



Indexing: **B+ Tree**

Abdu Alawini

University of Illinois at Urbana-Champaign

CS411: Database Systems

Learning Objectives

After this lecture, you should be able to:

- Describe how to search B+ Trees
- Describe how to insert new key(s) into a B+ tree
- Describe how to delete key(s) from a B+ tree

B+ Trees

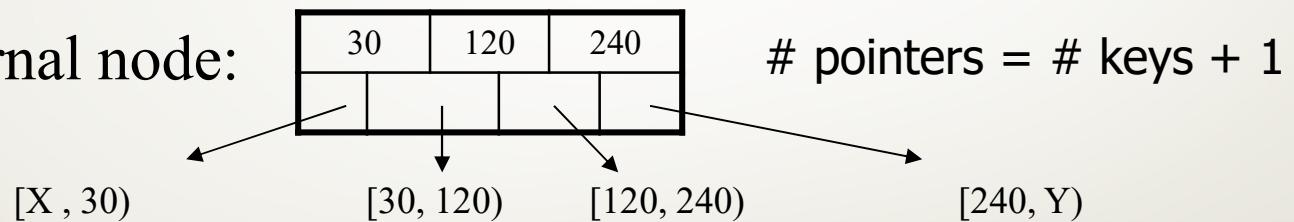
- Intuition:
 - The index can be very large.
 - Index of index?
 - Index of index of index?
 - How best to create such a multi-level index?
- B+ trees:
 - Textbook refers to B+ trees (a popular variant) as B-trees (as most people do)

Focus on the dense version:
applies to clustered and unclustered settings

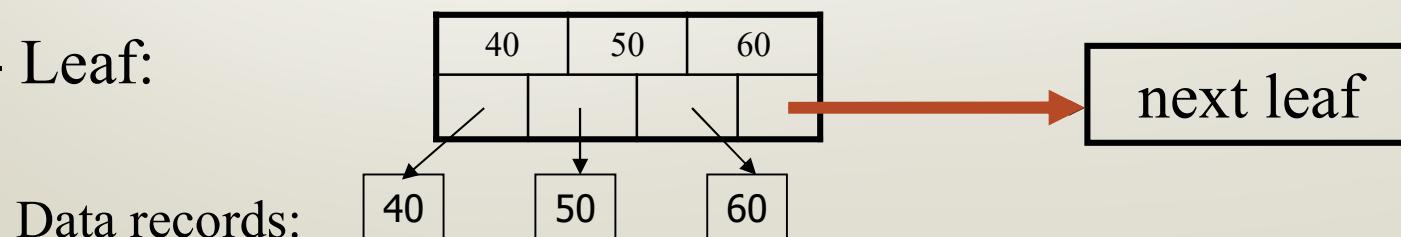
B+ Trees Basics

- B+ Trees are trees with nodes:
Nodes have keys and pointers to:
 - Other nodes [if the node is an internal node]
 - Data Records [if the node is a leaf]

- Internal node:



- Leaf:

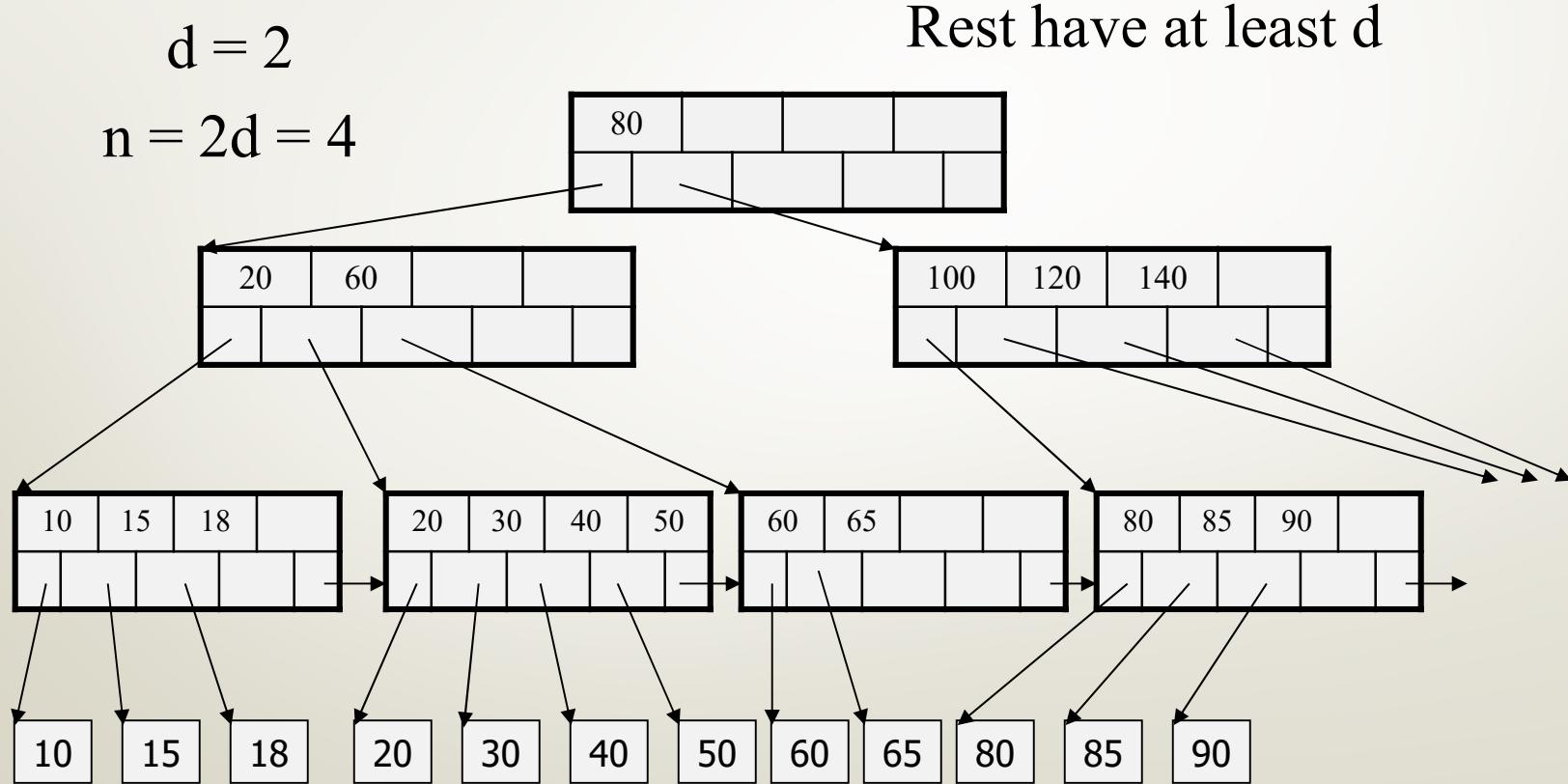


B+ Trees Basics

- Parameter d = the degree; n = max keys
- When n is even [*this is our focus for simplicity*]
 - each node has $[d, 2d]$ keys (except root); $n = 2d$
- At least half full at all times
 - d is the minimum amount it needs to be full.

