

## B+ Tree Index

```

CA\Users\cl2287\Desktop\perfect\perfect\Debug\BTree.exe
----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 1 < 1 Data , 0 indexpages >
Total data entries are : 50
Total index entries are : 0
Hight of the tree is  : 0
Average fill factors for leaf is : 0.857422
Maximum fill factors for leaf is : 0.857422      Minumum fill factors for leaf
is : 0.857422
Average fill factors for index is : 0
Maximum fill factors for index is : 0      Minumum fill factors for index is : 0
That's the end of dumping statistics.
Test 1 Passed!

----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 1 < 1 Data , 0 indexpages >
Total data entries are : 44
Total index entries are : 0
Hight of the tree is  : 0
Average fill factors for leaf is : 0.757813
Maximum fill factors for leaf is : 0.757813      Minumum fill factors for leaf
is : 0.757813
Average fill factors for index is : 0
Maximum fill factors for index is : 0      Minumum fill factors for index is : 0
That's the end of dumping statistics.
Test 2 Passed!

----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 3 < 2 Data , 1 indexpages >
Total data entries are : 31
Total index entries are : 1
Hight of the tree is  : 1
Average fill factors for leaf is : 0.526855
Maximum fill factors for leaf is : 0.542969      Minumum fill factors for leaf
is : 0.510742
Average fill factors for index is : 0.0556641
Maximum fill factors for index is : 0.0556641      Minumum fill factors for index
is : 0.0556641
That's the end of dumping statistics.
Test 3 Passed!

----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 1 < 1 Data , 0 indexpages >
Total data entries are : 25
Total index entries are : 0
Hight of the tree is  : 0
Average fill factors for leaf is : 0.833008
Maximum fill factors for leaf is : 0.833008      Minumum fill factors for leaf
is : 0.833008
Average fill factors for index is : 0
Maximum fill factors for index is : 0      Minumum fill factors for index is : 0
That's the end of dumping statistics.
Test 4 Passed!

----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 187 < 181 Data , 6 indexpages >
Total data entries are : 5000
Total index entries are : 180
Hight of the tree is  : 2
Average fill factors for leaf is : 0.512927
Maximum fill factors for leaf is : 0.994141      Minumum fill factors for leaf
is : 0.501953
Average fill factors for index is : 0.4375
Maximum fill factors for index is : 0.519531      Minumum fill factors for index
is : 0.0820313
That's the end of dumping statistics.
Test 5 Passed!

----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 73 < 72 Data , 1 indexpages >
Total data entries are : 2567
Total index entries are : 71
Hight of the tree is  : 1
Average fill factors for leaf is : 0.654053
Maximum fill factors for leaf is : 0.976563      Minumum fill factors for leaf
is : 0.519531
Average fill factors for index is : 0.998047
Maximum fill factors for index is : 0.998047      Minumum fill factors for index
is : 0.998047
That's the end of dumping statistics.
Test 6 Passed!

----- Now dumping statistics of current B+ Tree!-----
Total nodes are      : 110 < 107 Data , 3 indexpages >
Total data entries are : 3000
Total index entries are : 106
Hight of the tree is  : 2
Average fill factors for leaf is : 0.520188
Maximum fill factors for leaf is : 0.994141      Minumum fill factors for leaf
is : 0.501953
Average fill factors for index is : 0.510417
Maximum fill factors for index is : 0.984375      Minumum fill factors for index
is : 0.0410156
That's the end of dumping statistics.
Test 7 Passed!
Hit [enter] to continue...

```

Test Results for:

- Test 1: Only Insertion on a single node
- Test 2: Insertion and Deletion on a single node
- Test 3: Insertion with Leaf Split
- Test 4: Leaf Node Merge
- You don't have to pass this test case.
- Test 5: Large Workload, Insertion Only
- Test 6: Large Workload, mixed Insertion/Deletion
- Test 7: Test BTreeScan

This program implements a B+ Tree index for a database system.

Implementation:

Insertion and deletion:

Starting from a leaf page, the leaf page is split into 2 leaf pages and a root page after the first leaf page is full. Then we recursively implement insertion, going from the root node to the leaf node, and propagate a (key, value) pair back up the tree if a leaf or index node is full and split into 2. If a leaf node is split, then the pair (smallest key, new leaf page ID) is propagated up the tree. If an index node is split, the new index node's smallest key is deleted and its corresponding value (page ID) is set as the left link. The pair (smallest key (just deleted), new index page ID) is propagated up the tree. If the root node is full and split, we split the root index node into 2 index nodes, and also make a root node, whose left link points to the original index node, and the first inserted record points to the new index page. In the header page, reset the pointer to the root page .

We always split the nodes such that the smaller half is in the original node and the bigger half is in the new node. This way, the original node's left link and the upper tree level's node pointer to the original node can be left unchanged and correct.

The problems we encountered were that open scan returns the leaf page's (key, datarid), and at first we thought the returned rid is the record ID of where the record is stored in the leaf page. We took some time to plan how to code recursion of propagating up the (key,value) pair if a leaf or index node is split, and how to merge and redistribute should a page is deleted. We made our own functions to implement recursion.

- Any assumptions you made to get the code to work.

No assumption beyond the instructions in the instruction sheet.

- A description of any test cases you wrote and why they are useful. You should also include the code of these test cases in the project.

We did not write our own test cases. We used the btreeDriver test cases.

- Any known bugs. If your code fails a test case, explain the symptoms and what you think the problem might be.

We did not have any bug. All tests passed.

- A description of your performance measurements, if you elected to do the extra credit.

The statistics of tests 1-7 are included. The leaf nodes and non-root index nodes are at least 50% full, as required. From test 5 insertions, some leaf nodes are almost full (99.41%), but most are just over 50% full. From test 6 mixed insertion and deletion, the leaf average fill factor is somewhat higher (65.41%) than 50% because of node merging and redistribution from deletion.