIN
  FOR X IN 0..12 LOOP
    L_SPA_TASK_NAME := DBMS_SQLPA.CREATE_ANALYSIS_TASK(
                           TASK_NAME    => 'SPA_TASK_RUN_'||X,
                           DESCRIPTION  => 'SPA Analysis task at : '||TO_CHAR(SYSDATE, 'YYYY-MM-DD
HH24:MI:SS'),
                           SQLSET_NAME  => 'SQLSET_RUN_'||X,
                           SQLSET_OWNER => 'SPA');
    DBMS_OUTPUT.PUT_LINE('SPA Task Created as : '||L_SPA_TASK_NAME);
  END LOOP;
END;
/
```

- 并行任务生成10g Trial

```
i=0
while [ "$i" -le 12 ]
do
cat > ./exec_SPA_RUN_$i.sh <<EOFSCRIPT
sqlplus spa/xxx <<EOF
EXEC DBMS_SQLPA.EXECUTE_ANALYSIS_TASK( TASK_NAME => 'SPA_TASK_RUN_'||$i,EXECUTION_NAME => 'EXEC_10G_RUN_'||$i,EXECUTION_TYPE => 'CONVERT
SQLSET',EXECUTION_DESC => 'Convert 10g SQLSET for SPA Task at : '||TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS'));
exit
EOF
EOFSCRIPT
chmod u+x exec_SPA_RUN_$i.sh
nohup ./exec_SPA_RUN_$i.sh > exec_SPA_RUN_$i.log 2>&1 &
i=$((i+1))
done
```

# 2.4 生成SPA报告从10天到1天-效率提升关键之并行SPA分析(4)

- 并行任务生成11g Trial

```
i=0
while [ "$i" -le 12 ]
do
cat > ./exec_SPA_RUN_$i.sh <<EOFSCRIPT
sqlplus spa/xxx <<EOF
EXEC DBMS_SQLPA.EXECUTE_ANALYSIS_TASK('SPA_TASK_RUN_$i', 'TEST EXECUTE', 'EXEC_11G_RUN_$i', NULL, 'Execute SQL
in 11g for SPA Task at : '||TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS'));
exit
EOF
EOFSCRIPT
chmod u+x exec_SPA_RUN_$i.sh
nohup ./exec_SPA_RUN_$i.sh > exec_SPA_RUN_$i.log 2>&1 &
i=$((i+1))
done
```

最慢最关键的操作，耗时较长，需监控进度

**问题：**

- **11g trial是最慢的操作，13个进程耗时1天，容易出现ORA-01555**
- **个别语句性能下降多，会拖慢整个分析过程**
- **需要经常监视分析任务执行情况，对出现的问题及时解决**

# 2.4 生成SPA报告从10天到1天-11g Trial出现的问题及解决方法

- 出现ORA-01555错误
  - 增加undo_retention
  - 扩展undo tablespace

还有问题

- 在做11g Trial前，设置超时时间

```
exec DBMS_SQLPA.SET_ANALYSIS_TASK_PARAMETER
(task_name=>'SPA_TASK_RUN_1',
parameter=>'LOCAL_TIME_LIMIT',
value=>'5000');
```

- 运行时间长的SQL拖慢整个分析进度
  - 中断SPA分析任务

```
EXEC DBMS_SQLPA.INTERRUPT_ANALYSIS_TASK(task_name=>'SPA_TASK_RUN_1');
```

  - 恢复中断的SPA分析任务

```
EXEC DBMS_SQLPA.RESUME_ANALYSIS_TASK(task_name=>'SPA_TASK_RUN_1');
```

  - 中断SQL检查

见超时SQL检查，条件改为a.message like '%interrupted%';

----------超时SQL检查----------

```
select b.sql_id
from DBA_ADVISOR_FINDINGS
a,dba_advisor_sqlstats b
where a.task_id=b.task_id and
a.object_id=b.object_id
and a.TYPE='ERROR'
and a.task_name like 'SPA_TASK_RUN_1'
and a.message like '%timed out%';
```

- 定期监视SPA分析任务进度

