Name: Mir Basheer Ali ID: 1001400462

Name: Deeksha Bhat ID: 1001174031

PROJECT REPORT – B+ Tree Implementation

CSE 5331-Fall 2016

DBMS Models and implementation (Section 001)

Instructor: Sharma Chakravarthy

TA: Abhishek Santra

Overall Status

The purpose of the project was to understand how B+ trees are implemented as an index data structure and the organization of file using the MINIBASE. This needed us to implement the following methods

- insert()
- _insert()
- NaïveDelete()

- Insert()
  1. Is the entry point for the creation of tree which validates the input before the data entry is pushed into the index, the validations include; if the key RID pair entered is of the correct type and length.
  2. Check if the root of the header page is pointing to an invalid page, which signifies that the first leaf page needs to be created and is done by invoking the createFirstLeafPageAndInsertValue method.
  3. Call the _insert() recursively, checking if the value returned is not null. If the value returned is null then there has been no split in the index pages else, there is a split occurred and new index page needs to be created which is done by the invocation of the createFirstIndexPage method.
- _insert()
  1. This handles the insertion and the splits of the pages at multiple level. This method involves two cases
     - If the current page is an index page

OR

- If the current page is a leaf page

2. If current page is an index page
    - If the page is the index page unpin the page and invoke the _insert() method and check if the value is null which signifies that there has been no split in the lower level.
    - If the value return is non-null then check if the current entry can be inserted in the currentIndexPage by checking if the current index page has enough space for the new key.
    - If space available in the current index page is not sufficient create new index copy values from the current index page to the new index page, split the entries equally, compare the current entry to be inserted to the last entry in the split to know where the new key needs to be inserted.
    - Return the entry to be pushed up to the index level.
3. If the current page is a leaf page.
    - Check if the current leaf page has enough space for the new data entry to be pushed. If available, insert the new data entry and return null signifying no split has occurred.
    - If space available in the current leaf page is not sufficient create new leaf, copy values from the current leaf page to the new leaf page, split the entries equally, compare the current entry to be inserted to the last entry in the split to know where the new key needs to be inserted.
    - Return the new entry to be copied up.

- NaiveDelete()
    1. This method deletes the first entry obtained for the key passed. The method handles the delete of an entry but, it does not handle the merging and re-distribution.
    2. Invoke the findRunstart method which returns the left most leaf page consisting of the key. Since the leaf pages are doubly linked list we traverse the list to find the record to be deleted.

The above methods have been implemented and have been tested with the test cases provided. For the understanding and pseudocode we referred the text book along with the demo pdf provided.

File Description

The implementation was an extension to the project skeleton provided. We have added two more methods to the BtreeFile.java which consists of the methods to perform the actions.

Division of Labour

We worked together in the implementation of the project. We started with going through the javadocs provided to us to understand the methods we use in minibase. We used to discuss algorithms and how could that be implemented. We spent around 12 hrs each week for about 2 weeks to complete this project. We came up with logic for insert() and NaiveDelete() quite easily however _insert() method was quite challenging. Working together helped us to collaborate our ideas and implement it in our code.

Logical Error and Handling

- Error Description: Partial output of the inserted value i.e. after insertion of 100 values when the leaf page values were to be printed only the values of the first page were printed.

  Solution: Needed to split the page after the space in the current page is not enough for insertion of the key on leaf page.
- Error Description: Deep recursion leading to the Stack Over Flow Error due to wrong linking of the current and previous leaf page.

  Solution: Correctly linked previous and next page gave us expected output.
- Error Description: Insertion of same key at the beginning of the page. i.e. the value [1,1] was inserted twice in the beginning of the page all the other values were correctly inserted except this.

  Solution: return statement was missing after calling createFirstLeafPageAndInsertValue method which is used to create first leaf page lead to continuation of the program in insert method and insert the same value again. After adding return in appropriate return statement this was solved.
- Error description: UnpinPageException: Unpinning a page before pinning it.

  Solution : Pin the page before unpinning the page.