

Oracle Query Processing and Optimization

Håkan Jakobsson

Outline

- **Some basic topics**
 - Query transformations
 - Search space
 - Uncertainty
- **Examples of “advanced” issues**
 - Plan stability
 - Cost model
 - Optimization criteria
 - SQL Tune

Query Transformations

- **Query transformations convert SQL statements into semantically equivalent statements that can result in more efficient access paths**
 - **Examples: subquery-to-join transformations, MV rewrite**
- **Query optimization is a mixture of techniques at different levels**
 - **Query transformations easier early**
 - **Access path selection easier later**
 - **But transformations should be cost based and access path selection is necessary to generate cost estimates**
- **Oracle includes query transformations as part of the cost-based search space**
 - **Mainly software-engineering challenges**

Search Space

- **Large space of possible access paths, join methods, and orderings for large queries**
- **Global effects of local decisions**
 - **Choice of join method in the middle of the join order may affect whether you can eliminate an ORDER BY sort**
 - **Hence, just picking the locally least expensive join method may not give the lowest global cost**
- **Bushy join trees**
 - **Bushy trees are sometimes optimal, but considering them increases the search space and complexity of the optimizer**
- **Substantial research results in this area**

Search Space in Oracle

- **Left-deep trees unless query is inherently bushy or plan has hash joins**
- **Exhaustive search of join orderings for small joins**
- **Initial ordering heuristics and cut-off based on best plan so far**
 - **Does a very good job of keeping search space problems in check**
- **Avoid considering Cartesian products for large joins**
- **Separate passes for global effects like row ordering for ORDER BY**
- **Includes query transformations and MV rewrites**

Uncertainty about Cardinalities

- Can lead to bad join orders, bad join methods, and bad access paths
- Problem of computing accurate estimates for intermediate result sets
- Information about tables based on statistics
 - Correlation between columns unknown
`WHERE TITLE = 'MANAGER' AND SAL < 40000`
 - Hard to know properties of join results for multiple joins
 - Difficult predicates `WHERE ENAME LIKE '%SMITH%'`
 - Bind variables `WHERE SAL < :1`
 - Missing statistics

Relevant Oracle Features

- **Dynamic sampling for correlation, difficult predicates, and missing statistics**
- **Multicolumn statistics for correlation**
- **Bind peeking for bind variables**
- **SQL Tune (more later)**
- **Automatic gathering of statistics**
 - **Detection of stale statistics**
 - **Definition of stale is tricky**
 - **Efficient techniques for actual gathering including synopses**

Motivating Example: Oracle E-Business Suite

Object Type	Count:
=====	
MATERIALIZED VIEW	402
TYPE	584
JAVA CLASS	891
TABLE	21,980
INDEX	40,078
TRIGGER	3,343
VIEW	28,281
PACKAGE BODY	40,950
PACKAGE	41,939
Queries	515,000

Table references per query: AVG = 4 MAX = 232

Plan Stability

- **Problem of improving the optimizer without risking that some queries deteriorate**
- **Customers don't want something that works to be broken in a new release**
- **One query that deteriorates may outweigh improvements in all other queries**
- **Solutions to make database upgrades less risky:**
 - **Store plans – SQL Plan Management in Oracle**
 - **Versioning of optimizer behavior**

Cost Model

- **Affected by evolution of computer architecture**
 - Originally focused on disk I/O
 - Large memory sizes makes for more in-memory processing
 - L-2 cache hit misses important for performance
 - Emergence of chips with large numbers of cores
 - Solid-state drives
 - Exadata – moving computation to storage
- **Hardware evolution affects the adequacy of basic query processing algorithms**
- **Affects the optimizer's cost model for both existing and new query processing algorithms**
 - Painful to change model due to plan stability issues

Optimization Criteria

- **What are we optimizing for?**
 - **Throughput on the system – use plans with the least resource consumption**
 - **Response time – use parallelism as much as possible**
 - **First batch of rows – avoid blocking operations**
 - **Avoiding catastrophic optimizer decisions – favor plans that avoid bad worst-case scenarios (No specific mode in Oracle but several optimizer heuristics)**
- **Determining the right degree of parallelism is difficult**
 - **Adaptive degree of parallelism vs. queuing**
 - **Also, adaptive memory management**
- **Not just an optimizer issue – need a model for queries with different response time priorities**
 - **Resource Management in Oracle**

SQL Tune

- **“Super-optimization” for “high-load” SQL**
 - **Go after statements that use up the most resources**
- **Intended for recurrent workloads and based on the notion of correcting optimizer cardinalities**
- **Spend time finding actual cardinalities using dynamic sampling and partial evaluation of queries**
- **Store correction factors for the optimizer to use when the query is optimized in the future**
- **Allows queries to be tuned without changing the query text**
- **Part of framework that includes advice about index creation, statistics gathering, query restructuring**