```
SELECT SID,
       TASK_ID,
       (select distinct task_name
          from DBA_ADVISOR_EXECUTIONS b
         where a.task_id = b.task_id) task_name,
       SOFAR,
       TOTALWORK,
       ROUND(SOFAR / TOTALWORK, 2) * 100 || '%' fin_ratio,
       ELAPSED_SECONDS,
       LAST_UPDATE_TIME,
       START_TIME,
       START_TIME + (SYSDATE - START_TIME) / SOFAR * TOTALWORK EST_END_TIME
  FROM V$ADVISOR_PROGRESS a
 WHERE TASK_ID IN (SELECT TASK_ID
                     FROM DBA_ADVISOR_EXECUTIONS
                    where task_name like 'SPA_TASK%')
   AND SOFAR <> 0
 ORDER BY 2;
```

Trial完毕后，执行比较分析，生成报告（略）

SPA的一些视图：

DBA{USER}_ADVISOR_*

# 3.1 SPA报告分析和优化实施-SPA报告概览

**Task Information:**

Task Name : SPA_TASK_RUNY_1
Task Owner : SPA
Description : SPA Analysis task at : 2014-10-31 16:07:27

**Workload Information:**

SQL Tuning Set Name : SQLSET_RUN_1
SQL Tuning Set Owner : SPA
Total SQL Statement Count : 207418

**Execution Information:**

Execution Name : Compare_BUFFER_GETS_time
Execution Type : COMPARE PERFORMANCE
Description :
Scope : COMPREHENSIVE
Status : COMPLETED
Number of Timeouts : 1
Started : 11/01/2014 11:51:05
Last Updated : 11/01/2014 11:58:43
Global Time Limit : UNLIMITED
Per-SQL Time Limit : UNUSED
Number of Errors : 156

**Analysis Information:**

**Before Change Execution:**

Execution Name : EXEC_10G_RUNY_1
Execution Type : CONVERT SQLSET
Scope : COMPREHENSIVE
Status : COMPLETED
Started : 10/31/2014 16:08:44
Last Updated : 10/31/2014 16:08:44
Global Time Limit : UNLIMITED
Per-SQL Time Limit : UNUSED

**After Change Execution:**

Execution Name : EXEC_11G_RUNY_1
Execution Type : TEST EXECUTE
Scope : COMPREHENSIVE
Status : COMPLETED
Started : 10/31/2014 16:14:56
Last Updated : 10/31/2014 22:55:23
Global Time Limit : UNLIMITED
Per-SQL Time Limit : UNUSED
Number of Errors : 156

A,B库各有上千条SQL待分析

## Report Summary

**Projected Workload Change Impact:**

Overall Impact : -32.98%
Improvement Impact : 21.82%
Regression Impact : -54.8%

**SQL Statement Count**

| SQL Category | SQL Count | Plan Change Count |
|---|---|---|
| Overall | 207418 | 128672 |
| Improved | 4 | 0 |
| Regressed | 2 | 1 |
| Unchanged | 207255 | 128671 |
| with Timeout | 1 | 0 |
| with Errors | 156 | 0 |

**Top 300 SQL Sorted by Absolute Value of Change Impact on the Workload**

| object_id | sql_id | Impact on Workload | Execution Frequency | Metric Before | Metric After | Impact on SQL | Plan Change |
|---|---|---|---|---|---|---|---|
| 791392 | d1c3zfa57yvn1 | -52.87% | 42754 | 84862.1591196145 | 162924 | -91.99% | n |
| 745127 | 9gds54g5kzyjv | 10.74% | 199468 | 3401.31202498646 | 3 | 99.91% | n |
| 816902 | g08fa1xx7phqx | 7.93% | 355857 | 1417.43811980655 | 11 | 99.22% | n |
| 759906 | ams0w3nk6cfh8 | -1.93% | 689 | 17758.4339622642 | 194446 | -994.95% | y |
| 646970 | 1w9wshvp79kva | 1.88% | 339029 | 947.460627261975 | 597 | 36.99% | n |
| 648719 | 20pftn35ryp49 | 1.27% | 349307 | 813.160935795733 | 584 | 28.18% | n |
| 755460 | a8wkm3qj47z74 | .73% | 1973 | 72794.4825139382 | 49522 | 31.97% | n |
| 633923 | 0wa1bwdtc0bqk | .48% | 7253 | 4172.55466703433 | 20 | 99.52% | n |

## •关注点：

- 优先分析性能下降（Impact on SQL <0)且执行计划改变的SQL(Plan Change=y)
- 其次可关注性能下降执行计划未变的(Plan Change=n)
- 优先关注buffer gets报告，其次是cpu time, elapsed time
- errors,unsupport特别关注errors中是否有不应该出错的语句

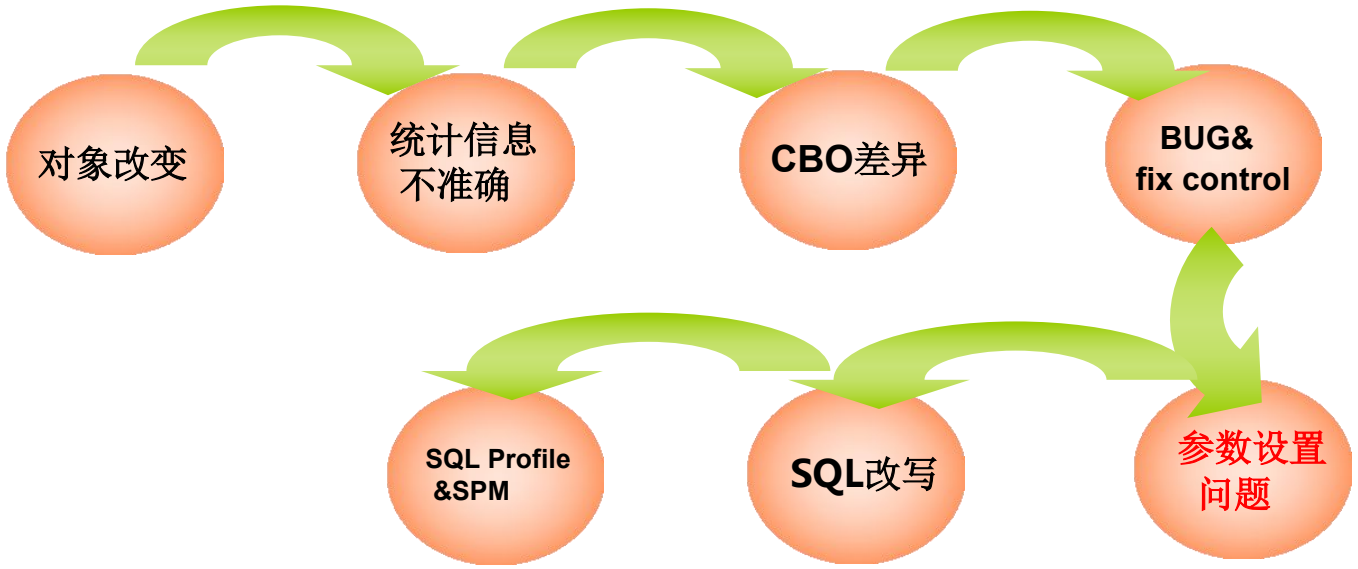
## •报告类型：

本次执行分析A,B库均在220w条，共约451w条。

Top N的buffer gets、cpu time、elapsed time

errors、unsupport 5种报告，本次升级每个库有13个日常STS，AWR 1个，模拟出账1个，每个库共75份报告，A,B库共150份。

# 3.2 SPA报告分析和优化实施-优化性能下降SQL ROAD MAP(1)




修改参数和fix control
需要迭代SPA分析

NVL(TO_CHAR(OP_TIME,'yyyymmdd'),' ') , NVL(LOGIN_NO,' ') , NVL(LOGIN_ACCEPT,0) , NVL (BAK_FIELD,' ') , NVL

**Bind Variables:**

1 - (CHAR): 13856372470

**Execution Statistics:**

| Stat Name | Impact on Workload | Value Before | Value After | Impact on SQL |
|---|---|---|---|---|
| elapsed_time | 2.46% | .000848 | .000218 | 74.3% |
| parse_time | | | .00076 | |
| cpu_time | 4.05% | .00049 | 0 | 100% |
| user_io_time | | | 0 | |
| buffer_gets | -2.85% | 21 | 38 | -73.81% |
| cost | -1.68% | 29 | 30 | -3.45% |
| reads | 1.04% | 0 | 0 | 100% |
| writes | 0% | 0 | 0 | 0% |
| io_interconnect_bytes | | | 0 | |
| rows | | 0 | 1 | |

**Note: time statistics are displayed in seconds**

**Notes:**

**After Change:**

1. The statement was first executed to warm the buffer cache.
2. Statistics shown were averaged over next 9 executions.

**Findings (3):**

1. The performance of this SQL has regressed.
2. This SQL statement returned zero rows.
3. The number of returned rows in execution 'EXEC_10G_RUNZ_0' is different than in execution 'EXEC_11G_RUNZ_0'.

**Execution Plan Before Change:**

**Plan Hash Value** : 3312903934

- 识别SPA报告中的误报情况：10g采集的指标是平均情况，只有参考意义。

# 3.2 SPA报告分析和优化实施-优化性能下降SQL ROAD MAP(2)

- **统计信息**

导入老库统计信息可能会丢失部分信息，另外由于收集算法和CBO算法差异导致无法在新库正确使用，需要更新有问题表的统计信息（opt_method使用repeat、收集直方图、扩展统计信息、delete stats后重新收集，调整采样比例等。

- **优化器算法差异、BUG&FIX CONTROL、优化器参数**

使用OPTIMIZER_FEATURES_ENABLE HINT进行优化器降级测试、10053、SQLT工具等分析。

- **其它**

比如版本改变，如WM_CONCAT函数、建立索引、改写SQL。

很少考虑建立索引和改写语句

- **终极方法**

SQL PROFILE、SPM。

# 3.2 SPA报告分析和优化实施-强大的SQLT工具

## 215187.1 SQLTXPLAIN (SQLT) 12.1.06 2014年1月30日

帮助诊断性能较差的 SQL 语句的工具

- SQLT 概览
- 安全模式
- 安装 SQLT
- 卸载 SQLT
- 升级 SQLT
- 常见问题
- 新增功能！

主要方法

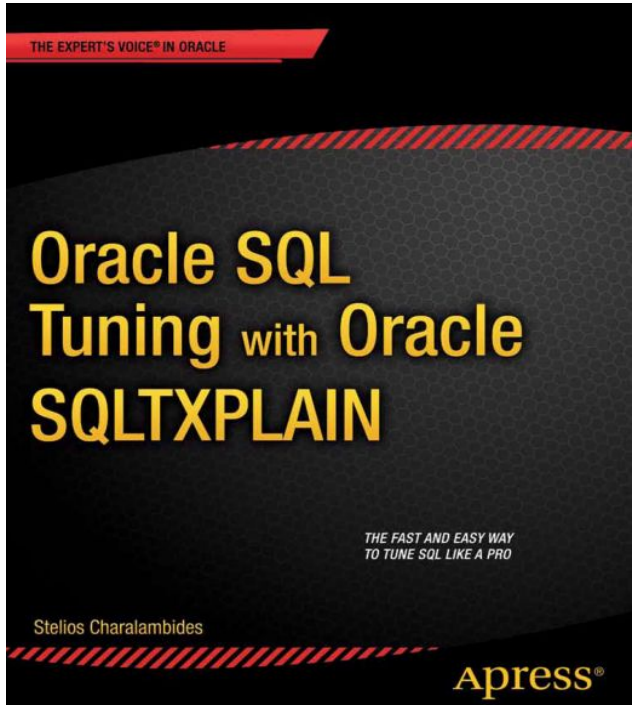
- XTRACT
- XECUTE
- XTRXEC
- XPLAIN
- XTRSBY
- XPREXT
- XPREXC

特殊方法

- COMPARE
- TRCANLZR
- TRCAXTR
- TRCASPLIT
- XTRSET

高级方法和模块

- PROFILE
- XGRAM
- XPLORE
- XHUME



# 3.3 SPA报告分析和优化实施-优化案例1之CBO参数设置问题(1)

- **参数设置与OJPPD查询转换问题，10.2.0.4可走OJPPD**