B+ Tree Example

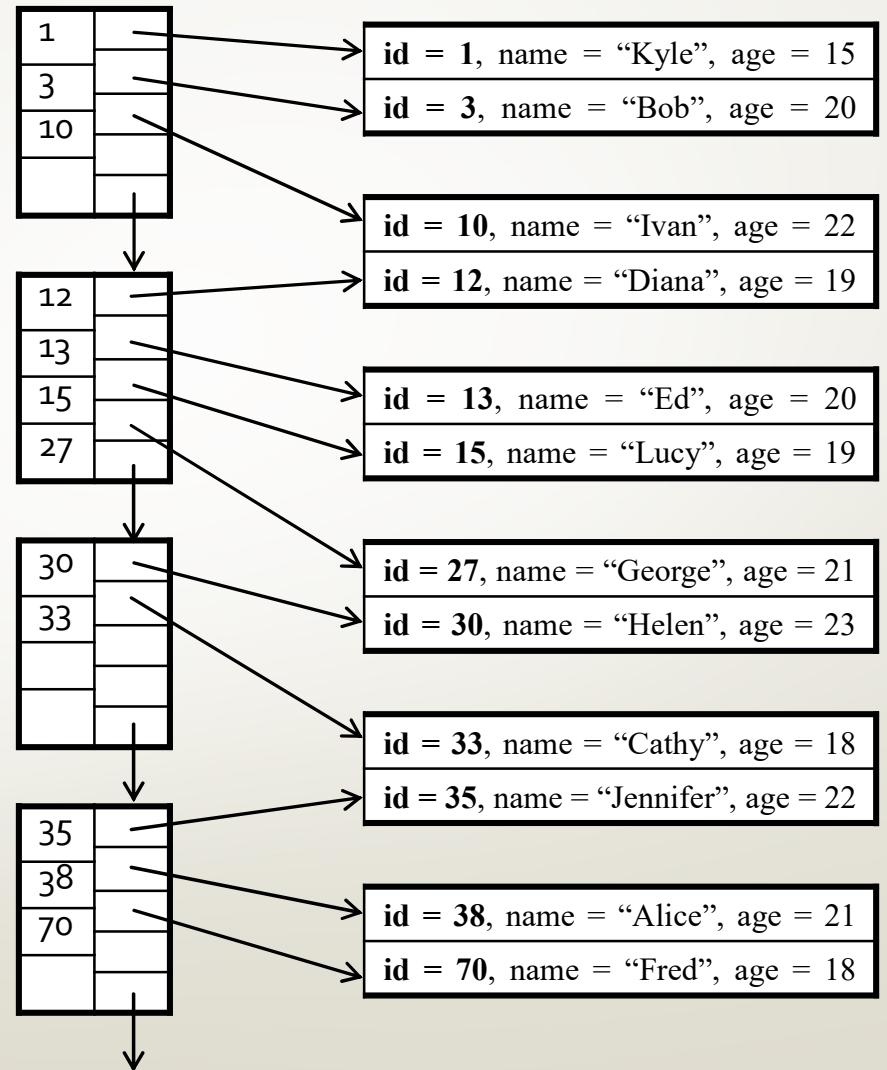
Root can have 1 or more filled in keys



Clustered dense (entry for every record)

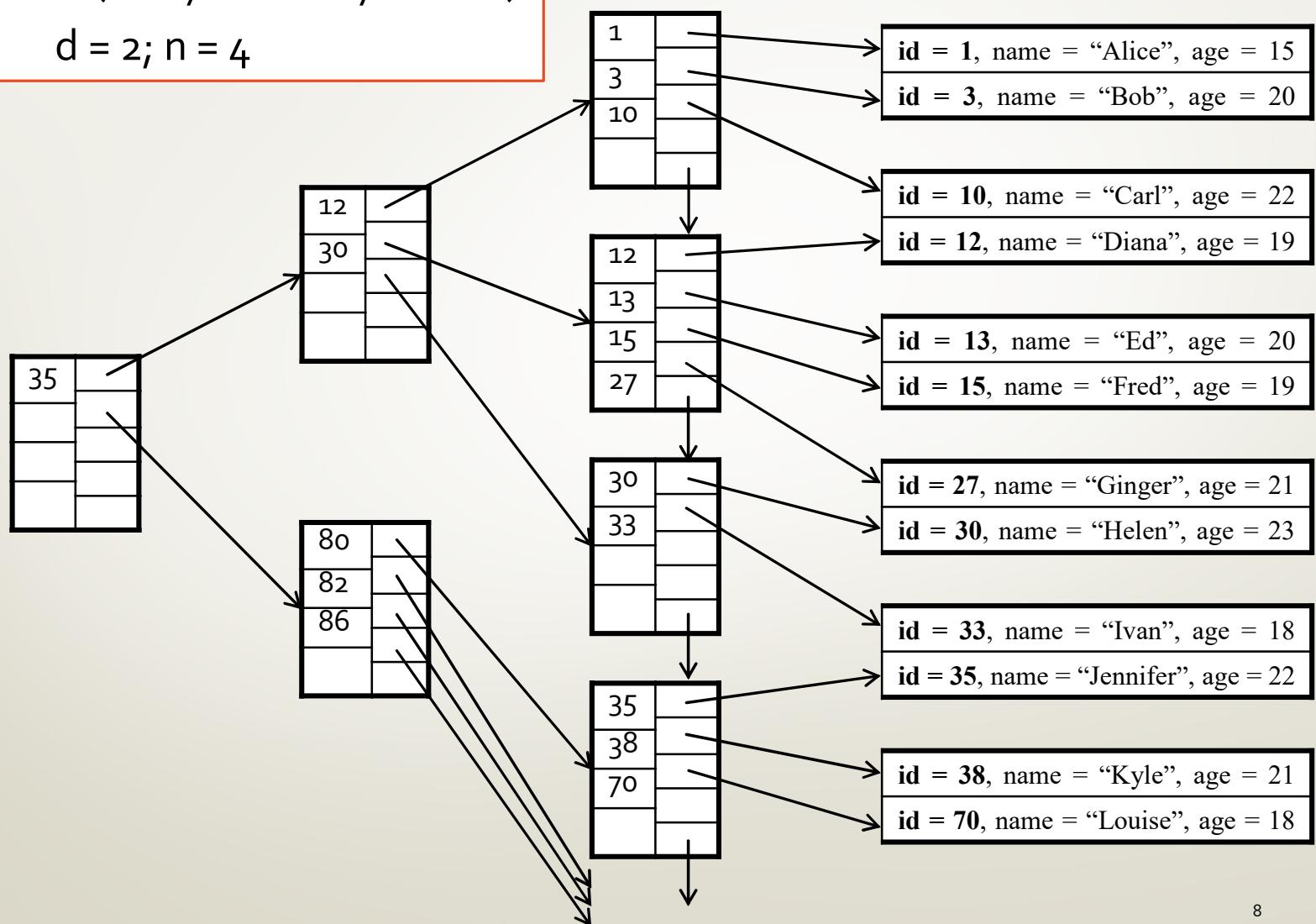
$$d = 2; n = 4$$

B+ Tree search key = **id**



Clustered dense (entry for every record)

