B^+ Tree

Dr. Manzoor Ahmad Chachoo
Scientist D
Department of Computer Science

University of Kashmir

27-04-2020

Outline

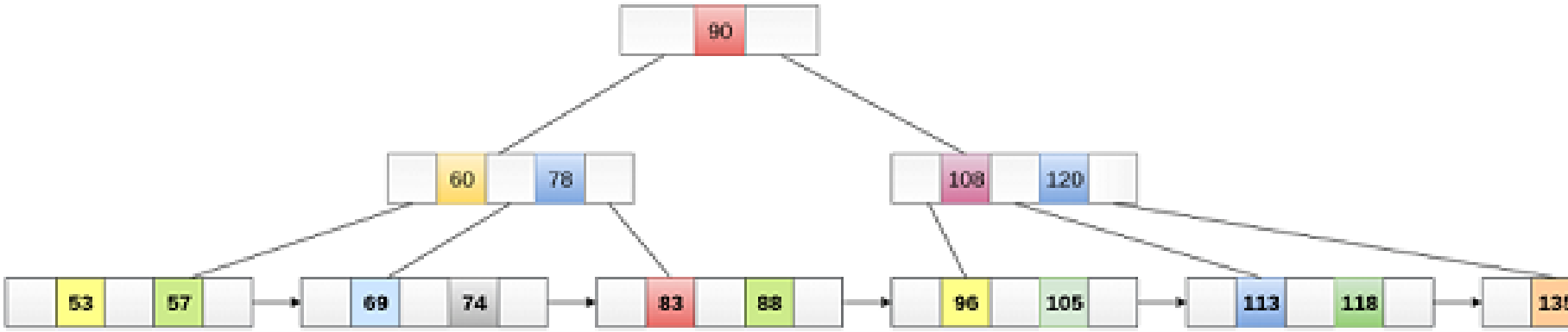
B^+ Tree

- B+ Tree is an extension of B Tree which allows efficient insertion, deletion and search operations.
- In B Tree, Keys and records both can be stored in the internal as well as leaf nodes. Whereas, in B+ tree, records (data) can only be stored on the leaf nodes while internal nodes can only store the key values.
- The leaf nodes of a B+ tree are linked together in the form of a singly linked lists to make the search queries more efficient.

B^+ Tree

- B+ Tree are used to store the large amount of data which can not be stored in the main memory.
- Due to the fact that, size of main memory is always limited, the internal nodes (keys to access records) of the B+ tree are stored in the main memory whereas, leaf nodes are stored in the secondary memory.

The internal nodes of B+ tree are often called index nodes. A B+ tree of order 3 is shown in the following figure.



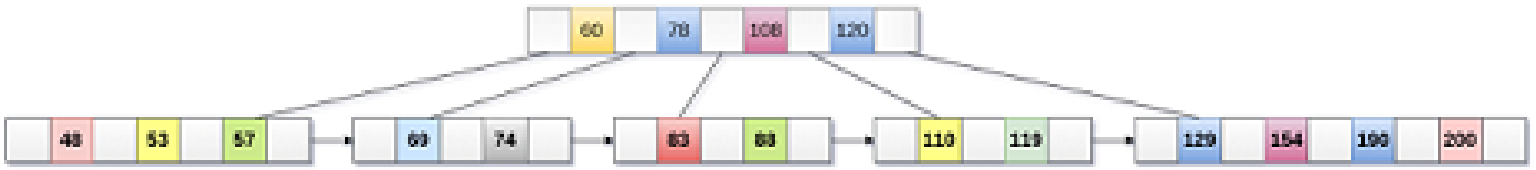
Advantages of B+ Tree

1. Records can be fetched in equal number of disk accesses.
2. Height of the tree remains balanced and less as compare to B tree.
3. We can access the data stored in a B+ tree sequentially as well as directly.
4. Keys are used for indexing.
5. Faster search queries as the data is stored only on the leaf nodes.

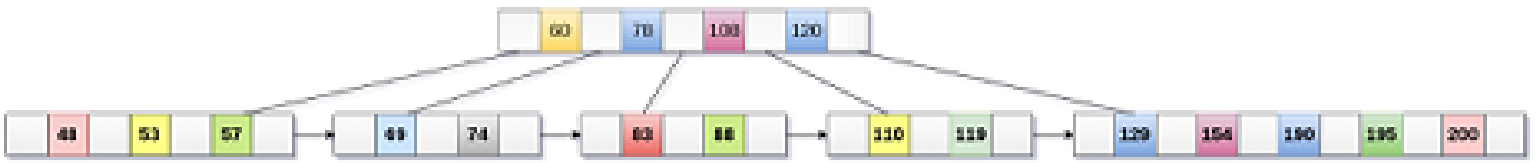
Insertion in B+ Tree

1. Insert the new node as a leaf node
2. If the leaf doesn't have required space, split the node and copy the middle node to the next index node.
3. If the index node doesn't have required space, split the node and copy the middle element to the next index page.