```
SELECT NVL(SUM(OTHER_AMOUNT), 0)
  FROM (SELECT B.WRTOFF_AMOUNT AS OTHER_AMOUNT
          FROM AC_ACCTBOOKCHG_201408 A,    --VIEW
          AC_WRTOFFREC_1408_3 B
          WHERE EXISTS (SELECT 1
                  FROM AC_CONTRACT_INFO C
                 WHERE A.CONTRACT_NO = C.CONTRACT_NO
                   AND C.ACCOUNT_TYPE = '1')
           AND A.PAYIO_CLS = 'X'
           AND A.PAYIO_SN = B.WRTOFF_SN
           AND A.CONTRACT_NO != :V1
           AND B.ID_NO = :V2
        UNION ALL
        SELECT A.PAY_AMOUNT AS OTHER_AMOUNT
          FROM AC_ACCTPAYREC_1408 A
         WHERE EXISTS (SELECT 1
                  FROM AC_CONTRACT_INFO C
                 WHERE A.CONTRACT_NO = C.CONTRACT_NO
                   AND C.ACCOUNT_TYPE = '1')
           AND A.REL_CONTRACT_NO = :V1)
```

在10.2.0.4中走OJPPD，可以通过OUTLINE查看得知：
OLD_PUSH_PRED(@"SEL$71CF4DF6" "A"@"SEL$2" 2)
升级到11.2.0.4,杯具发生了！

```
Execution Plan
----------------------------------------------------------
Plan hash value: 275940462

-------------------------------------------------------------------------------------------------------------------------
--
| Id  | Operation                           | Name                          | Rows  | Bytes | Cost (%CPU)| Time
|
-------------------------------------------------------------------------------------------------------------------------
--
|   0 | SELECT STATEMENT                    |                               |     1 |    13 |  3418   (1)| 00:00:42
|
|   1 |  SORT AGGREGATE                     |                               |     1 |    13 |            |
|
|   2 |   VIEW                              |                               |  1877 | 24401 |  3418   (1)| 00:00:42
|
|   3 |    UNION-ALL                        |                               |       |       |            |
|
|   4 |     NESTED LOOPS SEMI               |                               |     1 |    98 |    68   (0)| 00:00:01
|
|   5 |      NESTED LOOPS                   |                               |     1 |    86 |    66   (0)| 00:00:01
|
|   6 |       TABLE ACCESS BY INDEX ROWID   | AC_WRTOFFREC_1408_3           |    13 |   624 |     4   (0)| 00:00:01
|
|*  7 |        INDEX RANGE SCAN             | IDX_WRTOFFREC_IDNO_1408_3     |    13 |       |     2   (0)| 00:00:01
|
|   8 |       VIEW                          | AC_ACCTBOOKCHG_201408         |     1 |    38 |     5   (0)| 00:00:01
|
|   9 |        UNION-ALL PARTITION          |                               |       |       |            |
|
|* 10 |         TABLE ACCESS BY INDEX ROWID | AC_ACCTBOOKCHG_1408_0         |     1 |    38 |     4   (0)| 00:00:01
|
|* 11 |          INDEX RANGE SCAN           | IDX_PAYSNIO_ACCTBOOKCHG_1408_0 |     1 |       |     3   (0)| 00:00:01
|
                    ...
|* 28 |         TABLE ACCESS BY INDEX ROWID | AC_ACCTBOOKCHG_1408_9         |     1 |    38 |     4   (0)| 00:00:01
|
|* 29 |          INDEX RANGE SCAN           | IDX_PAYSNIO_ACCTBOOKCHG_1408_9 |     1 |       |     3   (0)| 00:00:01
|
|  30 |      PARTITION RANGE ITERATOR       |                               | 56870 |   667K|     2   (0)| 00:00:01
|
|* 31 |       TABLE ACCESS BY LOCAL INDEX ROWID| AC_CONTRACT_INFO           | 56930 |   667K|     2   (0)| 00:00:01
|
|* 32 |        INDEX UNIQUE SCAN            | PK_AC_CONTRACT_INFO_MBW       |     1 |       |     1   (0)| 00:00:01
|
|  33 |     NESTED LOOPS SEMI               |                               |  1876 | 95676 |  3350   (1)| 00:00:41
|
|  34 |      TABLE ACCESS BY INDEX ROWID    | AC_ACCTPAYREC_1408            |  1876 | 73164 |   347   (1)| 00:00:05
|
|* 35 |       INDEX RANGE SCAN              | IDX_ACCTPAYREC_1408_R         |   750 |       |     2   (0)| 00:00:01
|
|  36 |      PARTITION RANGE ITERATOR       |                               |   230K|  2700K|     2   (0)| 00:00:01
|
|* 37 |       TABLE ACCESS BY LOCAL INDEX ROWID| AC_CONTRACT_INFO           |   230K|  2700K|     2   (0)| 00:00:01
```

分表多，类似SQL多，执行频次高

# 3.3 SPA报告分析和优化实施-优化案例1之CBO参数设置问题(2)

## •11.2.0.4 无法进行OJPPD转换

Plan hash value: 1795327273

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 1 | 13 | 226K (2) | 00:45:14 |
| 1 | SORT AGGREGATE | | 1 | 13 | | |
| 2 | VIEW | | 2001 | 26013 | 226K (2) | 00:45:14 |
| 3 | UNION-ALL | | | | | |
| 4 | NESTED LOOPS SEMI | | 125 | 11250 | 222K (2) | 00:44:34 |
| * 5 | HASH JOIN | | 131 | 10218 | 222K (2) | 00:44:31 |
| 6 | TABLE ACCESS BY INDEX ROWID | AC_WRTOFFREC_1408_3 | 13 | 624 | 4 (0) | 00:00:01 |
| * 7 | INDEX RANGE SCAN | IDX_WRTOFFREC_IDNO_1408_3 | 13 | | 2 (0) | 00:00:01 |
| 8 | VIEW | AC_ACCTBOOKCHG_201408 | 657K | 18M | 222K (2) | 00:44:31 |
| 9 | UNION-ALL | | | | | |
| * 10 | TABLE ACCESS FULL | AC_ACCTBOOKCHG_1408_0 | 65968 | 2448K | 22327 (2) | 00:04:28 |
| * 11 | TABLE ACCESS FULL | AC_ACCTBOOKCHG_1408_1 | 65927 | 2446K | 22467 (2) | 00:04:30 |
| ... | | | | | | |
| * 19 | TABLE ACCESS FULL | AC_ACCTBOOKCHG_1408_9 | 65448 | 2428K | 22213 (2) | 00:04:27 |
| 20 | PARTITION RANGE ITERATOR | | 220K | 2580K | 2 (0) | 00:00:01 |
| * 21 | TABLE ACCESS BY LOCAL INDEX ROWID | AC_CONTRACT_INFO | 220K | 2580K | 2 (0) | 00:00:01 |
| * 22 | INDEX UNIQUE SCAN | PK_AC_CONTRACT_INFO_MHH | 1 | | 1 (0) | 00:00:01 |
| 23 | NESTED LOOPS SEMI | | 1876 | 95676 | 3350 (1) | 00:00:41 |
| 24 | TABLE ACCESS BY INDEX ROWID | AC_ACCTPAYREC_1408 | 1876 | 73164 | 347 (1) | 00:00:05 |
| * 25 | INDEX RANGE SCAN | IDX_AC_ACCTPAYREC_1408_R | 750 | | 2 (0) | 00:00:01 |
| 26 | PARTITION RANGE ITERATOR | | 230K | 2700K | 2 (0) | 00:00:01 |
| * 27 | TABLE ACCESS BY LOCAL INDEX ROWID | AC_CONTRACT_INFO | 230K | 2700K | 2 (0) | 00:00:01 |
| * 28 | INDEX UNIQUE SCAN | PK_AC_CONTRACT_INFO_MHH | 1 | | 1 (0) | 00:00:01 |

Predicate Information (identified by operation id):

