Project 1 : B+ Tree Implementation

Abstract :

This project is about implementing the insert and naïve delete methods in a B+ tree so that the code can be able to handle the insertion and deletion of data records which might further cause splitting and re-distribution of nodes/pages.

Overall Status :

We have gone through the pdf file shared in the canvas. It gave us an insight of what the project is all about and in understanding algorithms of the methods which are to be implemented. The syntax and description of the methods shared by Abhishek has helped us in writing the coding part by carefully analyzing the algorithms of each method.

a) Implementation of insert method:

We started with the insert method which has two parameters – a key and a record ID. At first, we checked whether if there is any header page by getting its page ID. If there is no header page, we created a leaf page, then get the page ID, set the pointers, inserted the record, unpinned the page and updated the header.

Further insertions will be done through _insert method which has three parameters – a key, record ID and a page ID. We checked whether page type is leaf or index. If it's a leaf page, we checked for the available space and inserted the data and unpinned the page. If the leaf page has no space, we split the data into the new leaf page. Here, splitting is done by dividing the records into half. For loop logic is used to transfer the records into the new leaf page and then the record in the old page will be deleted. If it's a index page, we checked if split has occurred or not. If split has occurred, we checked for the available space in the index page and inserted the data record into the page.

Getting the page ID – setting the pointers – transferring the records into new page – deleting records from old page – unpinning the pages. This process is same whether it's a leaf page or a index page.

Note : Above, We have just mentioned the outline of the implementation of our insert method. Detailed steps are mentioned in the form of comments(in btree.java file).

b) Implementation of Naïve Delete Method:

Naïve Delete method takes two parameters – a key and a record ID. We traversed through the pages till the target key is found. We compared the two keys and deleted the key through delEntry method. We used getNextPage function to implement multiple deletions. In this implementation, merging and re-distribution of nodes are not taken into consideration.

File Descriptions:

We haven't created any new files. Code is implemented using the functions that are provided to us.

Division of Labor:

We sat together in library and understood the concepts of B+ tree. We have gone through the slides and the documents that are provided to us and cleared each other doubts through mutual help. Once when we made sure that we had the basic knowledge in the B+ trees, We started working on the coding part.

Mahesh - _insert(leaf) and Naïve Delete.

Harsha - insert and _insert(index).

Error Handling:

a) Writing the for loop logic to transfer the records was a bit challenging one, we were getting errors in this part. We were using the getNext method and later upon scrutiny, we figured it out that it needs to be getCurrent.

b) While running the code, we were getting an index insert record exception. We found that it's because we have not used get keyType function, instead we explicitly mentioned the node type as index.

c) Once when we finished the code in eclipse, we connected to the omega server and when we try to execute in the omega remote server, it is throwing many errors. Then we spent a lot of time on understanding what went wrong. We came to know that eclipse installs it's own libraries when we execute the code in it's environment. So we copied our code into notepad++ and then we tried executing on the omega server. It compiled successfully and generated the desired output.