Lab 6: Oracle's Query Processor

Due date: Monday, March 16, at the final exam.

Lab Assignment

The assignment is to be performed in pairs. Each pair shall submit one report. Pair programming rules (you work on the assignment ONLY when your partner is next to you) will not be enforced, **however** I strongly recommend that each pair spends significant time working together.

The assignment is designed to get you to investigate the operation of Oracle's query processor.

The Tasks

The two main goals of the assignment is to allow you to investigate the operation of Oracle's query processor and obtain documentary evidence that query optimization matters.

In particular, you should investigate the following issues:

1. **Does query optimization matter?** We talk in class about the importance of proper optimization of queries. But is it really true that query optimization can have **drastic** effect on the query execution time? Find supporting (or disproving) evidence to the importance of query optimization in Oracle.

2. **How does Oracle's optimizer work?** Information about Oracle's query optimization process can be found in Oracle's documentation (appropriate files are linked from the course we page), or it can be reverse-engineered by running a gamut of queries and checking the query plans used by Oracle to execute them. Find out as much as you can about the choices Oracle makes when optimizing queries for different optimization goals. We are mostly interested in the operation

of Oracle's Cost-Based Optimizer (CBO), although, if you want to, you can also investigate the Rule-Based Optimizer (RBO).

Note: Regardless of how you gained the initial knowledge of the optimizer (reverse-engineering, or documentation), your conclusions about the work of the optimizer **must be supported by evidence**. If you choose to attempt reverse-engineering, your evidence will be obtained in the process of *exploratory analysis*: i.e., you will need to run a number of queries on one or more databases to determine how optimizer works and draw conclusions from it. If you read documentation, you still must perform *confirmatory analysis* of the material by ... run a number of queries on one or more databases to see if the results match with those described in documentation.

3. **Investigate efficiency of Join operations.** Oracle uses three main implementations for joins: nested loops, merge join (a.k.a., sort-based join) and `hashed join`. Find out how these operations perform relative to each other. What are the situations in which each join is the best performance-wise?

4. **Investigate the effect of B+-tree indexes on query processing.** Oracle automatically creates and maintains indexes on all primary keys and `UNIQUE` constraints. Additionally, users can create indexes on any column[1] of any table. Investigate the effect of indexes on:

   (a) Oracle's default query optimizer behavior.

   (b) Access path (data retrieval/select) query performance (using indexes vs. using full scans).

Hints

You are asked to investigate behavior of an existing software system (Oracle), report on it, and provide documentary evidence supporting your findings. Generally speaking, the suggested approach to investigating each of the questions is:

- Design an experimental setup. All activity in Oracle happens around databases and querying them. Therefore, you will need to come up with a database or databases (or simply collections of tables) that you will use in your investigations.

  Feel free to use any of the datasets from CSC 365 (either those available from my CSC 365 web page) or those you used in CSC 365 taken with a different instructor. Also feel free to create your own tables, and populate them with as many tuples as you see fit.

  Note: To start observing tangible differences in running time of queries you may need to have relational tables with large numbers of records.

[1] For simplicity, you can limit your investigations to indexes on individual columns.

(To really test Oracle, the relational tables must approach the available buffer size, to force Oracle to use multipass versions of algorithms).

- Know your assets. You have at your fingertips the abilities to:
  - Create, populate, modify relational tables.
  - Create, delete indexes.
  - Collect index and table statistics as necessary.
  - Set the overall optimizer approach and goal.
  - Write SQL queries and PL/SQL code (functions, stored procedures).
  - Determine the amount of time it took to execute each SQL command.
  - Guide query execution by including query hints in the SQL statements.
  - View query plans generated by Oracle's query optimizer.
- Design experiments. Ask yourself a question, formulate a hypothesis (on how the question will be answered), determine what needs to happen to obtain the answer to the question. Make appropriate preparations.
- Conduct experiments and collect data. Determine what you are measuring, what your independent and dependent variables are (dependent variable is, typically, the overall performance of an SQL command, independent variables are all things that might influence the dependent variable).
- Analyze data. Determine if the data supports your initial hypothesis or invalidates it.
- Report results. See below.

Submission

Deliverables. You will have two types of deliverables: a written report and supplemental materials.

Written Report. Written report shall serve as a detailed record of all your activities related to this assignment. It should discuss what hypotheses you were testing, how you were testing them, what results you have obtained and what you have learned from the exercise.

A **hard-copy** of the written report shall be submitted to the instructor at the beginning of the final exam on Monday, March 16. An **electronic version** of the report shall be posted by each team on the course wiki.

Supplemental materials. These include any SQL or PL/SQL code (queries, CREATE TABLE statements, stored procedures, etc.) and any other code you used to create test databases and run queries against them. Also, any raw data files you used, that are not in the 365 repository should be included.

Submit the archived collection of supplemental materials via handin:

```
$ handin dekhtyar lab06 lab06-<teamName>-materials.zip
```

Also, if space permits, upload your supplemental materials to the course Wiki.