```
   5 - access("A"."PAYIO_SN"="B"."WRTOFF_SN")
   7 - access("B"."ID_NO"=TO_NUMBER(:V2))
  10 - filter("PAYIO_CLS"='X' AND "CONTRACT_NO"<>TO_NUMBER(:V1))
  11 - filter("PAYIO_CLS"='X' AND "CONTRACT_NO"<>TO_NUMBER(:V1))
 ...
```

## •优化思路

1）按照优化要点逐步排查：索引是否失效，统计信息是否不准确（重收集是否能解决），新特性？BUG？FIX CONTROL设置？参数问题等。

2）收集统计信息无效。

3）在11.2.0.4中使用optimizer_features_enable分别测试10.2.0.4和11.2.0.3均可谓词推入到视图中走索引。

4）测试10.2之后和JPPD相关的fix control参数，也无效。（**from** v$system_fix_control **WHERE** sql_feature **LIKE** '%JPPD%'）

5）分析10053，详细参看是否是BUG导致，还是优化器改进问题，参数设置问题。

# 3.3 SPA报告分析和优化实施-优化案例1之CBO参数设置问题(3)

- 分析**10053**,关注被修改的默认参数，定位问题

```
Final query after transformations:******* UNPARSED QUERY IS *******

SELECT NVL(SUM("from$_subquery$_001"."OTHER_AMOUNT"), 0) "NVL(SUM(OTHER_AMOUNT),0)"
  FROM ((SELECT "B"."WRTOFF_AMOUNT" "OTHER_AMOUNT"
          FROM "DBCUSTADM"."AC_CONTRACT_INFO" "C",
               ((SELECT "AC_ACCTBOOKCHG_1408_0"."PAYIO_SN"    "PAYIO_SN",
                        "AC_ACCTBOOKCHG_1408_0"."CONTRACT_NO" "CONTRACT_NO",
                        "AC_ACCTBOOKCHG_1408_0"."PAYIO_CLS"   "PAYIO_CLS"
                   FROM DBACCADM."AC_ACCTBOOKCHG_1408_0" "AC_ACCTBOOKCHG_1408_0"
                  WHERE "AC_ACCTBOOKCHG_1408_0"."PAYIO_CLS" = 'X'
                    AND "AC_ACCTBOOKCHG_1408_0"."CONTRACT_NO" <> :B1) UNION ALL
                ...
                 (SELECT "AC_ACCTBOOKCHG_1408_9"."PAYIO_SN"    "PAYIO_SN",
                        "AC_ACCTBOOKCHG_1408_9"."CONTRACT_NO" "CONTRACT_NO",
                        "AC_ACCTBOOKCHG_1408_9"."PAYIO_CLS"   "PAYIO_CLS"
                   FROM DBACCADM."AC_ACCTBOOKCHG_1408_9" "AC_ACCTBOOKCHG_1408_9"
                  WHERE "AC_ACCTBOOKCHG_1408_9"."PAYIO_CLS" = 'X'
                    AND "AC_ACCTBOOKCHG_1408_9"."CONTRACT_NO" <> :B10)) "A",
               "DBACCADM"."AC_WRTOFFREC_1408_3" "B"
         WHERE "A"."PAYIO_SN" = "B"."WRTOFF_SN"
           AND "B"."ID_NO" = :B11
           AND "A"."CONTRACT_NO" = "C"."CONTRACT_NO"
           AND "C"."ACCOUNT_TYPE" = '1'
           AND "C"."CONTRACT_NO" <> :B12) UNION ALL
        (SELECT "A"."PAY_AMOUNT" "OTHER_AMOUNT"
           FROM "DBCUSTADM"."AC_CONTRACT_INFO"  "C",
                "DBACCADM"."AC_ACCTPAYREC_1408" "A"
          WHERE "A"."REL_CONTRACT_NO" = :B13
            AND "A"."CONTRACT_NO" = "C"."CONTRACT_NO"
            AND "C"."ACCOUNT_TYPE" = '1')) "from$_subquery$_001"
```

```
***************************************
PARAMETERS USED BY THE OPTIMIZER
********************************
  ************************************
  PARAMETERS WITH ALTERED VALUES
  ******************************
Compilation Environment Dump
。。。
_optimizer_cost_based_transformation = off
_optimizer_mjc_enabled               = false
_optimizer_squ_bottomup              = false
。。。
Bug Fix Control Environment
    fix  8560951 = enabled *
。。。

***************************
Query transformations (QT)
***************************
CBQT bypassed for query block SEL$1 (#0): Disabled by
parameter.
CBQT: Validity checks failed for 2a92tkckh82vs.
CSE: Considering common sub-expression elimination in query
block SEL$1 (#0)
。。。
***************************
Predicate Move-Around (PM)
***************************
。。。
OJPPD:     OJPPD bypassed: View semijoined to table.
JPPD:     JPPD bypassed: View not on right-side of outer-
join.
FPD:   Following are pushed to where clause of query block
SEL$12 (#0)
```

- 优化器算法改变，必须设置CBQT参数

在基于**COST**的查询转换部分，只能走**JPPD**（和**OJPPD**类似），**ORACLE**建议设置**CBQT**参数，基于**COST**查询转换更准确。
开启**COST**查询转换，初始化优化器参数 **_optimizer_cost_based_transformation设为默认值(linear)**。**CBQT**参数有如下值：**"exhaustive", "iterative", "linear", "on", "off"**。

另外通过测试得知，还需要设置**_optimizer_squ_bottomup** （**enables unnesting of subquery in a bottom-up manner**）**参数默认值true**。

# 3.3 SPA报告分析和优化实施-优化案例1之CBO参数设置问题(4)

- 改回默认参数，问题解决（影响大，进行迭代SPA分析）

**启用CBO查询转换后：**

OJPPD->JPPD

```
Final query after transformations:******* UNPARSED QUERY IS *******
SELECT NVL(SUM("from$_subquery$_001"."OTHER_AMOUNT"), 0) "NVL(SUM(OTHER_AMOUNT),0)"
  FROM ((SELECT "B"."WRTOFF_AMOUNT" "OTHER_AMOUNT"
          FROM "DBCUSTADM"."AC_CONTRACT_INFO" "C",
               ((SELECT "AC_ACCTBOOKCHG_1408_0"."PAYIO_SN"     "PAYIO_SN",
                        "AC_ACCTBOOKCHG_1408_0"."CONTRACT_NO" "CONTRACT_NO"
                   FROM DBACCADM."AC_ACCTBOOKCHG_1408_0" "AC_ACCTBOOKCHG_1408_0"
                  WHERE "AC_ACCTBOOKCHG_1408_0"."PAYIO_CLS" = 'X'
                    AND "AC_ACCTBOOKCHG_1408_0"."CONTRACT_NO" <> :B1
                    AND "AC_ACCTBOOKCHG_1408_0"."PAYIO_SN" = "B"."WRTOFF_SN")
             ...
                 UNION ALL
                 (SELECT "AC_ACCTBOOKCHG_1408_9"."PAYIO_SN"     "PAYIO_SN",
                         "AC_ACCTBOOKCHG_1408_9"."CONTRACT_NO" "CONTRACT_NO"
                    FROM DBACCADM."AC_ACCTBOOKCHG_1408_9" "AC_ACCTBOOKCHG_1408_9"
                   WHERE "AC_ACCTBOOKCHG_1408_9"."PAYIO_CLS" = 'X'
                     AND "AC_ACCTBOOKCHG_1408_9"."CONTRACT_NO" <> :B10
                     AND "AC_ACCTBOOKCHG_1408_9"."PAYIO_SN" = "B"."WRTOFF_SN")) "A",
               "DBACCADM"."AC_WRTOFFREC_1408_3" "B"
         WHERE "B"."ID_NO" = :B11
           AND "A"."CONTRACT_NO" = "C"."CONTRACT_NO"
           AND "C"."ACCOUNT_TYPE" = '1') UNION ALL
        (SELECT "A"."PAY_AMOUNT" "OTHER_AMOUNT"
           FROM "DBCUSTADM"."AC_CONTRACT_INFO"  "C",
                "DBACCADM"."AC_ACCTPAYREC_1408" "A"
          WHERE "A"."REL_CONTRACT_NO" = :B12
            AND "A"."CONTRACT_NO" = "C"."CONTRACT_NO"
            AND "C"."ACCOUNT_TYPE" = '1')) "from$_subquery$_001"
```

# 3.3 SPA报告分析和优化实施-优化案例2之fix control设置问题