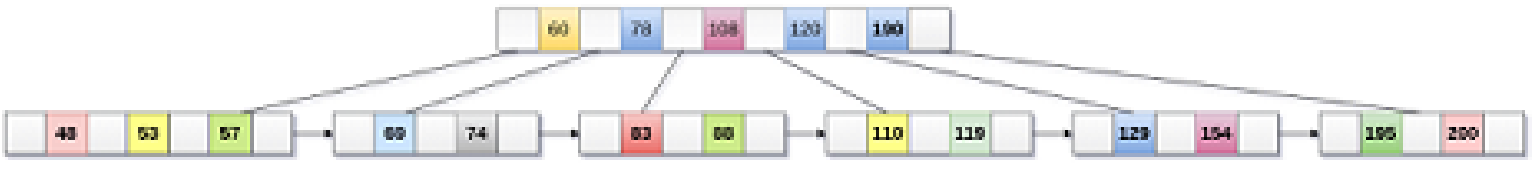
Insertion in B+ Tree

1. Example : Insert the value 195 into the B+ tree of order 5 shown in the following figure.
2. 

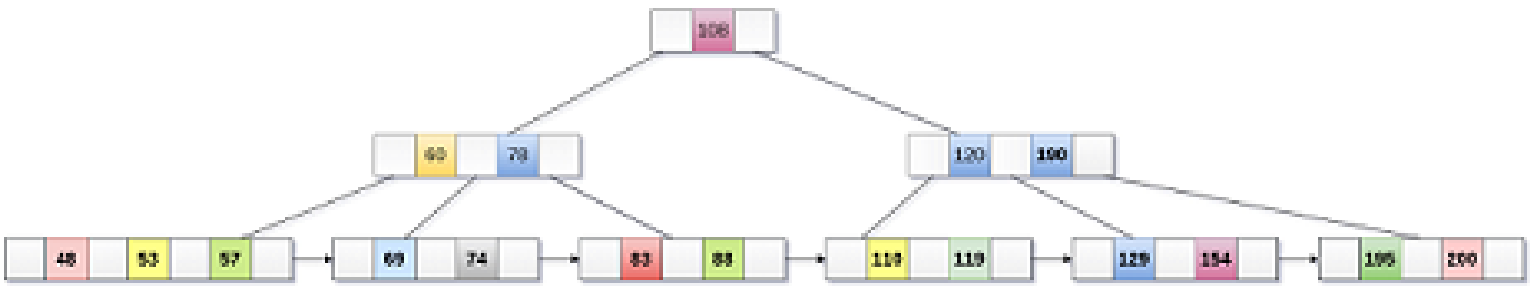
Insertion in B+ Tree

1. 195 will be inserted in the right sub-tree of 120 after 190. Insert it at the desired position.
2. 

Insertion in B+ Tree

1. The node contains greater than the maximum number of elements i.e. 4, therefore split it and place the median node up to the parent.
2. 

Insertion in B+ Tree

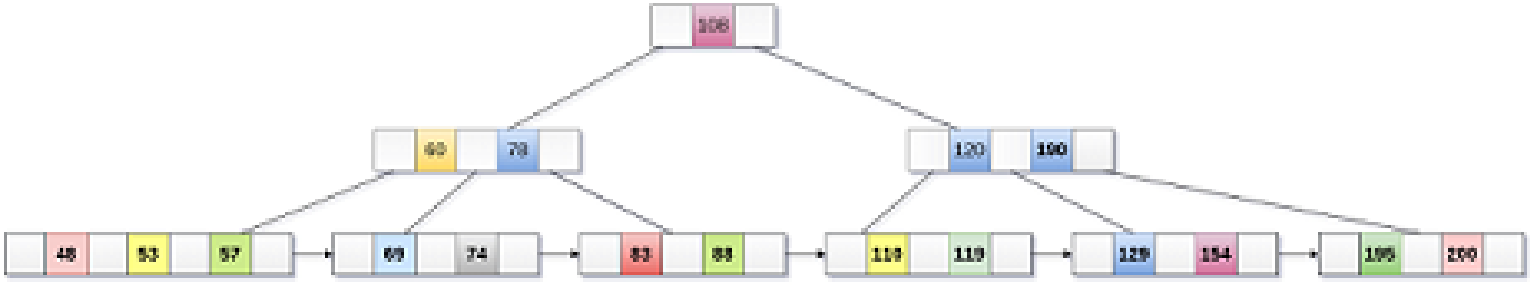
1. Now, the index node contains 6 children and 5 keys which violates the B+ tree properties, therefore we need to split it, shown as follows.
2. 