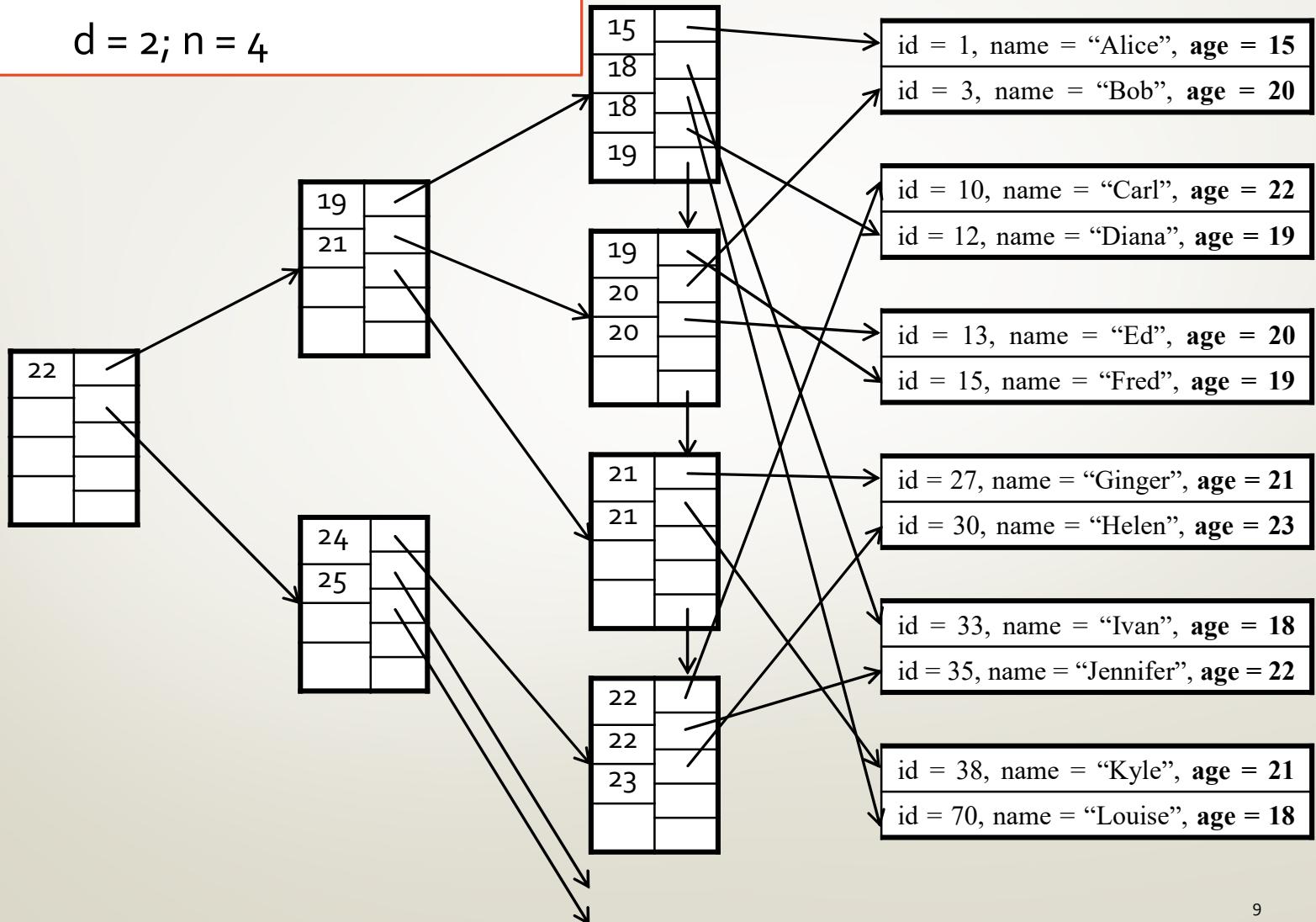
$$d = 2; n = 4$$



Unclustered dense (entry for every record)

$$d = 2; n = 4$$

B+ Tree search key = **age**



B+ Tree Design

- How large should d be?
- Example:
 - Key size = 4 bytes
 - Pointer size = 8 bytes
 - Block size = 4096 bytes
- $2d \times 4 + (2d+1) \times 8 \leq 4096$
- $d = 170$; $2d = 340$

So up to 340 records in leaf blocks

B+ Trees in Practice

- Typical d: 100. Typical fill-factor: 66.5%.
 - average “fanout” = $66.5 * 2 = 133$
- Typical capacities:
 - Height 4: $133^4 = 312,900,700$ records
 - Height 3: $133^3 = 2,352,637$ records
- Can often hold top levels in main memory:
 - Level 1 = 1 page = 8 Kbytes
 - Level 2 = 133 pages = 1 Mbyte
 - Level 3 = 17,689 pages = 133 MBytes

When Do B+ Trees Help?

- Do B+ Trees always help?
 - No. e.g., an array of sorted integers.
- Types of queries to answer with a B+ Tree:
 - *Exact key value*, e.g., SELECT name FROM people WHERE age=20
 - *Range queries*, e.g., SELECT name FROM people WHERE age>=20 and age<=70