```
select nvl(sum(to_char(a.WRTOFF_AMOUNT / 100,
       'fm9999999990.00')), 0) sum_wrtoff_amount,
       nvl(sum(to_char(a.FEEBEF_WRTOFF / 100,
       'fm9999999990.00')), 0) sum_feebef_wrtoff
  from ac_acctitematr      b,
       ur_user_info        c,
       ac_wrtoffrec_201511 a,    --view
       ac_acctbookitm      d
 where a.acct_item = b.acct_item
   and a.id_no = c.id_no
   and a.acctbook_itm = d.acctbook_itm
   and c.phone_no = :var0
   and to_date(beg_billcycl, 'yyyyMMdd') =
  TO_date(:var1, 'yyyymmdd');
```

同案例1谓词无法推入

```
-------检查10053--------------
Cost-Based Join Predicate Push-down
***********************************
JPPD: Checking validity of push-down in query block SEL$1
(#1)
JPPD:   Checking validity of push-down from query block
SEL$1 (#1) to query block SET$1 (#2)
JPPD:     Passed validity checks
JPPD:     JPPD bypassed: Cartesian product found

--------检查fix control设置----
SQL> select bugno,value,sql_feature,description from
v$system_fix_control where bugno=9380298;

     BUGNO      VALUE SQL_FEATURE           DESCRIPTION
---------- ---------- --------------------- --------------
----------------------------------------------------------
   9380298          0 QKSFM_JPPD_9380298    JPPD for
cartesian joins under fix control
```

```
Execution Plan
----------------------------------------------------------
Plan hash value: 275940462

-------------------------------------------------------------------------------------------------------------
----------
| Id  | Operation                           | Name                          | Rows  | Bytes | Cost (%CPU)|
Time     |
-------------------------------------------------------------------------------------------------------------
----------
|   0 | SELECT STATEMENT                    |                               |     1 |    13 |  3418   (1)|
00:00:42 |
|   1 |  SORT AGGREGATE                     |                               |     1 |    13 |            |
|
|   2 |   VIEW                              |                               |  1877 | 24401 |  3418   (1)|
00:00:42 |
|   3 |    UNION-ALL                        |                               |       |       |            |
|
|   4 |     NESTED LOOPS SEMI               |                               |     1 |    98 |    68   (0)|
00:00:01 |
|   5 |      NESTED LOOPS                   |                               |     1 |    86 |    66   (0)|
00:00:01 |
|   6 |       TABLE ACCESS BY INDEX ROWID   | AC_WRTOFFREC_1408_3           |    13 |   624 |     4   (0)|
00:00:01 |
|*  7 |        INDEX RANGE SCAN             | IDX_WRTOFFREC_IDNO_1408_3     |    13 |       |     2   (0)|
00:00:01 |
|   8 |       VIEW                          | AC_ACCTBOOKCHG_201408         |     1 |    38 |     5   (0)|
00:00:01 |
|   9 |        UNION-ALL PARTITION          |                               |       |       |            |
|
|* 10 |         TABLE ACCESS BY INDEX ROWID | AC_ACCTBOOKCHG_1408_0         |     1 |    38 |     4   (0)|
00:00:01 |
|* 11 |          INDEX RANGE SCAN           | IDX_PAYSNIO_ACCTBOOKCHG_1408_0 |     1 |       |     3   (0)|
00:00:01 |
                ...
|* 28 |         TABLE ACCESS BY INDEX ROWID | AC_ACCTBOOKCHG_1408_9         |     1 |    38 |     4   (0)|
00:00:01 |
|* 29 |          INDEX RANGE SCAN           | IDX_PAYSNIO_ACCTBOOKCHG_1408  |     1 |       |     3   (0)|
00:00:01 |
|  30 |      PARTITION RANGE ITERATOR       |                               | 56930 |   667K|     2   (0)|
00:00:01 |
|* 31 |       TABLE ACCESS BY LOCAL INDEX ROWID| AC_CONTRACT_INFO           | 56930 |   667K|     2   (0)|
00:00:01 |
|* 32 |        INDEX UNIQUE SCAN            | PK_AC_CONTRACT_INFO_MHH       |     1 |       |     1   (0)|
00:00:01 |
|  33 |     NESTED LOOPS SEMI               |                               |  1876 | 95676 |  3350   (1)|
00:00:41 |
|  34 |      TABLE ACCESS BY INDEX ROWID    | AC_ACCTPAYREC_1408            |  1876 | 73164 |   347   (1)|
00:00:05 |
|* 35 |       INDEX RANGE SCAN              | IDX_AC_ACCTPAYREC_1408_R      |   750 |       |     2   (0)|
00:00:01 |
|  36 |      PARTITION RANGE ITERATOR       |                               |   230K|  2700K|     2   (0)|
00:00:01 |
|* 37 |       TABLE ACCESS BY LOCAL INDEX ROWID| AC_CONTRACT_INFO           |   230K|  2700K|     2   (0)|
00:00:01 |
|* 38 |        INDEX UNIQUE SCAN            | PK_AC_CONTRACT_INFO_MHH       |     1 |       |     1   (0)|
00:00:01 |
-------------------------------------------------------------------------------------------------------------
----------

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

   7 - access("B"."ID_NO"=TO_NUMBER(:V2))
```

开启fix后正常

# 3.3 SPA报告分析和优化实施-优化案例3之无法走索引问题

```
SELECT COUNT(1)
      FROM UR_USERTEAM_INFO_CHG
       WHERE INSERT_TIME >= TRUNC(SYSDATE)
       AND (OPER_TYPE > 'D' OR OPER_TYPE < 'D');
```

Execution Plan

老库执行计划，新库全表扫描

Plan hash value: 2460767915

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 1 | 10 | 892 (1) | 00:00:11 |
| 1 | SORT AGGREGATE | | 1 | 10 | | |
| 2 | CONCATENATION | | | | | |
| * 3 | TABLE ACCESS BY INDEX ROWID | UR_USERTEAM_INFO_CHG | 1 | 10 | 2 (0) | 00:00:01 |
| * 4 | INDEX RANGE SCAN | IDX_UR_USERTEAM_INFO_CHG_03 | 1 | | 2 (0) | 00:00:01 |
| * 5 | TABLE ACCESS BY INDEX ROWID | UR_USERTEAM_INFO_CHG | 1 | 10 | 889 (1) | 00:00:11 |
| * 6 | INDEX RANGE SCAN | IDX_UR_USERTEAM_INFO_CHG_03 | 50440 | | 87 (2) | 00:00:02 |

Predicate Information (identified by operation id):

