Experiment's description:

This experiment is similar to previous one, but here indexes are not dropped and added again.

In the java application a scenario is simulated, in which random input user data is generated and buffered, and then bulk-loaded into DB. In the meanwhile, there is a user agent that requests some random data. PostgreSQL is used for database. Application uses only one database, containing only one table (Database index on ID column (integer value-Primary key)).

The Java application consists of following threads:

-CreateTextFileThread:

In this thread, a text file is created that contains 10,000 rows of sample user data. Each row has 10 columns, some varchar (constant value in all rows), some numeric (random value). This is used as a simulation for having incoming user input data and buffering it into a file. This way, we can later use "COPY" operation for bulk-loading which is faster than series of "INSERT" operations for all rows. (See chapter 14. Performance Tips in PostgreSQL's documentation for more information.)

-InsertRowsThread:

Here, the created text file is copied into database. As in PostgreSQL's documentation was mentioned that creating index on pre-existing data is faster than updating index incrementally as each row is loaded. In this experiment, we did not drop indexes to compare results with previous experiment in which we did drop and add again the indexes. So, in general this thread does the following steps:

1. loads the entire text file (the buffer) into table using "COPY" command
2. runs "ANALYZE" command

Running "ANALYZE" after bulk loading, ensures that the planner has up-to-date statistics about the table, which results in better decision making during query planning. (See chapter 14. Performance Tips in PostgreSQL's documentation for more information.)

-The RequestRow Threads:

These threads just send random queries based on table's numeric columns and then go to sleep for 1 or 2 seconds after getting the results of the query.

The CreateTextFile Thread and InsertRows Thread take turns, so that data is loaded after new text file is created. Queries are sent simultaneously. Each time a thread operates and goes to sleep is called a cycle in excel files.

Other classes are there just to run threads or to save the results.

The excel files show the results of running the program for about 2 hours.

1-Compare copy time: This file compares the duration of “COPY” operation in cases which first, indexes are removed before copy and added again after copy and second, without index removal. The results show that removing and adding index is costly and the copy operation is faster without index removal. (The tip in documentation is probably referring to some other situation.)

2-RequestSalarysleep1Excel: This file shows the number of results and duration of queries sent by a sample request agent. Here the condition of select query is correctly bounded, and the results show that the duration of response to a query is almost not growing as the database grows.

In all excel files duration is reported in millisecond.