Lab 8 : CS 387 Jan-Apr 2024

Examining and understanding query plans, query optimization, in postgres

Part-A Preparing the database

1. Create a database called univ8
2. Load the university schema into your database – DDL.sql
3. In case you had loaded the schema + data, and want to start from scratch, you need to drop tables first – DDL-drop.sql
4. Use the psql command line to load file largeRelationsInsertFile.sql . You will see "INSERT 0 1" being printed for every insert. It will take some time but will terminate successfully. You may see an unchanging screen even though the system is continuously printing the above line. Hit enter to see if prints are ongoing.

Part-B Exploring various query plans and query optimizations

In each of the queries below, you can create/delete indices on appropriate relation attributes, as necessary.

1. **query1.sql** : Create a query where PostgreSQL uses bitmap index scan on relation takes. Explain why PostgreSQL may have chosen this plan. Reference: https://www.geeksforgeeks.org/bitmap-indexing-in-dbms/ . Also note that PostgreSQL does not have a specific command for bitmap index creation.
2. **query2.sql** : Create a selection query with an AND of two predicates, whose chosen plan uses an index scan on one of the predicates.
3. **query 3.sql** : Create a selection query with an OR of two predicates, whose chosen plan uses an index scan on one or both of the predicates.
4. **query4.sql** : Create a query where PostgreSQL chooses a (plain) index nested loops join. NOTE: PostgreSQL uses nested loops join even for indexed nested loops join. The nested loops operator has 2 children. The first child is the outer input, and it may have an index scan or anything else, that is irrelevant. The second child must have an index scan or bitmap index scan, using an attribute from the first child.

5. Create an index as below, and see the time taken to create the index: create index i1 on takes(id, semester, year);

6. Similarly see how long it takes to drop the above index using: drop index i1;

7. **times.csv** : Create a table takes2 which has the same schema as takes. Insert the first N entries of takes into takes2. Now create index i2 on takes2, similar to index i1 on takes. Measure the time taken for index i2 creation and deletion, as a function of N. Explain your observation. Your CSV file should have 3 columns: N, createTime, dropTime . Have say 5-6 rows for different values of N. Last row of CSV is just a plain text sentence or two with your explanation.