**B+ TREE IMPLEMENTATION**

Team 18 – Bivash Thakuri

– Utsav Acharya

CSE 4331/5331

Prof. Sharma Chakravarthy

University of Texas, Arlington

INTRODUCTION

B+ tree is a multilevel indexing data structure employed widely by all the database systems. In a B+ tree, all the data pointers are stored in the leaf nodes whereas the index nodes contain the pointers to the next level index nodes or leaf nodes. All the leaf nodes are connected by doubly linked list which provides faster access mechanism.

PROJECT DESCRIPTION:

In this project, Minibase, a database management system developed by Raghu Ramakrishna, the author of the book - Database Management System, is used as a code base to implement B+ tree data structure. Three functions – insert(), \_insert(), and naivedelete() have to be implemented in the BTreeFile.java. The B+ tree stored data entries in the leaf pages as alternative 2 form.

This project comprises of more documentation reading than writing code. After you understand the documentation, go through the given packages and classes, and understand the functionalities of each classes, implementing a B+ tree is not so hard. However, without the knowledge of the code base, you can’t even write a single line of code.

OVERALL STATUS

Both the insert and the delete algorithm are fully implemented.

FILE DESCRIPTIONS

Neither any files nor any additional test cases have been added

DIVISION OF LABOR

The work-load was divided into two parts:

1. Implementation of insert() and \_insert() function – Bivash Thakuri

Reading the documentation took about 1 hour / day for about 2 weeks. Writing and debugging code took around 2 hour / day for about 2 weeks. So, in overall, it took 14 hours reading the codebase and 28 hours writing the code.

1. Implementation of naivedelete() function and report – Utsav Acharya

Reading the documentation took about 1 hour / day for about 2 weeks. Writing and debugging code took around 2 hour / day for about 1 week. And, writing the report took around 3 hours. So, in overall, it took 14 hours reading the codebase and 14 hours writing the code, and 3 hours writing the report.

LOGICAL ERRORS