Bind Peeking – The Endless Tuning Nightmare

Penny Cookson

Managing Director and Principal Consultant

Working with Oracle products since 1987

Oracle Magazine Educator of the Year 2004

♦ Oracle ACE

www.sagecomputing.com.au
penny@sagecomputing.com.au
Characteristics

I haven’t changed anything

Its really slow this morning

I did the same thing yesterday and it was fine

Actually its OK now

No its not

Thank you so much you’ve fixed it (I haven’t done anything)
Oracle Version < 9

SELECT COUNT(l.quantity) FROM bookings_skew l WHERE resource_code = :v1;

‘BRLG’

‘PC1’

How many rows do I expect?

<Version 9 database
No bind peeking

<Version 9 database
No bind peeking

SELECT COUNT(l.quantity) FROM bookings_skew_large l WHERE resource_code = :v1;

FULL SCAN

FULL SCAN
Oracle Version < 9

SELECT COUNT(l.quantity) FROM bookings_skew l WHERE resource_code = :v1;

How many rows do I expect?

‘PC1’

‘BRLG’

<Version 9 database
No bind peeking
What is Bind Peeking?

SELECT COUNT(l.quantity) FROM bookings_skew l WHERE resource_code = :v1;

How many rows do I expect?

Full SCAN

'RBLG'

'PC1'

>=Version 9 database Bind peeking
What is Bind Peeking?

SELECT COUNT(l.quantity) FROM bookings_skew l WHERE resource_code = :v1;

How many rows do I expect?

INDEXED ACCESS

‘PC1’

‘BRLG’

>=Version 9 database
Bind peeking
SINGLE TABLE ACCESS PATH
Column (#3): RESOURCE_CODE(VARCHAR2)
  AvgLen: 5.00  NDV: 9  Nulls: 0  Density: 0.11111
Table: BOOKINGS_SKEW  Alias: L
Access Path: TableScan
  Cost: 7701.80  Resp: 7701.80  Degree: 0
  Cost_io: 7426.00  Cost_cpu: 1605072831
  Resp_io: 7426.00  Resp_cpu: 1605072831
Access Path: index (AllEqRange)
  Index: BK_RESSKEW
  resc_io: 32580.00  resc_cpu: 509053165
  ix_sel: 0.11111  ix_sel_with_filters: 0.11111
  Cost: 32667.47  Resp: 32667.47  Degree: 1
Best:: AccessPath: TableScan
  Cost: 7701.80  Degree: 1  Resp: 7701.80  Card: 614137.56  Bytes: 0

***************************************
Histogram – Majority First

*************************************************
SINGLE TABLE ACCESS PATH

Column (#3): RESOURCE_CODE(VARCHAR2)
AvgLen: 5.00 NDV: 9 Nulls: 0 Density: 9.0892e-008

Histogram: Freq #Bkts: 9 UncompBkts: 5966 EndPtVals: 9
Table: BOOKINGS_SKEW Alias: L

Access Path: TableScan
Cost: 7710.70 Resp: 7710.70 Degree: 0
Cost_io: 7426.00 Cost_cpu: 1656892911
Resp_io: 7426.00 Resp_cpu: 1656892911

Access Path: index (AllEqRange)
Index: BK_RESSKEW
resc_io: 94139.00 resc_cpu: 1468544140
ix_sel: 0.31612 ix_sel_with_filters: 0.31612
Cost: 94391.34 Resp: 94391.34 Degree: 1

Best:: AccessPath: TableScan
Cost: 7710.70 Degree: 1 Resp: 7710.70 Card: 1727558.30 Bytes: 0