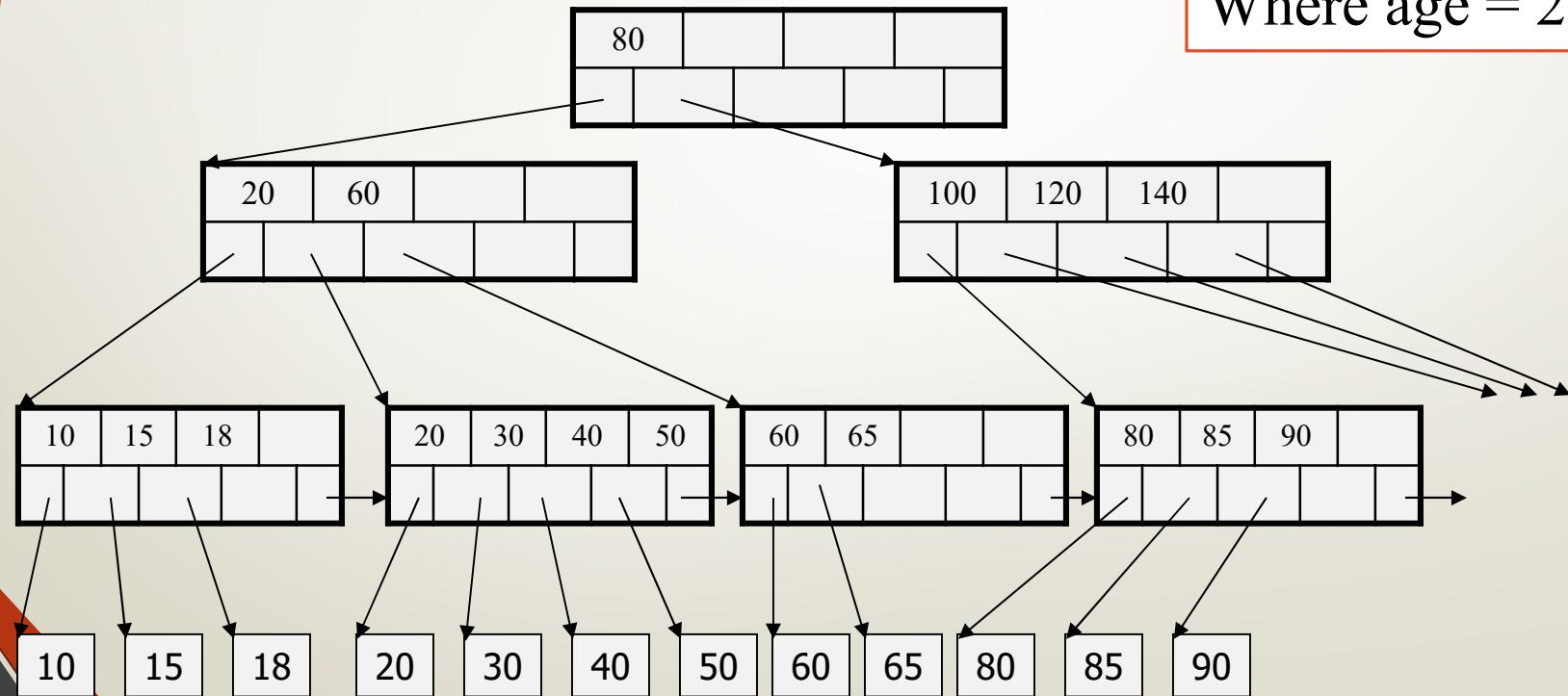


Searching a B+ Tree

Exact key values:

- Start at the root;
- Proceed down to the leaf

Select name
From people
Where age = 25



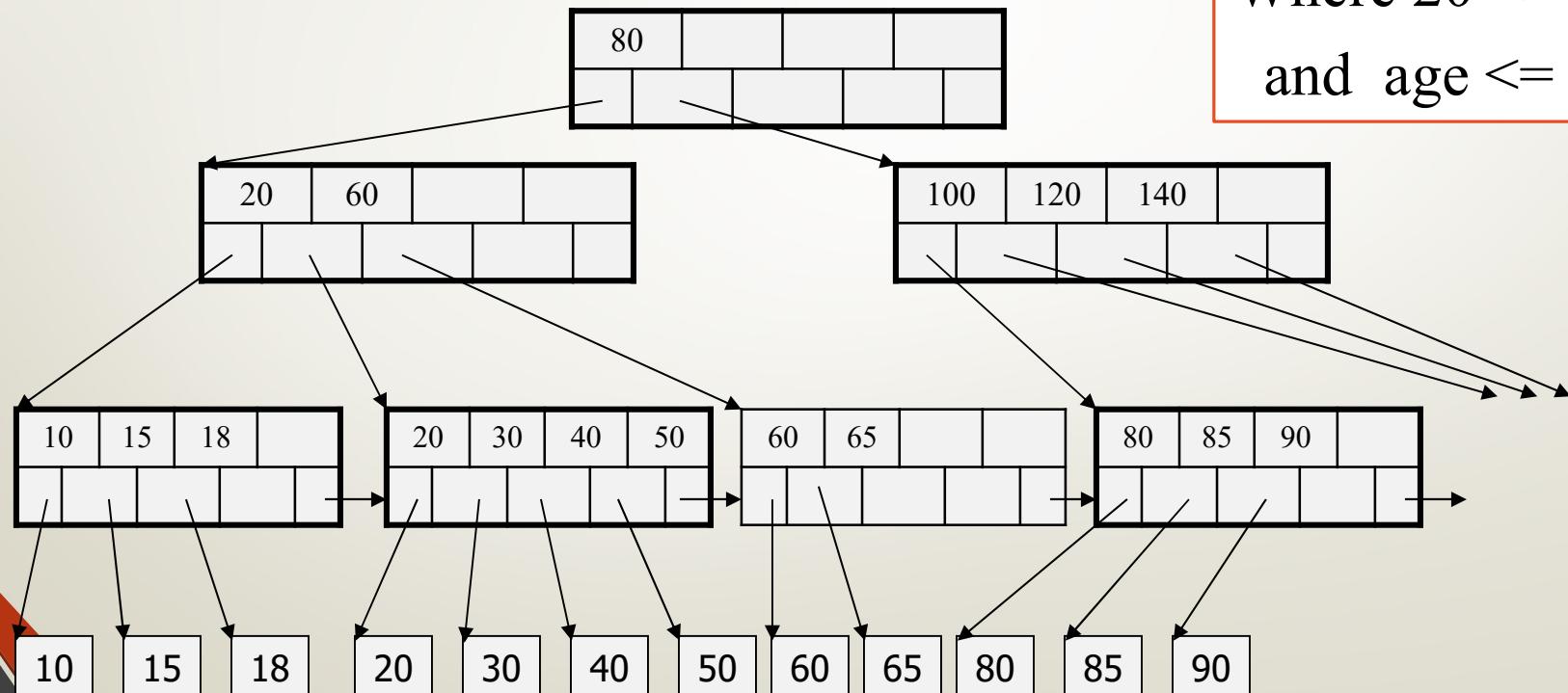


Searching a B+ Tree

Range queries:

