CSE 462 Homework #6 (Optional): Indexing

Name: ____________________ Date: April 26, 2011

***** Submit electronically on May 06, 2011 no later than 11:59pm. *****

This problem set is worth 250 points. You **must** type your answers.

1. (50pts) Read sections 8.3 and 8.4 of the textbook and skim through the following sections of the PostgreSQL documentation for an overview of some features supported by this DBMS. Then, provide short answers for the given questions.

```
http://www.postgresql.org/docs/current/static/indexes.html
http://www.postgresql.org/docs/current/static/maintenance.html
http://www.postgresql.org/docs/current/static/sql-analyze.html
http://www.postgresql.org/docs/current/static/using-explain.html
http://www.postgresql.org/docs/current/static/planner-stats.html
http://www.postgresql.org/docs/current/static/planner-stats-details.html
```

a) What is an index?

b) What is a query plan?

c) What are database statistics (or planner statistics)?

d) What is the relationship between indexes, database statistics, and query plans?

e) In PostgreSQL, what is the difference between the commands `EXPLAIN` and `EXPLAIN ANALYZE`?

2. (100pts) For each statement below, say whether it is `true` or `false`. Then, justify your answer with either an example or a short explanation.

a) Indexes may be created on one or more attributes of a table.

b) Multiple indexes may be created for a given table.

c) In a read-only database, indexes have no maintenance cost.

d) Every time a tuple is update in a table, all its indexes must be updated.

e) It is usually a good idea to have an index for the attributes of a primary key.

f) It is usually a bad idea to have an index for the attributes of a foreign key.

g) A sequential scan is some times preferrable to an index scan.

h) An index scan is some times preferrable to a sequential scan.

i) Database statistics and query plan selection are not related in any way.

j) If the statistics of a database are not up-to-date, the query planner may select indexes inadequatly.

3. (100pts) Using the script provided in the materials page as a reference, follow the step-by-step instructions below. For each item, include the commands you execute as well as the answers to the questions.

a) Create relations $R(A, B, C)$ and $S(D, E, C)$, where A, B, D, and E are `INT` and both C fields are `VARCHAR(80)`. Typical queries executed on this database are:

```
1) SELECT * FROM S WHERE E = K1;
2) SELECT * FROM S WHERE E BETWEEN K1 AND K2;
3) SELECT * FROM R WHERE A = K1 AND B < K2;
4) SELECT * FROM R WHERE A > K1 AND B < K2;
5) SELECT * FROM R WHERE A < K1 AND A > K2;
6) SELECT * FROM R NATURAL JOIN S WHERE A = K1 AND D < K2;
7) SELECT * FROM R NATURAL JOIN S WHERE A > K1 AND D < K2;
8) SELECT * FROM R NATURAL JOIN S WHERE A < K1 AND D = K2;
9) SELECT * FROM R NATURAL JOIN S WHERE (A < K1 OR A > K2) AND D < K3;
```

b) Add at 4 million records to R and 1 million to S. Every attribute must be assigned a random value– a random integer in the range of 1 to 4 million, or a string of random characters with a random length in the range of 1 to 80 characters.

c) Delete from R all tuples that have duplicate values on their A, B attributes.

d) Delete from S all tuples that have duplicate values on its D attribute.

e) Obviously, not all queries can be optimized for all possible values of their parameters. Is there any query above which always benefits from an index? Which one(s)? What index would you create to improve each such query? Provide evidence to support your claim: run `EXPLAIN` before and after creating the index(es), show the results, and provide a quick interpretation.

f) Are there queries, do you suppose, that *may benefit* from the creation of a primary key on A, B in R (recall that a `UNIQUE INDEX` is created by PostgreSQL for the primary key)? For such queries, show that the effective use of the primary key index depends on the values of the Ks in the queries. It suffices to run `EXPLAIN` on the queries with adequate values for the Ks, before and after creating the primary key, and then provide a quick explanation of the results you observe.

g) Are there queries, do you suppose, that *may benefit* from the creation of a primary key on D in S? For such queries, show that the effective use of the primary key index depends on the values of the Ks in the queries. It suffices to run `EXPLAIN` on the queries with adequate values for the Ks, before and after creating the primary key, and then provide a quick explanation of the results you observe.

h) Are there queries, do you suppose, that could further benefit from the creation of additional indexes? For each such index, indicate which queries it could benefit. Then, show that the effective use of the index depends on the values of the Ks in the queries. It suffices to run `EXPLAIN` on the queries with adequate values for the Ks, before and after creating the index, and then provide a quick explanation of the results you observe.