***************************************
**Histogram – Minority First**

************************************************

**SINGLE TABLE ACCESS PATH**

Column (#3): RESOURCE_CODE(VARCHAR2)  
AvgLen: 5.00 NDV: 9 Nulls: 0 Density: 9.0892e-008  
Histogram: Freq #Bkts: 9 UncompBkts: 5966 EndPtVals: 9  
Table: BOOKINGS_SKEW Alias: L  
Card: **Original: 5464800 Rounded: 43968** Computed: 43967.55  
Non Adjusted: 43967.55  
Access Path: TableScan  
Cost: 7693.35 Resp: 7693.35 Degree: 0  
Cost_io: 7426.00 Cost_cpu: 1555877511  
Resp_io: 7426 Resp_cpu: 1555877511  
Access Path: index (AllEqRange)  
Index: BK_RESSKEW  
resc_io: 2399.00 resc_cpu: 37397785  
ix_sel: 0.0080456 ix_sel_with_filters: 0.0080456  
Cost: 2405.43 Resp: 2405.43 Degree: 1  
Best:: AccessPath: IndexRange Index: BK_RESSKEW  
Cost: 2405.43 Degree: 1 Resp: 2405.43 Card: 43967.55 Bytes: 0  
************************************************

<table>
<thead>
<tr>
<th>RESO</th>
<th>COUNT(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCR1</td>
<td>495711</td>
</tr>
<tr>
<td>CONF</td>
<td>495720</td>
</tr>
<tr>
<td>LNCH</td>
<td>743576</td>
</tr>
<tr>
<td>BRSM</td>
<td>743583</td>
</tr>
<tr>
<td>PC1</td>
<td>47858</td>
</tr>
<tr>
<td>FLP1</td>
<td>495720</td>
</tr>
<tr>
<td>BRLG</td>
<td>1739277</td>
</tr>
<tr>
<td>TAP1</td>
<td>247864</td>
</tr>
<tr>
<td>VCR2</td>
<td>495715</td>
</tr>
</tbody>
</table>
### What is the CBO OK at

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Literal/Bind Var</th>
<th>Histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even Distribution</td>
<td>Equality</td>
<td>Literal</td>
<td>N/A</td>
</tr>
<tr>
<td>Even Distribution</td>
<td>Equality</td>
<td>Bind</td>
<td>N/A</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Literal</td>
<td>NO</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Literal</td>
<td>YES</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Bind</td>
<td>NO</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Bind</td>
<td>YES</td>
</tr>
</tbody>
</table>
Histograms

Versions < 9: Histograms are no use with bind variables

Versions >= 9: Histograms are worse than no use with bind variables

Unless…

Each distinct SQL uses only either minority or majority values but not both

Or you don’t keep/use the statements in the shared pool (in which case you might as well use literals)
So if I have no Skewed Data I am OK?

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Literal/Bind Var</th>
<th>Histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even Distribution</td>
<td>Equality</td>
<td>Literal</td>
<td>N/A</td>
</tr>
<tr>
<td>Even Distribution</td>
<td>Equality</td>
<td>Bind</td>
<td>N/A</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Literal</td>
<td>NO</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Literal</td>
<td>YES</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Bind</td>
<td>NO</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Bind</td>
<td>YES</td>
</tr>
</tbody>
</table>
MINORITY RANGE FIRST

********************************************************************************
SINGLE TABLE ACCESS PATH
********************************************************************************

Column (#2): EVENT_NO(NUMBER)
AvgLen: 5.00 NDV: 99178 Nulls: 0 Density: 1.0083e-005 Min: 211 Max: 100175
Table: BOOKINGS_EVEN Alias: L
Access Path: TableScan
Cost: 7742.79 Resp: 7742.79 Degree: 0
Cost_io: 7492.00 Cost_cpu: 1459499283
Resp_io: 7492.00 Resp_cpu: 1459499283

Access Path: index (RangeScan)
Index: BK_EVTEVEN
resc_io: 4992.00 resc_cpu: 37790328
ix_sel: 9.0040e-004 ix_sel_with_filters: 9.0040e-004
Cost: 4998.49 Resp: 4998.49 Degree: 1
Best:: AccessPath: IndexRange  Index: BK_EVTEVEN
Cost: 4998.49 Degree: 1 Resp: 4998.49 Card: 4972.66 Bytes: 0
********************************************************************************