```
3 - filter("INSERT_TIME">=TRUNC(SYSDATE@!))
4 - access("OPER_TYPE">'D')
5 - filter("INSERT_TIME">=TRUNC(SYSDATE@!))
6 - access("OPER_TYPE"<'D')
    filter(LNNVL("OPER_TYPE">'D'))
```

- 优化思路

1）在11.2.0.4中使用optimizer_features_enable分别测试10.2.0.4-11.2.0.2有效，11.2.0.3无效。

2) 使用SQLT XPLORE测试。

## Completed Tests for Plan 2460767915 87907 57367

| # | Test Id | Test | Baseline Value | Plan Cost | Buffer Gets | CPU (secs) | Disk Reads | ET (secs) | Actual Rows | Estim Rows |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 00244 | ALTER SESSION SET optimizer_features_enable = '10.1.0'; | 11.2.0.4 | 42 | 4 | 0 | 0 | .003 | 1 | 1 |
| 2 | 00245 | ALTER SESSION SET optimizer_features_enable = '10.1.0.3'; | 11.2.0.4 | 42 | 4 | 0 | 0 | .003 | 1 | 1 |
| 3 | 00246 | ALTER SESSION SET optimizer_features_enable = '10.1.0.4'; | 11.2.0.4 | 42 | 4 | 0 | 0 | .003 | 1 | 1 |
| 4 | 00247 | ALTER SESSION SET optimizer_features_enable = '10.1.0.5'; | 11.2.0.4 | 42 | 4 | 0 | 0 | .003 | 1 | 1 |
| 5 | 00248 | ALTER SESSION SET optimizer_features_enable = '10.2.0.1'; | 11.2.0.4 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |
| 6 | 00249 | ALTER SESSION SET optimizer_features_enable = '10.2.0.2'; | 11.2.0.4 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |
| 7 | 00250 | ALTER SESSION SET optimizer_features_enable = '10.2.0.3'; | 11.2.0.4 | 51 | 4 | .01 | 0 | .003 | 1 | 1 |
| 8 | 00251 | ALTER SESSION SET optimizer_features_enable = '10.2.0.4'; | 11.2.0.4 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |
| 9 | 00252 | ALTER SESSION SET optimizer_features_enable = '10.2.0.5'; | 11.2.0.4 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |
| 10 | 00253 | ALTER SESSION SET optimizer_features_enable = '11.1.0.6'; | 11.2.0.4 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |
| 11 | 00254 | ALTER SESSION SET optimizer_features_enable = '11.1.0.7'; | 11.2.0.4 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |
| 12 | 00255 | ALTER SESSION SET optimizer_features_enable = '11.2.0.1'; | 11.2.0.4 | 51 | 4 | .01 | 0 | .003 | 1 | 1 |
| 13 | 00256 | ALTER SESSION SET optimizer_features_enable = '11.2.0.2'; | 11.2.0.4 | 51 | 4 | .01 | 0 | .003 | 1 | 1 |
| 14 | 00755 | ALTER SESSION SET "_fix_control" = '8275054:0'; | 8275054:1 | 51 | 4 | 0 | 0 | .003 | 1 | 1 |

- BUG NO：8275054

simplify range to inequality，将谓词改成<>.

# 3.3 SPA报告分析和优化实施-优化案例4之优化器新特性导致的问题(1)

- 10g执行正常，11.2执行报无效的日期错误

```
select r.proj_id,
               min(to_date(lt.enum_value_code, 'yyyy-mm-dd')) cy_date
          from task_struts_runtime_data r,
               (select t.ts_values_id,
                       t.enum_value_code,
                       l.ts_label_id,
                       l.name,
                       l.data_type,
                       l.is_null,
                       l.is_key_word
                  from task_struts_values t, TASK_STRUTS_LABEL l
                 where t.ts_label_id = l.ts_label_id
                   and t.sts = 'A'
                   and l.sts = 'A'
                   and l.standard_code = 1205) lt
         where r.ts_values_id = lt.ts_values_id
           and r.sts = 'A'
           and lt.enum_value_code is not null
           and length(lt.enum_value_code)>=8
           and length(lt.enum_value_code)<=10
           group by r.proj_id;
ERROR at line 2:
ORA-01841: (full) year must be between -4713 and +9999, and not be 0
```

字段存储数字，日期格式字符

- 问题分析

正确执行计划LT关联查询先执行，之后与外表关联

错误执行计划是TASK_STRUTS_VALUES先与外表关联然后分组，作为VIEW再与TASK_STRUTS_LABEL关联，导致报错。

- 正确执行计划计划

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 962 | 69264 | 3108 (2) | 00:00:38 |
| 1 | HASH GROUP BY | | 962 | 69264 | 3108 (2) | 00:00:38 |
| * 2 | HASH JOIN | | 962 | 69264 | 3107 (2) | 00:00:38 |
| * 3 | HASH JOIN | | 962 | 54834 | 2430 (2) | 00:00:30 |
| * 4 | TABLE ACCESS FULL | TASK_STRUTS_VALUES | 962 | 43290 | 922 (2) | 00:00:12 |
| * 5 | TABLE ACCESS FULL | TASK_STRUTS_LABEL | 17500 | 205K | 1508 (2) | 00:00:19 |
| * 6 | TABLE ACCESS FULL | TASK_STRUTS_RUNTIME_DATA | 403K | 5914K | 675 (2) | 00:00:09 |

- 出错计划

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 965 | 32810 | 2862 (3) | 00:00:35 |
| 1 | HASH GROUP BY | | 965 | 32810 | 2862 (3) | 00:00:35 |
| * 2 | HASH JOIN | | 965 | 32810 | 2861 (3) | 00:00:35 |
| 3 | VIEW | VW_GBC_9 | 965 | 21230 | 1444 (3) | 00:00:18 |
| 4 | HASH GROUP BY | | 965 | 50180 | 1444 (3) | 00:00:18 |
| * 5 | HASH JOIN | | 965 | 50180 | 1443 (3) | 00:00:18 |
| * 6 | TABLE ACCESS FULL | TASK_STRUTS_VALUES | 965 | 35705 | 845 (3) | 00:00:11 |
| * 7 | INDEX FAST FULL SCAN | PK_DATA | 406K | 5951K | 594 (3) | 00:00:08 |
| * 8 | TABLE ACCESS FULL | TASK_STRUTS_LABEL | 18405 | 215K | 1417 (3) | 00:00:18 |

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

   2 - access("ITEM_1"="L"."TS_LABEL_ID")
   5 - access("R"."TS_VALUES_ID"="T"."TS_VALUES_ID")
   6 - filter(LENGTH("T"."ENUM_VALUE_CODE")<=10 AND LENGTH("T"."ENUM_VALUE_CODE")>=8
              AND "T"."ENUM_VALUE_CODE" IS NOT NULL AND "T"."STS"='A')
   7 - filter("R"."STS"='A')
   8 - filter(TO_NUMBER("L"."STANDARD_CODE")=1205 AND "L"."STS"='A')
```

# 3.3 SPA报告分析和优化实施-优化案例4之优化器新特性导致的问题(2)

- 分析10053

```
****************************************
Cost-Based Group-By/Distinct Placement
****************************************
GBP/DP: Checking validity of GBP/DP for query block SEL$F5BB74E1 (#1)
GBP: Checking validity of group-by placement for query block
SEL$F5BB74E1 (#1)

GBP: Using search type: exhaustive
GBP: Considering group-by placement on query block SEL$F5BB74E1 (#1)
GBP: Starting iteration 1, state space = (1,2,3) : (0,0,0)
GBP: Original query
SELECT "VW_GBC_1"."ITEM_3" "PROJ_ID", MIN("VW_GBC_1"."ITEM_2")
"CY_DATE"
  FROM (SELECT "T"."TS_LABEL_ID" "ITEM_1",
               MIN(TO_DATE("T"."ENUM_VALUE_CODE", 'yyyy-mm-dd'))
"ITEM_2",
               "R"."PROJ_ID" "ITEM_3"
          FROM "PMSUSER"."TASK_STRUTS_VALUES"       "T",
               "PMSUSER"."TASK_STRUTS_RUNTIME_DATA" "R"
         WHERE "T"."STS" = 'A'
           AND LENGTH("T"."ENUM_VALUE_CODE") <= 10
           AND LENGTH("T"."ENUM_VALUE_CODE") >= 8
           AND "T"."ENUM_VALUE_CODE" IS NOT NULL
           AND "R"."STS" = 'A'
           AND "R"."TS_VALUES_ID" = "T"."TS_VALUES_ID"
         GROUP BY "T"."TS_LABEL_ID", "R"."PROJ_ID") "VW_GBC_1",
       "PMSUSER"."TASK_STRUTS_LABEL" "L"
 WHERE "VW_GBC_1"."ITEM_1" = "L"."TS_LABEL_ID"
   AND "L"."STS" = 'A'
   AND "L"."STANDARD_CODE" = 1205
 GROUP BY "VW_GBC_1"."ITEM_3"
```

