Project Proposal

Dynamic indexer for PostgreSQL

Team

- Vamsi Krishna Reddy Satti 160050064
- Vighnesh Reddy Konda 160050090
- Niranjan Vaddi 160050099
- Sai Praneeth Reddy Sunkesula 160050100

Abstract

Indexing the tables in database systems can increase efficiency based on the types of operations done on the tables. PostgreSQL does not enable indexing on non primary key columns by default. However indexing can be useful when there are frequent reads from similar columns on the table.

Project Goals

We will implement an application for PostgreSQL to detect the columns that needs to be indexed based on the frequency of scans done by postgreSQL and create indexes for those columns. By this the database adapts and access patterns improve as the workload changes without any human intervention in cases where the reads are more frequent than writes. We plan to implement this by extracting the queries on the database from the logfile.

Implementation details

We intend to achieve this in two phases:

1. Collect Queries
2. Generate Indexes

1. Collect Queries

a. Stream the postgres log file directly to the application, filter out queries on system tables, and parse out the queries and their duration.

b. Queries with same parse trees but with different values are grouped together (fingerprinting) with the help of libpg_query library[1].

2. **Generate Indexes**
    a. Get the initial cost of queries and columns scanned for execution of these queries, by using Postgres query planner.
    b. Create hypothetical indexes[1] on columns that aren't already indexed by using HypoPG[2] extension for PostgreSQL.
    c. Get costs again and see if there was any decrease in cost by using the above hypothetical indexes.
    d. Index the columns based on whether any improvements in costs are expected by using the additional indices.

Testing

Load a large database and design various patterns of table accesses and compare the performance of postgres during heavy load on database with the indexer on and off. We shall evaluate the improvements of automated indexing using scripts for database accesses.

References

[1] libpg_query library
[2] HypoPG Extension

[1] In HypoPG, a hypothetical or virtual index is an index that doesn't really exists, and thus doesn't cost CPU, disk or any resource to create. They're useful to know if specific indexes can increase performance for problematic queries, since you can know if PostgreSQL will use these indexes or not without having to spend resources to create them.