Deletion in B+ Tree

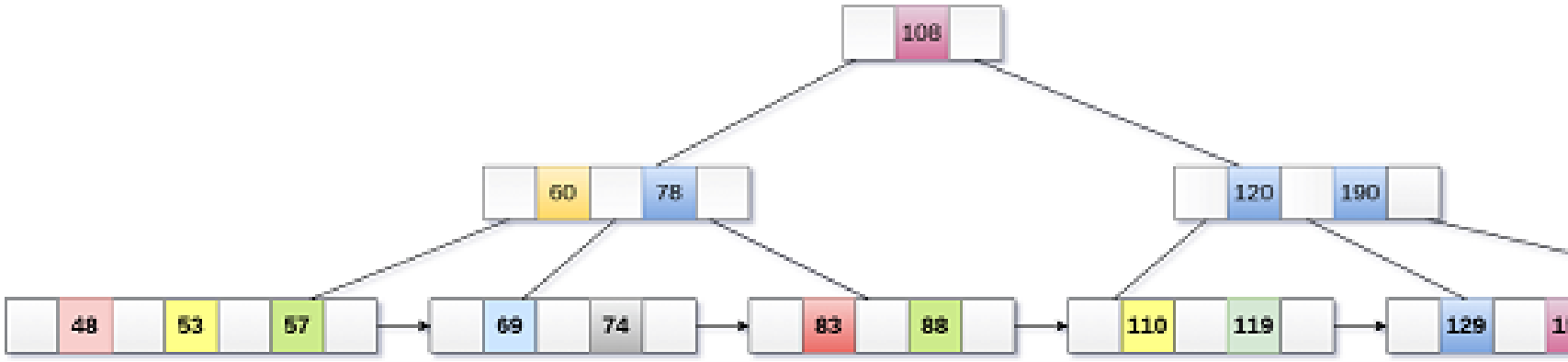
1. Delete the key and data from the leaves.
2. if the leaf node contains less than minimum number of elements, merge down the node with its sibling and delete the key in between them.
3. if the index node contains less than minimum number of elements, merge the node with the sibling and move down the key in between them. Records can be accessed only in a particular sequence..

Deletion in B+ Tree

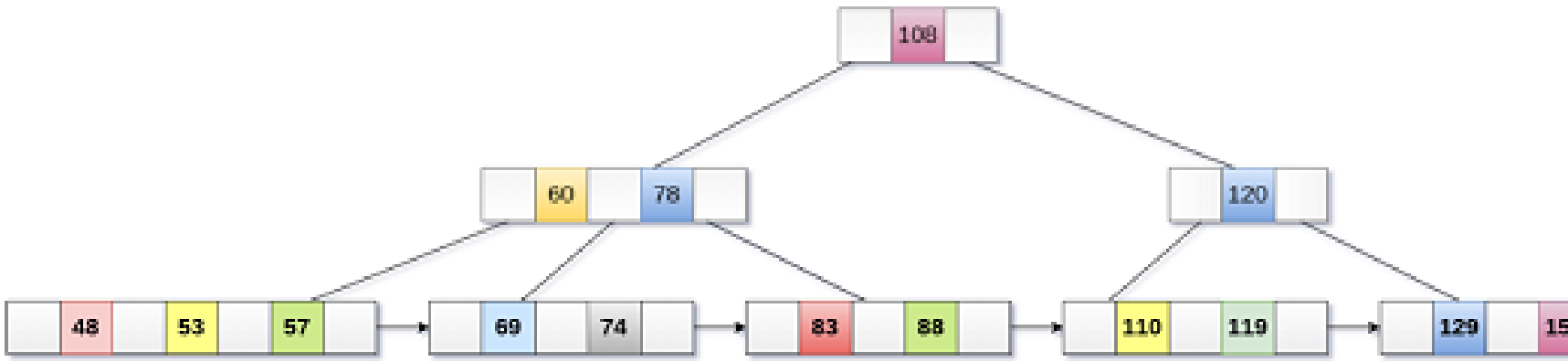
1. Delete the key 200 from the B+ Tree shown in the following figure.
2.



Deletion in B+ Tree

1. 200 is present in the right sub-tree of 190, after 195. delete it.
2. 

Deletion in B+ Tree

1. Merge the two nodes by using 195, 190, 154 and 129
2. 

Deletion in B+ Tree

1. Now, element 120 is the single element present in the node which is violating the B+ Tree properties. Therefore, we need to merge it by using 60, 78, 108 and 120.
   Now, the height of B+ tree will be decreased by 1.
2.