utoug
utah oracle users group

Dealing with the Optimizer Complexity


Liron Amitzi


liron@brillix.com


@amitzil


https://amitzil.wordpress.com

About Me

- Liron Amitzi
- Oracle DBA since 1998 (and Oracle 7)
- Database consultant since 2002
- Oracle ACE
- An independent senior DB consultant
- Recently moved to Vancouver, BC
- BCOUG president

utoug
utah oracle users group

500+ Technical Experts Helping Peers Globally

ORACLE ACE Program

ORACLE ACE Director

ORACLE ACE

ORACLE ACE Associate

---

3 Membership Tiers

- Oracle ACE Director
- Oracle ACE
- Oracle ACE Associate

bit.ly/OracleACEProgram

Connect:

oracle-ace_ww@oracle.com

Facebook.com/oracleaces

@oracleace

Oracle Groundbreakers

Nominate yourself or someone you know: acenomination.oracle.com

Agenda

- Optimizer overview
- Static optimization
- Dynamic optimization
- Static and dynamic optimization features
- Improving our tuning skills

A Few Years Ago…

What??

- What might cause such a strange behavior?
- How can we debug issues like that?

- Can it be server overload issues?
- Can it be optimizer problem?
- What else can affect the performance in such a way?

The Optimizer Job

- One of the most complex pieces in Oracle
- It has one job, to get the best plan possible for a query as quickly as possible
- The optimizer uses static information (like statistics) to make decisions
- Lately, Oracle made the optimizer much more sophisticated, so it can be aware of data collected during an execution or after it
- And this makes our lives much harder…

Static Optimization

- Static information has existed since CBO was introduced
- This is the optimizer's way to get information about the relevant objects, so it can make smart decisions
- Object statistics, histograms, system statistics, table structure and more are included here
- There are a lot of math calculations
- Oracle is still improving the information collected

Static Optimization Features

- These are some of the features that allow gathering more statistics and in more efficient ways:
  - Extended statistics (11.1) - gathering statistics for a group of columns or for expressions
  - Synopsis (11.1) - allow collecting NDV more accurately
  - New histogram types (12.1) - top frequency and hybrid histograms
  - GTT local stats (18c) - allows session statistics for global temp tables
  - Real-Time stats (19c) - additional statistics info for ongoing DMLs

Dynamic Optimization

- The static optimization information is limited and the optimizer sometimes simply can't know stuff
- With this new world of features, the optimizer can be much more accurate than ever before
- These features kick in when the optimizer realizes something just before, during or after an execution
- Then it can react to this realization on the spot, or save information for future use

Pre-Execution Features

- The optimizer uses these features to get relevant information before it executes the optimization process:
  - Adaptive cursor sharing (11.2) - allow choosing the best plan with different values for bind variables
  - Automatic degree of parallelism (11.2) - determines the degree automatically with regards to resources, and allowing statement queuing
  - Dynamic statistics (12.1) - (FKA dynamic sampling), now also allows persistency of the dynamic information gathered

During or Post Execution

- The optimizer uses these features when it realizes that something was wrong with its calculations:
  - Cardinality feedback (11.2) - when expected and actual cardinality differ, feedback is used to allow better plan generation in the future
  - Adaptive plans (12.1) - during an execution, if actual cardinality is way off, the optimizer can change the plan on the fly

So What is the Problem?

- With dynamic optimization, the optimizer is much less deterministic
- We can't simply copy statistics and some of the data to reproduce the problem in dev
- We can't even generate a plan and expect it to be the one that is used

What Do We Do Now?

- The DBA job is much more complex now
- We don't only need to understand what the optimizer does, but what can affect the plan (and now, there is a lot that can)
- The most important thing is to know as much as we can about the optimizer features and behavior
- Don't trust an offline execution plan alone

What Do We Do Now? (cont.)

The optimizer tells us a lot about what it does, refer to the "Note" part after the execution plan in the output:

```
Note
-----
   - cardinality feedback used for this statement

   - this is an adaptive plan

   - dynamic statistics used: dynamic sampling (level=5)
```

What Do We Do Now (cont.)?

- Useful tools:
  - GATHER_PLAN_STATISTICS hint to capture information of estimated vs. actual cardinality
  - DBMS_XPLAN.display_cursor(format=>'allstats last') to query estimated vs. actual cardinality
  - DBMS_XPLAN.display_cursor(format => 'adaptive') to see adaptive plans information

What Do We Do Now (cont.)?

- Data Dictionary Views:
  - Adaptive Cursor Sharing - the is_bind_sensitive and is_bind_aware columns in V$SQL indicate if the feature is used for the query
  - Dynamic Statistics - the other_xml column in V$SQL_PLAN contains information about dynamic statistics usage
  - Cardinality Feedback - the use_feedback_stats column in V$SQL_SHARED_CURSOR will show if feedback was used for the query
  - Statement Queuing - the status column in V$SQL_MONITOR will show the status if the execution (QUEUED)

SQL Plan Management (SPM)

- SPM allows a conservative SQL execution plan management
- If SQL plan baselines are present for a query, the optimizer will have to use one of them
- If it seems that there is a better plan, the optimizer will have to validate it before it can use it
- By default CAPTURE SQL PLAN is off, but USE SQL PLANS is on

Back to the Story

- It took me a lot of head scratching but not too much time to find the reason
- The problem in this case was Cardinality Feedback
- This feature in conjunction with an optimizer bug, caused the optimizer to flip between 2 plans, one good and one bad, in both the cardinality was wrong
- The solution in this case was to fix the statistics so there won't be cardinality issues

Summary

- As the optimizer is getting more dynamic, the DBA job is changing from tuning a query to understanding what the hell is going on
- Today, more than ever, we need to identify the actual plan that was used
- **We need to be very aware of features changing the optimizer behavior during and after the execution**

Q&A

Liron Amitzi

liron@brillix.com

@amitzil

https://amitzil.wordpress.com