- SQLT XPLORE

**Plan for Test:00103 ALTER SESSION SET "_optimizer_group_by_placement" = FALSE;**

Plan hash value: 3687732830

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 971 | 62144 | 2853 (3) | 00:00:35 |
| 1 | HASH GROUP BY | | 971 | 62144 | 2853 (3) | 00:00:35 |
| * 2 | HASH JOIN | | 971 | 62144 | 2852 (3) | 00:00:35 |
| * 3 | HASH JOIN | | 971 | 47579 | 2250 (2) | 00:00:28 |
| * 4 | TABLE ACCESS FULL | TASK_STRUTS_VALUES | 971 | 35927 | 844 (3) | 00:00:11 |
| * 5 | TABLE ACCESS FULL | TASK_STRUTS_LABEL | 17831 | 208K | 1406 (2) | 00:00:17 |
| * 6 | INDEX FAST FULL SCAN | PK_DATA | 408K | 5986K | 599 (3) | 00:00:08 |

Predicate Information (identified by operation id):

```
2 - access("R"."TS_VALUES_ID"="T"."TS_VALUES_ID")
3 - access("T"."TS_LABEL_ID"="L"."TS_LABEL_ID")
4 - filter(LENGTH("T"."ENUM_VALUE_CODE")>=8 AND LENGTH("T"."ENUM_VALUE_CODE")<=10
          AND "T"."ENUM_VALUE_CODE" IS NOT NULL AND "T"."STS"='A')
5 - filter(TO_NUMBER("L"."STANDARD_CODE")=1205 AND "L"."STS"='A')
6 - filter("R"."STS"='A')
```

关闭隐含参数

# 3.3 SPA报告分析和优化实施-优化案例5之统计信息问题(1)

```
SELECT  DSS_PAGE_B.*
 FROM (SELECT DSS_PAGE_A.*, ROWNUM RN
         FROM (SELECT B.GROUP_NAME,
                      C.BRAND_NAME,
                      D.RES_NAME,
                      E.STATUS_NAME,
                      F.HLR_CODE,
                      COUNT(1)
                 FROM RS_SIM_INFO_VIEW A,
                      DCHNGROUPMSG     B,
                      RS_BRAND_DICT    C,
                      RS_CODE_DICT     D,
                      RS_STATUS_DICT   E,
                      RS_IMSIHLR_REL   F
                WHERE A.GROUP_ID = '127000278'
                  AND A.SIM_TYPE IN
                      (SELECT RES_CODE FROM RS_CODE_DICT WHERE '1' = '1')
                  AND '1' = '1'
                  AND '1' = '1'
                  AND A.IMSI_HEAD = F.IMSI_HEAD
                  AND A.SIM_TYPE = D.RES_CODE
                  AND D.BRAND_CODE = C.BRAND_CODE
                  AND E.KIND_CODE = 'K'
                  AND A.SIM_STATUS = E.STATUS_CODE
                  AND B.GROUP_ID = A.GROUP_ID
                  AND A.SIM_NO >= '89860005126436720074'
                  AND A.SIM_NO <= '8986005126436720050'
                GROUP BY B.GROUP_NAME,
                         C.BRAND_NAME,
                         D.RES_NAME,
                         E.STATUS_NAME,
                         F.HLR_CODE
                ORDER BY B.GROUP_NAME,
                         C.BRAND_NAME,
                         D.RES_NAME,
                         E.STATUS_NAME,
                         F.HLR_CODE) DSS_PAGE_A) DSS_PAGE_B
WHERE DSS_PAGE_B.RN <= 100
  AND DSS_PAGE_B.RN > 0
```

- 正确执行计划计划

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 1037 | 161K | 2226 (1) | 00:00:27 |
| * 1 | VIEW | | 1037 | 161K | 2226 (1) | 00:00:27 |
| 2 | COUNT | | | | | |
| 3 | VIEW | | 1037 | 147K | 2226 (1) | 00:00:27 |
| 4 | SORT GROUP BY | | 1037 | 165K | 2226 (1) | 00:00:27 |
| * 5 | HASH JOIN | | 1037 | 165K | 2224 (1) | 00:00:27 |
| * 6 | HASH JOIN | | 983 | 142K | 2214 (1) | 00:00:27 |
| 7 | TABLE ACCESS BY INDEX ROWID | RS_STATUS_DICT | 9 | 108 | 2 (0) | 00:00:01 |
| * 8 | INDEX RANGE SCAN | IDX_STATUSDICT_01 | 9 | | 1 (0) | 00:00:01 |
| * 9 | HASH JOIN | | 1027 | 136K | 2213 (1) | 00:00:27 |
| * 10 | HASH JOIN | | 265 | 28885 | 8 (0) | 00:00:01 |
| 11 | NESTED LOOPS | | 34 | 2686 | 5 (0) | 00:00:01 |
| 12 | TABLE ACCESS BY INDEX ROWID | DCHNGROUPMSG | 1 | 62 | 2 (0) | 00:00:01 |
| * 13 | INDEX UNIQUE SCAN | IDX_GROUPMSG_01 | 1 | | 2 (0) | 00:00:01 |
| 14 | TABLE ACCESS FULL | RS_BRAND_DICT | 34 | 578 | 3 (0) | 00:00:01 |
| 15 | TABLE ACCESS FULL | RS_CODE_DICT | 405 | 12150 | 3 (0) | 00:00:01 |
| 16 | VIEW | RS_SIM_INFO_VIEW | 1571 | 42417 | 2204 (1) | 00:00:27 |
| 17 | UNION-ALL | | | | | |
| 18 | PARTITION RANGE ALL | | 675 | 34425 | 1458 (1) | 00:00:18 |
| * 19 | TABLE ACCESS BY LOCAL INDEX ROWID | RS_SIM_INFO_DEAD | 675 | 34425 | 1458 (1) | 00:00:18 |
| * 20 | INDEX RANGE SCAN | IDX_SIMDEAD_03 | 3150 | | 35 (0) | 00:00:01 |
| 21 | PARTITION RANGE ALL | | 20 | 1020 | 56 (0) | 00:00:01 |
| * 22 | TABLE ACCESS BY LOCAL INDEX ROWID | RS_SIM_INFO_PRESELL | 20 | 1020 | 56 (0) | 00:00:01 |
| * 23 | INDEX RANGE SCAN | IDX_SIMPRESELL_03 | 130 | | 28 (0) | 00:00:01 |
| 24 | PARTITION RANGE ALL | | 183 | 9333 | 450 (0) | 00:00:06 |
| * 25 | TABLE ACCESS BY LOCAL INDEX ROWID | RS_SIM_INFO_SOLD | 183 | 9333 | 450 (0) | 00:00:06 |
| * 26 | INDEX RANGE SCAN | IDX_SIMSOD_03 | 807 | | 30 (0) | 00:00:01 |
| 27 | PARTITION RANGE ALL | | 693 | 35343 | 240 (0) | 00:00:03 |
| * 28 | TABLE ACCESS BY LOCAL INDEX ROWID | RS_SIM_INFO_UNSELL | 693 | 35343 | 240 (0) | 00:00:03 |
| * 29 | INDEX RANGE SCAN | IDX_SIMUNSELL_03 | 4507 | | 38 (0) | 00:00:01 |
| 30 | TABLE ACCESS FULL | RS_IMSIHLR_REL | 8881 | 88215 | 10 (0) | 00:00:01 |