- As above
- Then sequential traversal using “next leaf” pointers

Select name
From people
Where $20 \leq \text{age}$
and $\text{age} \leq 70$

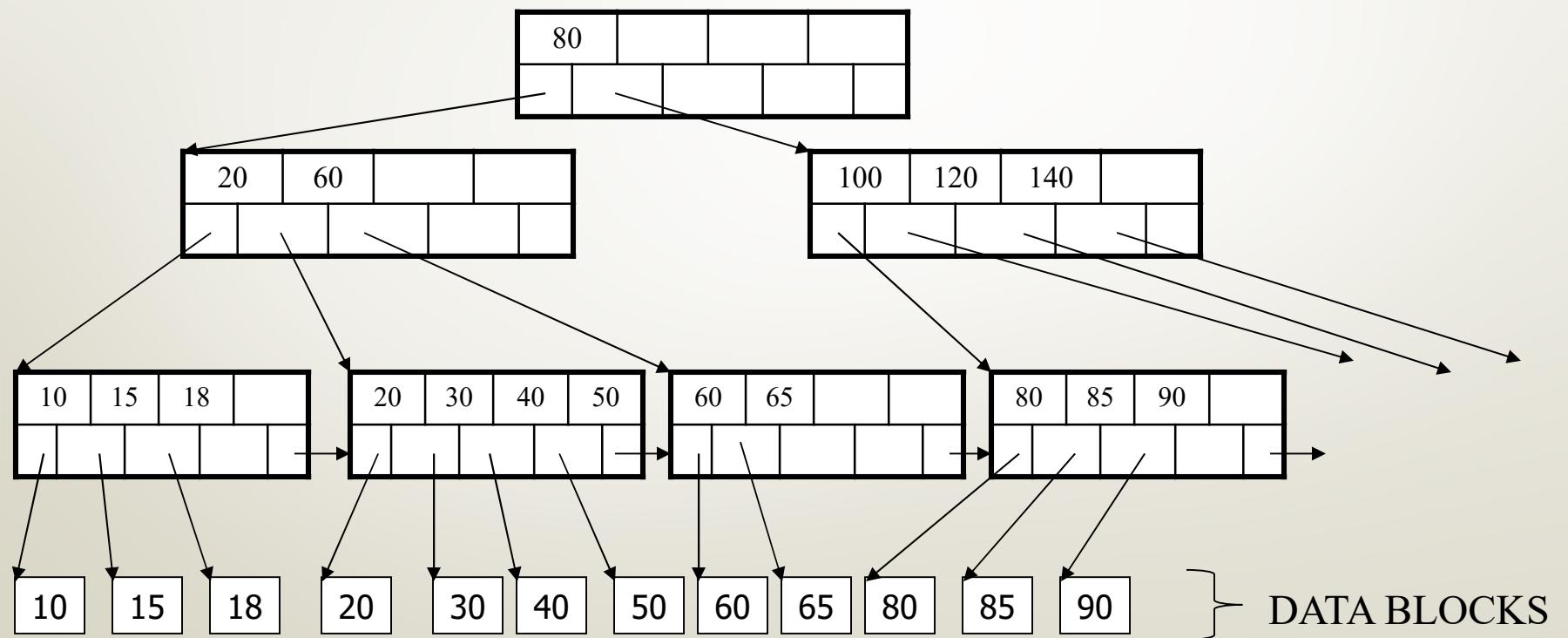


Handling data changes in B+ Trees

Insertion in a B+ Tree

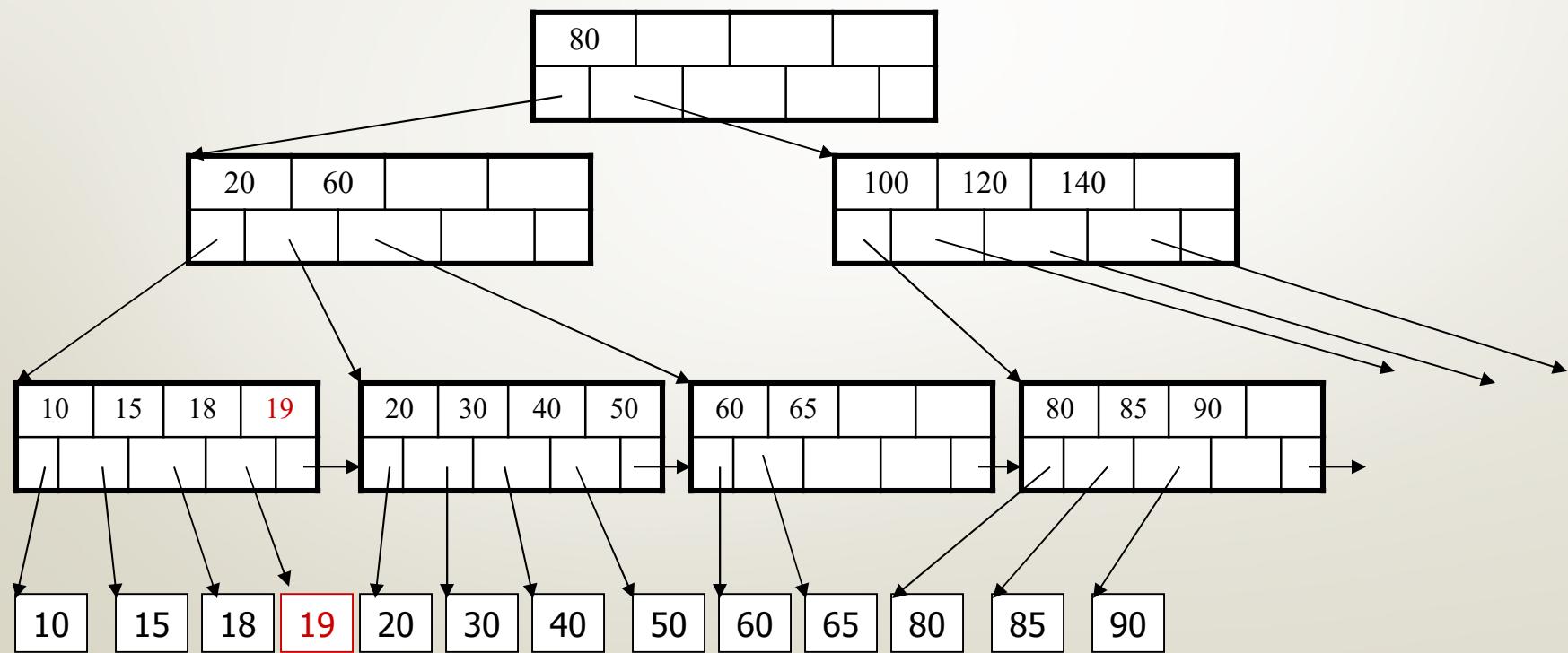
Assume d=2.

