IMPLEMENTATION OF LSM TREES

REPORT 2019-2020

(CS 631)

TEAM MEMBER:

RISHI RAJ SINGH (193050036)

SANDEEP PANWAR (193050019)

1. MOTIVATION AND GOAL OF PROJECT :

In computer science, the **log-structured merge-tree** (or LSM tree) is a data structure with performance characteristics that make it attractive for providing indexed access to files with high insert volume, such as transactional log data.

Goal of our project was to implement LSM trees, using existing architecture of Postgres. We started with implementing LSM trees in Postgres but later switched to implementing our own B+ tree and using the functions of B+ tree as basic structure and implementing LSM tree over it.

2. CODE DESCRIPTION :

We have 3 files on which B+ Tree and LSM tree is implemented.

Bplustree.h : This file contains functions important for the working of B+ Tree.

- The class of a node is defined with attributes like if it is a leaf node or not, size of node etc.
- It has code for B+ tree class. It also includes all the functions like insertion, splitting etc.

Bplustree.cpp : This file is just a way to use a single B+ tree. It consist of a menu that can be used to insert, display, and delete nodes.

LSM.cpp : This contains the most important part of our project. We are asking for the size of a node in B+ tree and then the max size of the primary B+ tree. Then the menu is straightforward and can be used to insert data, display all tables and for exit. A linked list of all the root of B+ tree is created. When the first B+ Tree is full it is pushed and a new free node is created. When it is also full, first two B+ Tree are merged to form a B+ tree which is pushed and the primary is free again.

- There are two insert functions one for inserting into linked list and another for inserting into B+ tree.
- When a tree is full it is merged and appropriate merge function is present.
- A display function is present that will traverse the linked list and display all the trees.

3. DIFFERENT METHODS WE TRIED :

METHOD 1: NBTREE

Going through the code of Postgres we found nbtree related files. We went through them and we understood the working of nbtree insert and nbtree.h. When we tried to implement LSM tree using the functions we found no way to check our progress.

METHOD 2: STARTING FROM THE START

We started from Postgres.h file tried uncommenting pretty printing to understand working of query processing. But to the level we went we didn't find indexing related stuff.

METHOD 3: ONLINE HELP

We tried to make Postgres realize that our index exist. So we tried to Edit pg_am & pg_am_d, but that didn't lead to presumably result.

This websites were used to understand the working of catalogs.

Finally we decided to start from scratch.

4. BUG-FIXES :

There were two major bugs which took a lot of time to fix:

- After merge when we were trying to insert again, segmentation fault would show up.
- Trying to make sure that the 2nd B+tree is of twice the size of first and so on.

5.3 REFERENCES :

- https://medium.com/postgres-professional/indexes-in-postgresql-2-ad687857989
- http://paperhub.s3.amazonaws.com/18e91eb4db2114a06ea614f0384f2784.pdf
- https://www.cs.umb.edu/~poneil/lsmtree.pdf
- https://www.postgresql.org/docs/7.2/pg-system-catalogs.html