**MIN**  **MAX**
-------  ------
100      100200
<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Literal/Bind Var</th>
<th>Histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even Distribution</td>
<td>Equality</td>
<td>Literal</td>
<td>N/A</td>
</tr>
<tr>
<td>Even Distribution</td>
<td>Equality</td>
<td>Bind</td>
<td>N/A</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Literal</td>
<td>NO</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Literal</td>
<td>YES</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Bind</td>
<td>NO</td>
</tr>
<tr>
<td>Skewed</td>
<td>Equality</td>
<td>Bind</td>
<td>YES</td>
</tr>
<tr>
<td>Even Distribution</td>
<td>Range</td>
<td>Bind</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Partitions?

Which statistics shall we use?

Partition empty

Resource_code has 1 value

Resource_code has approx 5600 values

<100 <200 <300 <400 <500

Global statistics resource_code has 6116 values
What Can We do?

Own code:-

Use Literals – but only for skewed data
Write separate code for minority/majority
  Give user separate menu options
Daily report v Annual report
Build sql dynamically with hints for various cases
User not aware
(and keep your histograms)
What Can We do?

Package:-
If we have no histogram what is it doing? → Full scan

Is a full scan acceptable? Yes
  No histogram
  Turn off bind peeking
  _optim_peek_user_binds=false

Is a full scan acceptable? No
  Run the majority ones first

Different session for majority/minority cases and change session variables
What Can We do?

Package:-
If we have no histogram what is it doing? → Index scan

(So none of these will help
   No histogram
   Turn off bind peeking
   _optim_peek_user_binds=false)

Remove index?
   Create an outline and force a full scan
Run the majority ones first

Different session for majority/minority cases and change session variables
What Can We do?

Package:-

Purge the statement from the shared pool each execution (10.2.0.4):

```
dbms_shared_pool.purge
    ('&address, &hash_value','c')
```
## Oracle 11g - Data

<table>
<thead>
<tr>
<th>RESO</th>
<th>NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCR1</td>
<td>524288</td>
</tr>
<tr>
<td>LNCH</td>
<td>786432</td>
</tr>
<tr>
<td>CONF</td>
<td>524288</td>
</tr>
<tr>
<td>BRSM</td>
<td>786432</td>
</tr>
<tr>
<td>PC1</td>
<td>262144</td>
</tr>
<tr>
<td>FLPC</td>
<td>524288</td>
</tr>
<tr>
<td>BRLG</td>
<td>1310720</td>
</tr>
<tr>
<td>TAP1</td>
<td>262144</td>
</tr>
<tr>
<td>VCR2</td>
<td>524288</td>
</tr>
</tbody>
</table>

---

**sum**  
5505024
### Oracle 11g - Data

<table>
<thead>
<tr>
<th>RESO</th>
<th>NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC2</td>
<td>99</td>
</tr>
<tr>
<td>PC3</td>
<td>99</td>
</tr>
<tr>
<td>PC4</td>
<td>99</td>
</tr>
<tr>
<td>PC5</td>
<td>99</td>
</tr>
<tr>
<td>PC6</td>
<td>99</td>
</tr>
<tr>
<td>PC7</td>
<td>99</td>
</tr>
<tr>
<td>PC8</td>
<td>99</td>
</tr>
<tr>
<td>PC9</td>
<td>99</td>
</tr>
<tr>
<td>PC11</td>
<td>101</td>
</tr>
<tr>
<td>PC12</td>
<td>102</td>
</tr>
<tr>
<td>PC13</td>
<td>103</td>
</tr>
<tr>
<td>PC14</td>
<td>104</td>
</tr>
<tr>
<td>PC15</td>
<td>199</td>
</tr>
<tr>
<td>PC16</td>
<td>399</td>
</tr>
<tr>
<td>PC17</td>
<td>799</td>
</tr>
<tr>
<td>PC18</td>
<td>999</td>
</tr>
<tr>
<td>PC19</td>
<td>1999</td>
</tr>
<tr>
<td>PC20</td>
<td>3999</td>
</tr>
<tr>
<td>PC10</td>
<td>199</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RESO</th>
<th>NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC1</td>
<td>47865</td>
</tr>
<tr>
<td>TAP1</td>
<td>247853</td>
</tr>
<tr>
<td>FLPC</td>
<td>495713</td>
</tr>
<tr>
<td>CONF</td>
<td>495714</td>
</tr>
<tr>
<td>VCR2</td>
<td>495718</td>
</tr>
<tr>
<td>VCR1</td>
<td>495720</td>
</tr>
<tr>
<td>BRSM</td>
<td>743571</td>
</tr>
<tr>
<td>LNCH</td>
<td>743581</td>
</tr>
<tr>
<td>BRLG</td>
<td>1737243</td>
</tr>
</tbody>
</table>