Insert K=19



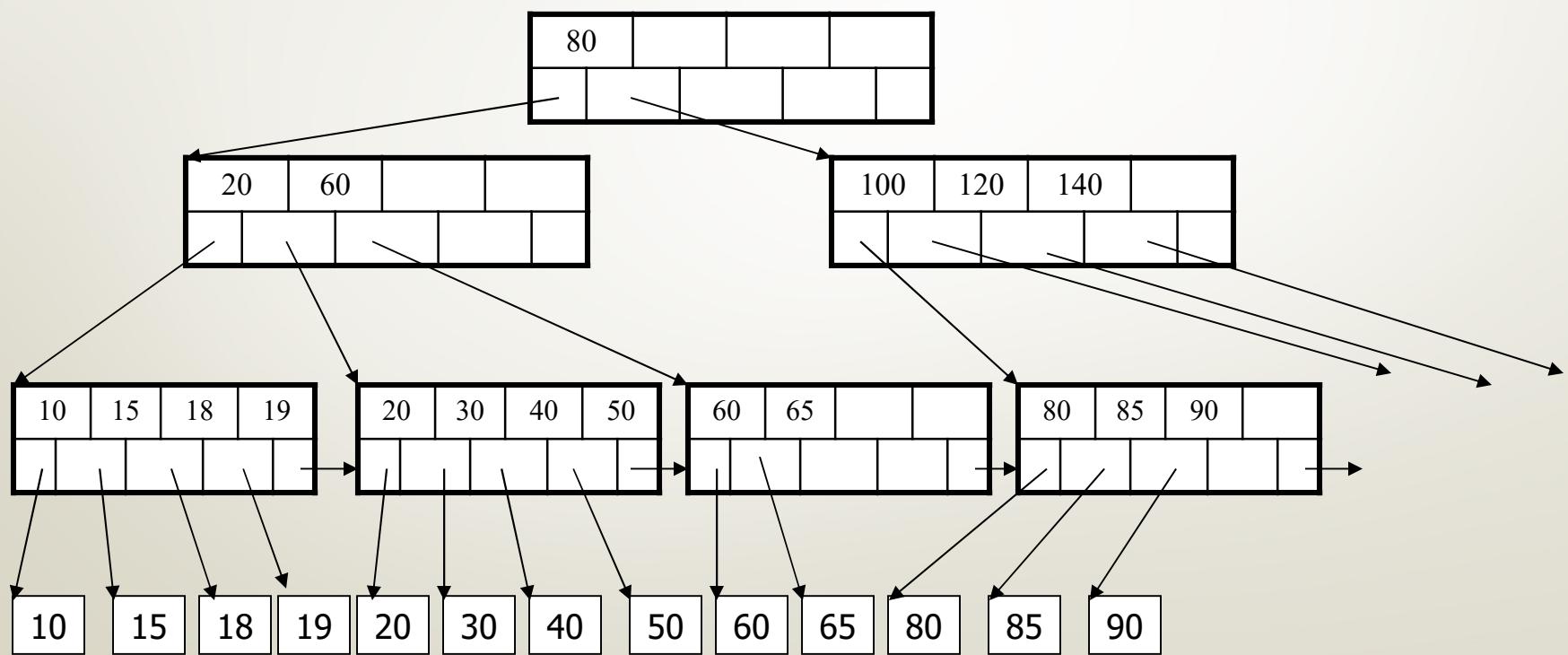
Insertion in a B+ Tree

After insertion



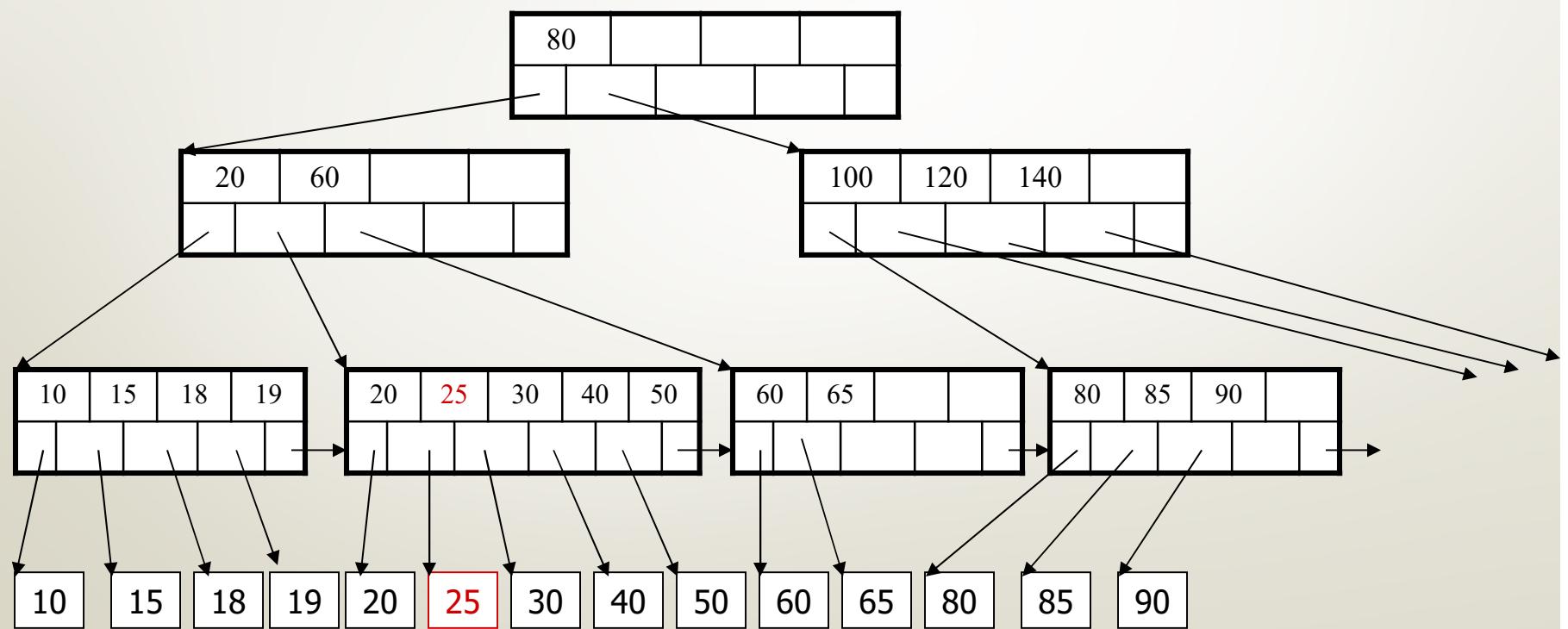
Insertion in a B+ Tree

Now insert 25



