Adaptive Cursor Sharing: An Introduction

Harald van Breederode
Oracle University
17-NOV-2010

About Me

- Senior Principal DBA Trainer – Oracle University
- 25 years Unix Experience
- 12 years Oracle DBA Experience
- Oracle8i, 9*i*, *10g* and 11*g* OCP
- Oracle10*g* and Oracle11*g* OCM
- DBA Certification Exam Team Reviewer
- DBA Curriculum Development Reviewer
- Blog: prutser.wordpress.com
- Visually Impaired (Legally Blind)

ORACLE
Certified Master
Oracle Database 11*g*
Administrator

ORACLE

Agenda

- Overview
- SQL statement processing
- Data Skew
- Using Bind variables
- Optimizer Bind Variable Peeking
- Adaptive Cursor Sharing
- Demos
- Questions & Answers

Overview

The goal of this presentation is to answer the following questions:

- What is data skew?
- What are Histograms?
- What is Optimizer Bind Variable Peeking?
- What is Adaptive Cursor Sharing?

Basics of SQL statement processing

- Executing a SQL statement requires a cursor
- Lifetime of a cursor:
  - Open
  - Parse
  - Execute
  - Fetch
  - Close
- The Optimizer generates and selects a execution plan
- Cursors in the library cache can be shared
  - Good for library cache efficiency
  - In general good for performance
- Implemented as parent and child cursors
- `v$sqlarea` and `v$sql`

What is Selectivity?

- Used to calculate the cardinality of a result set
  - Cardinality is the number of rows that needs processing
- Number between 0 and 1
- Basic algorithm: #Rows / #DistinctValues
- Visible in optimizer trace file (event 10053)
- Cardinality greatly influences execution plan selection
- Good selectivity means fewer rows => Index access path
- Bad selectivity means more rows => Full table scan
  - Poorly estimated selectivity may result into sub-optimal execution plans

What is Data Skew?

- Non-uniform distribution of data
- Basically two flavors:
  - Popular and non-popular non-unique values
    - 10,10,10,10,20,10,10,20,10,10,30
  - Non-consecutive unique values
    - 1,2,3,4,5,1001,1002,1003,1004,1005
- Skewed data may cause optimizer selectivity challenges

Question

How can we help the optimizer to better deal with skewed data?

Answer

Create histograms on columns containing skewed data

What is a Histogram?

- Data dictionary objects to store skew metadata
- Introduced in Oracle8i
- Store metadata in 1..254 buckets
- Two different types:
  - Width-balanced (a.k.a. frequency histogram)
  - Height-balanced
- Type depends on number of buckets and number of DV
  - #DV <= #Buckets => Width-balanced
  - #DV > #Buckets => Height-balanced
- Enables the optimizer to better calculate selectivity
- Created by using `METHOD_OPT` clause of `DBMS_STATS`
- Viewed using `DBA_TAB_HISTOGRAMS`

Using Bind Variables

- Goal is to improve sharing of cursors
- Reduce SQL memory usage
- Cure against SQL injection
- Cursor lifetime becomes:
  - OPEN, PARSE, BIND, EXECUTE, FETCH, CLOSE
- May cause optimizer selectivity problems with skewed data
- Should be carefully implemented by SQL developers
- Can be enforced by the `CURSOR_SHARING` parameter
  - `EXACT` – No literal replacement
  - `FORCE` – (8i) Replace all literals by bind variables
  - `SIMILAR` – (9i) Only replace literals when safe to do so

Question

Is there a solution for the bind variable optimizer selectivity problem concerning skewed data?

Answer

Use Optimizer Bind Variable Peeking

Optimizer Bind Variable Peeking

- Goal is to avoid optimizer selectivity problems with Bind variables and skewed data
- Optimizer peeks at value to be bound at hard parse time
- Introduced in Oracle9i
- Peeked value is visible in execution plan using `PEEKED_BINDS` in format clause of `DBMS_XPLAN`

Question

What happens if the peeked value is not representative for future executions?

Answer

Again, we end up with optimizer selectivity problems

Adaptive Cursor Sharing

Goal is to strike a balance between never sharing cursors and always sharing cursors.

- Introduced in Oracle11g
- Enabled by default
- Only applicable for SQL statements with bind variables
- If a Bind variable influences execution plan selection:
  - Cursor is marked `IS_BIND_SENSITIVE`
  - Optimizer monitors cursor executions
  - Cardinality feedback is generated
- Optimizer learns from wrongly calculated selectivity
- A new child cursor is created if a better plan is possible
- The new cursor is marked `IS_BIND_AWARE`

Adaptive Cursor Sharing process flow

- Application executes SQL statement with bind variables
- Optimizer performs the following action at hard parse time
  - Peek at bind variables
  - Generate and select optimal execution plan
  - Cursor is marked `IS_BIND_SENSITIVE` (we assume!)
- During execution, cursor is monitored
- Cardinality feedback is generated upon completion
- At next execution cardinality feedback is used possibly resulting in a new child and if so, then cursor is marked `IS_BIND_AWARE`
- If a previous cursor exists for the same selectivity it is marked as not sharable which leads to cursor flush

Adaptive Cursor Sharing Views

Views containing Adaptive Cursor Sharing information:

- V$SQL
  - Columns: IS_BIND_SENSITIVE, IS_BIND_AWARE
- V$SQL_CS_STATISTICS
  - Adaptive Cursor Sharing execution statistics
- V$SQL_CS_SELECTIVITY
  - Selectivity ranges for Adaptive Cursors
- V$SQL_CS_HISTOGRAM
  - Adaptive Cursor Sharing monitoring information

Interaction between Adaptive Cursor Sharing and the CURSOR_SHARING parameter

- Well-written applications should use:
  - `CURSOR_SHARING=EXACT`
- Misbehaving Oracle8i applications should use:
  - `CURSOR_SHARING=FORCE` (or better, upgrade to 11g!)
- Misbehaving Oracle9i/10g applications should use:
  - `CURSOR_SHARING=SIMILAR`
- Misbehaving Oracle11g applications should use:
  - `CURSOR_SHARING=FORCE`
  - Adaptive Cursor Sharing will compensate over-sharing of cursors caused by `CURSOR_SHARING=FORCE`

Live Demo!

QUESTIONS & ANSWERS

And Finally

Thank you for your kind attention!

For a copy of my demonstration scripts email me at:

Harald.van.Breederode@oracle.com

Remember: prutser.wordpress.com