**BOOKINGS_SKEW**

SUM 5505024
Bind Peeking + Adaptive Cursors  Summary

Statements with bind variables (+histograms) are bind sensitive

The first time you execute a statement with different selectivity it uses the original plan

The second time it changes the plan and become bind aware

New values will use a plan for the appropriate selectivity range

Be careful when statements become invalidated by:-
  Gathering statistics
  Flushing the shared pool
  Restarting the database
  Or when they are aged out
Adaptive Cursors Functionality

Adaptive Cursors 11.1.0.6
- Bind variable with Equality and Histogram
- Not for range conditions

Adaptive Cursors 11.1.0.7
- Bind variable with Equality and Histogram
- Range conditions
- Do Not Support LIKE

Future?
- LIKE
- SELECT /*+ HINT TO MAKE IT BIND AWARE */
Gathering Statistics

Early CBO: “Make sure you gather statistics regularly”

Later CBO: “Don’t gather statistics unless data patterns change”

BUT

If you have new majority values you need to recreate the histogram
SQL Plan Management

Manual capture
DBMS_SPM.LOAD_PLANS_FROM_SQLSET
DBMS_SPM.LOAD_PLANS_FROM_CURSOR_CACHE

Auto capture of repeatable statements
OPTIMIZER_CAPTURE_SQL_PLAN_BASELINE = TRUE

SQL Management Base
Baseline = (Stored and Accepted plans)

Manual load/accept of new plan
DBMS_SPM.LOAD_PLANS_FROM_SQLSET
DBMS_SPM.LOAD_PLANS_FROM_CURSOR_CACHE

New Plan identified during execution

SQL Tuning Advisor identifies new plan – SQL*Profile accepted

Auto accept of new plan (if it performs better)
DBMS_SPM.EVOLVE_SQL_PLAN_BASELINE

Stored not accepted
SQL Plan Management – Binds

- Will not automatically handle adaptive cursors
- New plan identified on first execution
- New plan recorded as not accepted
- Plan will not evolve
- All bind variable values use same baseline plan
- Plans show as not bind sensitive or aware

<table>
<thead>
<tr>
<th>SQL_TEXT</th>
<th>SQL_ID</th>
<th>CHILD_NUMBER</th>
<th>HASH_VALUE</th>
<th>EXECUTIONS</th>
<th>SQL_PLAN_BASELINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT COUNT(quantity)...</td>
<td>2wpuu88g5an3m</td>
<td>0</td>
<td>508907635</td>
<td>2 (null)</td>
<td>(null)</td>
</tr>
<tr>
<td>SELECT COUNT(quantity)...</td>
<td>2wpuu88g5an3m</td>
<td>1</td>
<td>508907635</td>
<td>2 (null)</td>
<td>SYS_SQL_PLAN_845e4b385166fb0e</td>
</tr>
<tr>
<td>SELECT COUNT(quantity)...</td>
<td>2wpuu88g5an3m</td>
<td>2</td>
<td>508907635</td>
<td>2 (null)</td>
<td>SYS_SQL_PLAN_845e4b385166fb0e</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQL_ID</th>
<th>CHILD_NUMBER</th>
<th>IS_BIND_SENSITIVE</th>
<th>IS_BIND_AWARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2wpuu88g5an3m</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>2wpuu88g5an3m</td>
<td>1</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>2wpuu88g5an3m</td>
<td>2</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>
Plan name: SYS_SQL_PLAN_845e4b38128d5d7c
Enabled: YES  Fixed: NO  Accepted: NO  Origin: AUTO-CAPTURE