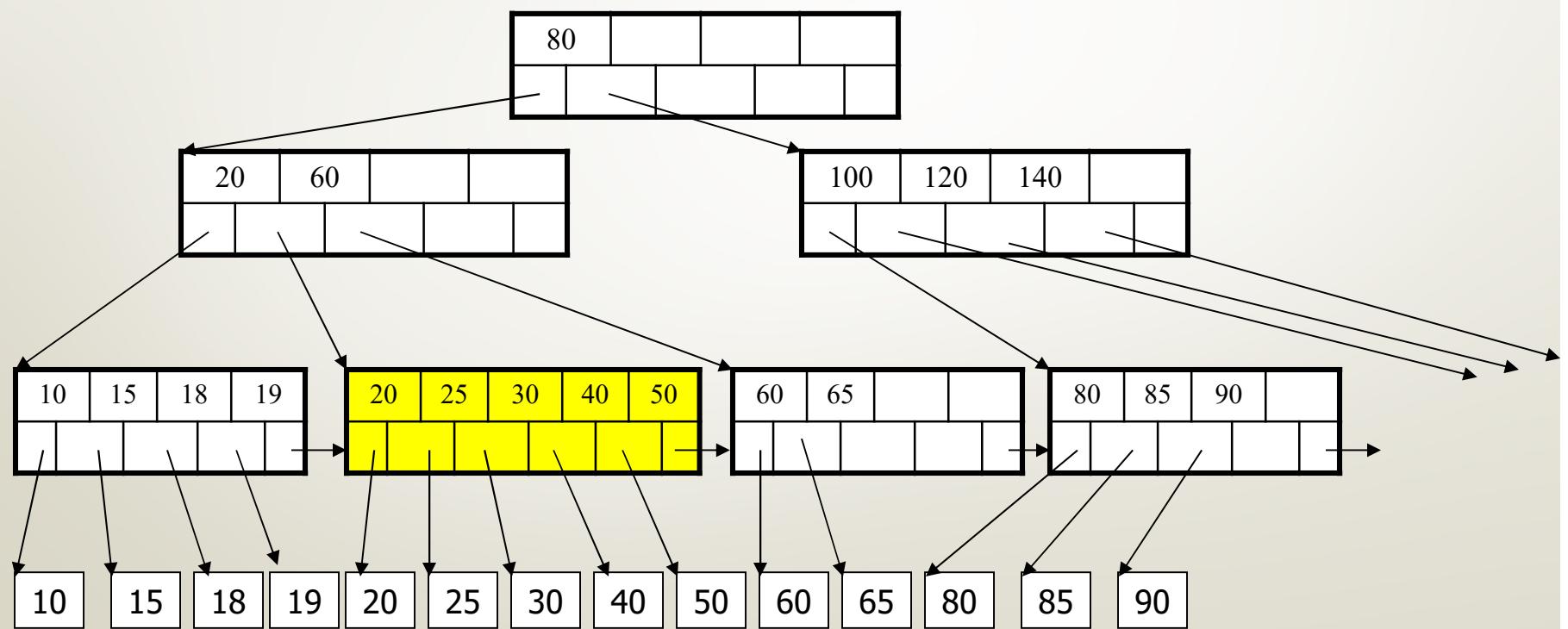
Insertion in a B+ Tree

After insertion



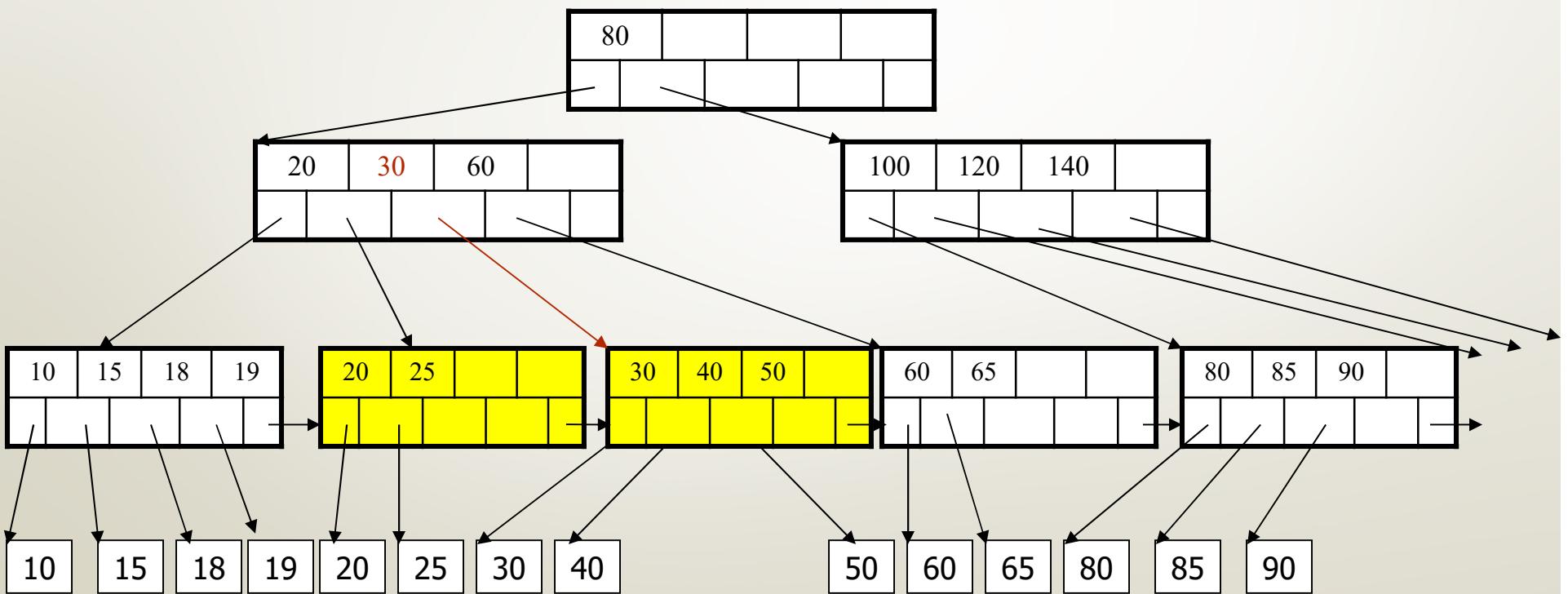
Insertion in a B+ Tree

But now have to split !



