CS564 Project 3 B+ Tree

Group Member

- Erzhen Zhang
    - CS Login: erzhen
    - NetID: ezhang25
- Yingjie Shen
    - CS Login: yingjie
    - NetID: shen92
- Xinping Liu
    - CS Login: xinping
    - NetID: xliu735

Implementation Explanation

Under following assumptions, we implemented this project.

1) Valid page number and records are nonzero.
2) No duplicate keys
3) Index are integer data type

In this project, we use level to find out which nodes are leaf nodes and which nodes are internal nodes. And in the project, we choose to store smaller key in the left subtree of a specific node, larger key in the right subtree of that specific node.

Pin and Unpin

In this case, we used recursive insertion. Firstly, we need to get a page for the current internal node, and we need to find out which node that we should visit next. And, we need to decide if the child node has split and this current node need to be split when we return from the next level. If those are not the case, the page is unpinned. If child splits and current node does not need to

split, current node would be unpinned, and the dirty bit would be set to true. If current node needs a split, new page will be allocated. Then pages of both nodes will be unpinned. Before scanning, the page will be found and unpinned before we move to the next one. However, if this page is in leaf node, then we keep this page pinned. In the scanning, we read the pinned leaf page. If we reach the end, the next page will be pinned, and the original page will be unpinned. But if we get an element with a greater key than the upper bound, the we complete the scan and the pinned page will be unpinned. And, if we called the end scan, the current pinned page will be unpinned as well.

Additional Tests

- test4()

This test creates key-value pairs from 0 to 5000 in random order to test if the output amount matches the input vector and finds keys that are not in the range of input.