Plan hash value: 927983165

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>8</td>
<td>9717</td>
<td>(1) 00:01:57</td>
</tr>
<tr>
<td>1</td>
<td>SORT AGGREGATE</td>
<td></td>
<td>1</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 2</td>
<td>TABLE ACCESS FULL</td>
<td>BOOKINGS_LARGE</td>
<td>640K</td>
<td>5006K</td>
<td>9717</td>
<td>(1) 00:01:57</td>
</tr>
</tbody>
</table>

Predicate Information (identified by operation id):

2 - filter("RESOURCE_CODE"=:V3)

Plan name: SYS_SQL_PLAN_845e4b385166f00e
Enabled: YES  Fixed: NO  Accepted: YES  Origin: AUTO-CAPTURE

Plan hash value: 1457437069

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>8</td>
<td>35565</td>
<td>(1) 00:07:07</td>
</tr>
<tr>
<td>1</td>
<td>SORT AGGREGATE</td>
<td></td>
<td>1</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>TABLE ACCESS BY INDEX ROWID</td>
<td>BOOKINGS_LARGE</td>
<td>640K</td>
<td>5006K</td>
<td>35565</td>
<td>(1) 00:07:07</td>
</tr>
<tr>
<td>* 3</td>
<td>INDEX RANGE SCAN</td>
<td>BK_RES2</td>
<td>640K</td>
<td></td>
<td>1764</td>
<td>(1) 00:00:22</td>
</tr>
</tbody>
</table>
Evolve SQL Plan Baseline Report

Inputs:

 PLAN_LIST = SYS_SQL_PLAN_845e4b38128d5d7c
 TIME_LIMIT = DBMS_SPM.AUTO_LIMIT
 VERIFY = YES
 COMMIT = YES

Plan: SYS_SQL_PLAN_845e4b38128d5d7c

Plan was verified: Time used 107.25 seconds.
Failed performance criterion: Compound improvement ratio <= 1.2.

<table>
<thead>
<tr>
<th>Execution Status:</th>
<th>Baseline Plan</th>
<th>Test Plan</th>
<th>Improv. Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rows Processed:</td>
<td>COMPLETE</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Elapsed Time(ms):</td>
<td>51902</td>
<td>10310</td>
<td>5.03</td>
</tr>
<tr>
<td>CPU Time(ms):</td>
<td>4078</td>
<td>1281</td>
<td>3.18</td>
</tr>
<tr>
<td>Buffer Gets:</td>
<td>42327</td>
<td>35384</td>
<td>1.2</td>
</tr>
<tr>
<td>Disk Reads:</td>
<td>23204</td>
<td>14800</td>
<td>1.57</td>
</tr>
<tr>
<td>Direct Writes:</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Fetches:</td>
<td>23201</td>
<td>1016</td>
<td>22.79</td>
</tr>
<tr>
<td>Executions:</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Report Summary

Number of SQL plan baselines verified: 1.
Number of SQL plan baselines evolved: 0.
Evolve SQL Plan Baselines

Plans that have not yet been accepted can be evolved (verified) to confirm they are suitable plan baselines.

<table>
<thead>
<tr>
<th>Name</th>
<th>SQL Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS_SQL_PLAN_845e4b38128d5d7c</td>
<td>SELECT COUNT(quantity) FROM bookings_large WHERE...</td>
</tr>
</tbody>
</table>

- Verify Performance: Yes or No
- Time Limit: Auto, Unlimited, Specify 1 minutes
- Action: Report and Accept, Report only

Force Acceptance of the plan
Questions?

www.sagecomputing.com.au
penny@sagecomputing.com.au