Predicate Information (identified by operation id):

正确的是走GROUP_ID索引

# 3.3 SPA报告分析和优化实施-优化案例5之统计信息问题(2)

- 错误执行计划计划

| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
|---|---|---|---|---|---|---|
| 0 | SELECT STATEMENT | | 3 | 477 | 38 (6) | 00:00:01 |
| * 1 | VIEW | | 3 | 477 | 38 (6) | 00:00:01 |
| 2 | COUNT | | | | | |
| 3 | VIEW | | 3 | 438 | 38 (6) | 00:00:01 |
| 4 | SORT GROUP BY | | 3 | 489 | 38 (6) | 00:00:01 |
| * 5 | HASH JOIN | | 3 | 489 | 37 (3) | 00:00:01 |
| 6 | TABLE ACCESS BY INDEX ROWID | RS_IMSIHLR_REL | 1 | 15 | 2 (0) | 00:00:01 |
| 7 | NESTED LOOPS | | 3 | 453 | 35 (0) | 00:00:01 |
| * 8 | HASH JOIN | | 3 | 408 | 30 (0) | 00:00:01 |
| * 9 | HASH JOIN | | 4 | 476 | 27 (0) | 00:00:01 |
| 10 | VIEW | RS_SIM_INFO_VIEW | 4 | 108 | 22 (0) | 00:00:01 |
| 11 | UNION-ALL | | | | | |
| * 12 | TABLE ACCESS BY GLOBAL INDEX ROWID | RS_SIM_INFO_DEAD | 1 | 51 | 5 (0) | 00:00:01 |
| * 13 | INDEX RANGE SCAN | IDX_SIMDEAD_01 | 2 | | 3 (0) | 00:00:01 |
| * 14 | TABLE ACCESS BY GLOBAL INDEX ROWID | RS_SIM_INFO_PRESELL | 1 | 51 | 10 (0) | 00:00:01 |
| * 15 | INDEX RANGE SCAN | IDX_SIMPRESELL_01 | 22 | | 2 (0) | 00:00:01 |
| * 16 | TABLE ACCESS BY GLOBAL INDEX ROWID | RS_SIM_INFO_SOLD | 1 | 51 | 4 (0) | 00:00:01 |
| * 17 | INDEX RANGE SCAN | IDX_SIMSOD_01 | 1 | | 3 (0) | 00:00:01 |
| * 18 | TABLE ACCESS BY GLOBAL INDEX ROWID | RS_SIM_INFO_UNSELL | 1 | 51 | 3 (0) | 00:00:01 |
| * 19 | INDEX RANGE SCAN | IDX_SIMUNSELL_01 | 2 | | 2 (0) | 00:00:01 |
| 20 | NESTED LOOPS | | 405 | 37260 | 5 (0) | 00:00:01 |
| 21 | TABLE ACCESS BY INDEX ROWID | DCHNGROUPMSG | 1 | 62 | 2 (0) | 00:00:01 |
| * 22 | INDEX UNIQUE SCAN | IDX_GROUPMSG_01 | 1 | | 2 (0) | 00:00:01 |
| 23 | TABLE ACCESS FULL | RS_CODE_DICT | 405 | 12150 | 3 (0) | 00:00:01 |
| 24 | TABLE ACCESS FULL | RS_BRAND_DICT | 34 | 578 | 3 (0) | 00:00:01 |
| * 25 | INDEX RANGE SCAN | IDX_IMSIHLR_01 | 1 | | 1 (0) | 00:00:01 |
| 26 | TABLE ACCESS BY INDEX ROWID | RS_STATUS_DICT | 9 | 108 | 2 (0) | 00:00:01 |
| * 27 | INDEX RANGE SCAN | IDX_STATUSDICT_01 | 9 | | 1 (0) | 00:00:01 |

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

   1 - filter("DSS_PAGE_B"."RN"<=100 AND "DSS_PAGE_B"."RN">0)
   5 - access("A"."SIM_STATUS"="E"."STATUS_CODE")
   8 - access("D"."BRAND_CODE"="C"."BRAND_CODE")
   9 - access("A"."SIM_TYPE"="D"."RES_CODE")
  12 - filter("GROUP_ID"='127000278')
  13 - access("SIM_NO">='8986000512643672007 4' AND "SIM_NO"<='8986005126436720050')
  14 - filter("GROUP_ID"='127000278')
  15 - access("SIM_NO">='8986000512643672007 4' AND "SIM_NO"<='8986005126436720050')
  16 - filter("GROUP_ID"='127000278')
  17 - access("SIM_NO">='8986000512643672007 4' AND "SIM_NO"<='8986005126436720050')
  18 - filter("GROUP_ID"='127000278')
  19 - access("SIM_NO">='8986000512643672007 4' AND "SIM_NO"<='8986005126436720050')
```

- 为什么走错索引

由于SIM_NO是CHAR字符串类型，长度20，CBO内部计算选择性会先将字符串转为RAW，然后RAW转为数字，左起ROUND 15位。因此，可能字符串值差别大的，转换成数字后值差别不大，导致选择性计算错误。以RS_SIM_INFO_DEAD为例：

NUM_ROWS：139495263

NUM_DISTINCT:139495263

LOW_VALUE:38383836303038303132393830313632303836 20

HIGH_VALUE：4E554C4C3530383930202020202020202020202020

NUM_NULLS:0

**--谓词最大最小值内部一样**

```
SQL> select get_internal_value('89860005126436720074') from dual;

GET_INTERNAL_VALUE('89860005126436720074')
--------------------------------------------------------------------------------
29192917497290600000000000000000000000

SQL>
SQL> select get_internal_value('8986005126436720050') from dual;

GET_INTERNAL_VALUE('8986005126436720050')
--------------------------------------------------------------------------------
29192917497290600000000000000000000000
```

# 3.3 SPA报告分析和优化实施-优化案例5之统计信息问题(3)

sel: 1/139495263+1/139495263+(291929174972906000000000000000000000-291929174972906000000000000000000000)/(high_value-low_value)

=1/139495263+1/139495263

card:=139495263*(1/139495263+1/139495263)=2

实际返回行数

```
  COUNT(*)
----------
  29737305
```

估算值与实际值差别巨大

- 解决方法

收集SIM_NO列直方图，由于内部算法有一定限制，导致值不同的字符串，内部计算值可能一致，所以收集直方图后，针对字符串值不同，但是转换成数字后相同的，ORACLE会将实际值存储到ENDPOINT_ACTUAL_VALUE中，用于校验，提高执行计划的准确性。

## 3.4 SPA报告分析和优化实施-优化实施

- 改回CBO参数的默认值，需要迭代SPA分析
- 修改fix control设置，需要迭代SPA分析
- 统计信息处理策略
  1）原库未收集统计信息的且无性能下降，新库不收集
  2）原库不准确的，更新（method_opt=>’for all columns size repeat’)
  3) 其它有问题的表，单独收集策略，如建立直方图
  4）LOCK SOME STATISTICS
- SQL PROFILE绑定，终极方法，一般数量较少
- 提前准备实施脚本，升级前一天完成
- 升级后第二天业务高峰期收集dictionary和fixed objects统计信息
- 准备性能保障方案

**多次迭代SPA分析，确保升级平稳过渡**


DBA+社群是中国最大的涵盖各种架构师、数据库、中间件的微信社群！线上分享2次/周、线下沙龙1次/月，顶级峰会6次/年，直接受众10000+，间接影响50万+ITer。DBA+社群致力于搭建一个学习交流、专业人脉、跨界合作的公益平台，更多精彩请持续关注dbaplus微信订阅号！

# THANKS FOR WATCHING