Insertion in a B+ Tree

After the split

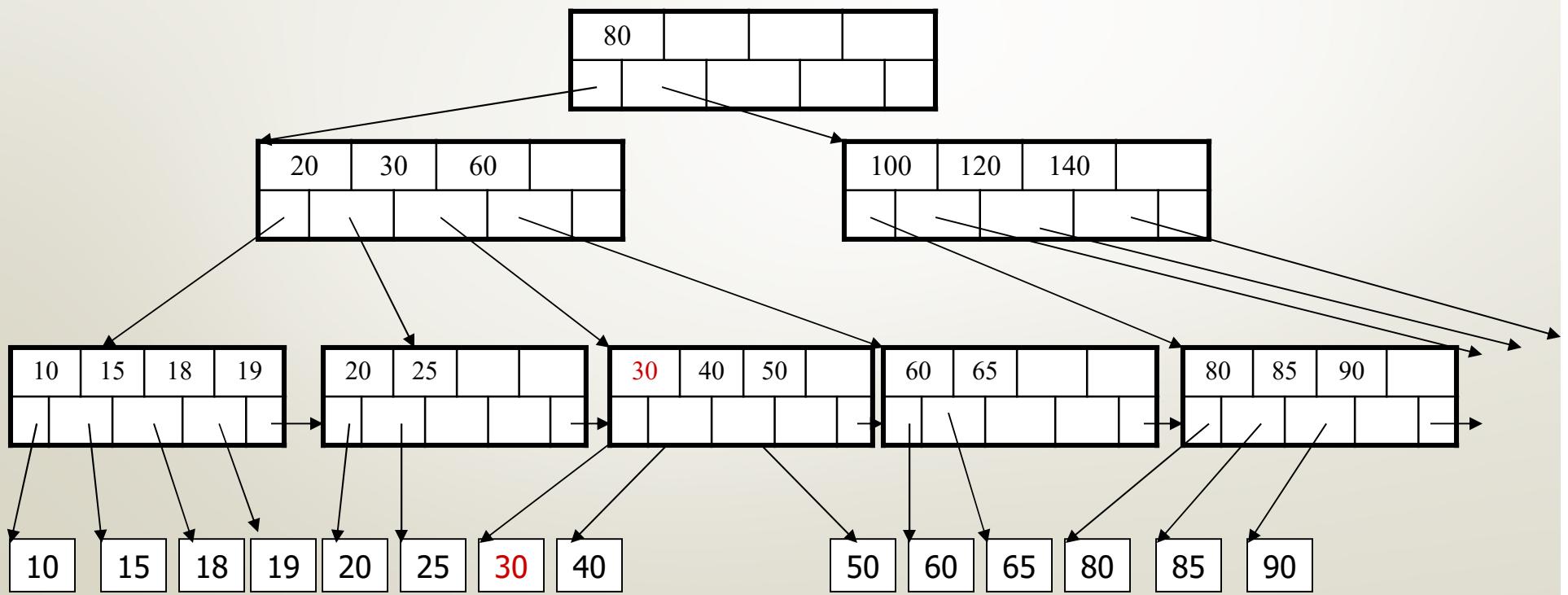


Outline

- B+ Trees
 - ✓ Inserting
 - Deletion

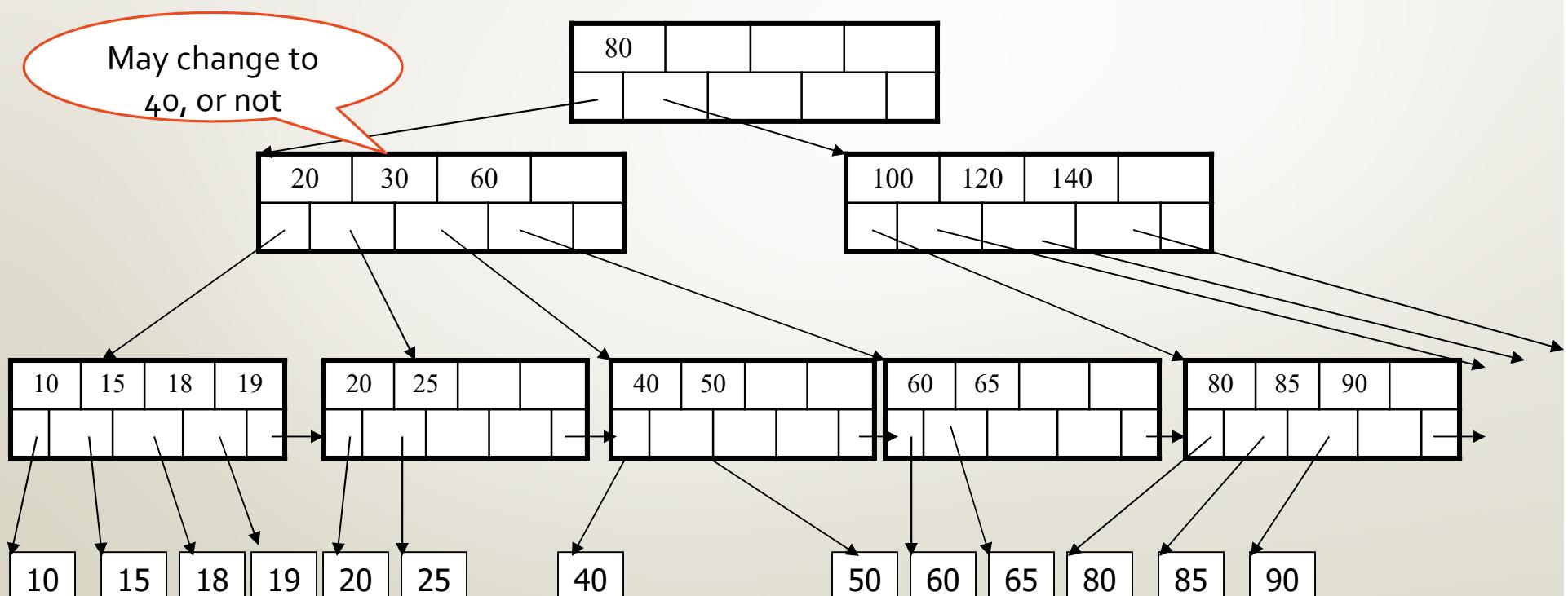
Deletion from a B+ Tree

Delete 30



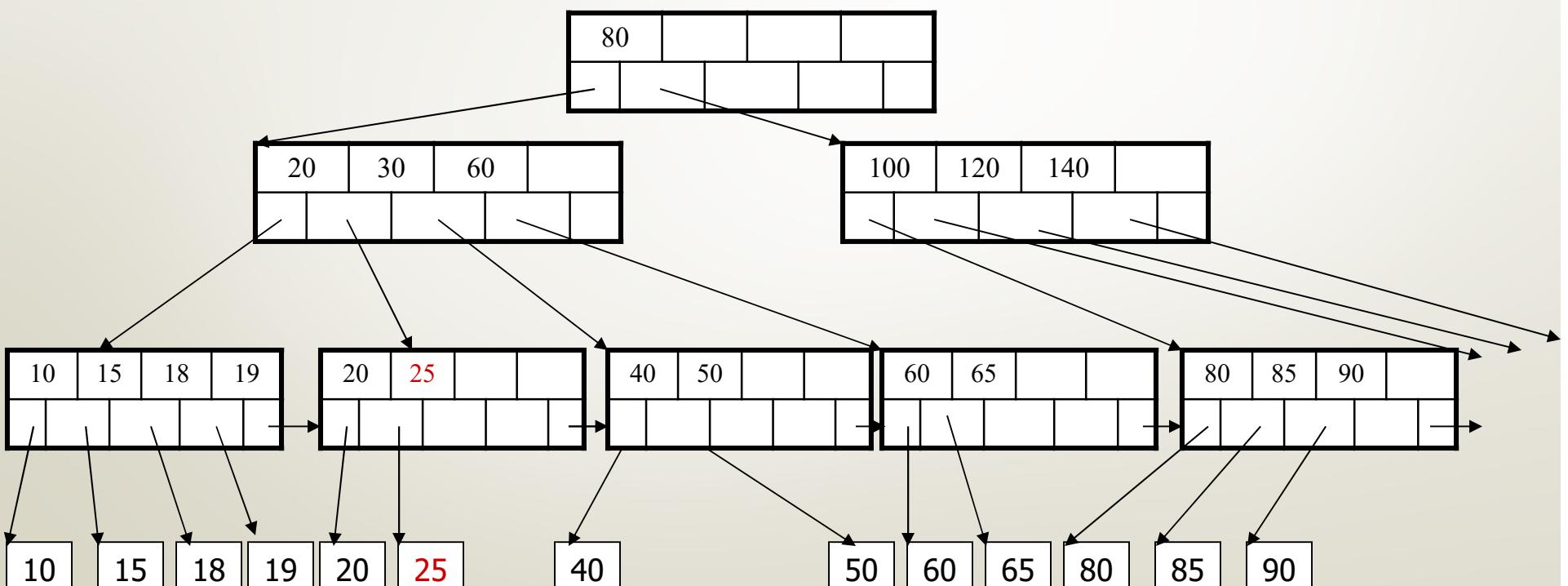
Deletion from a B+ Tree

After deleting 30



Deletion from a B+ Tree

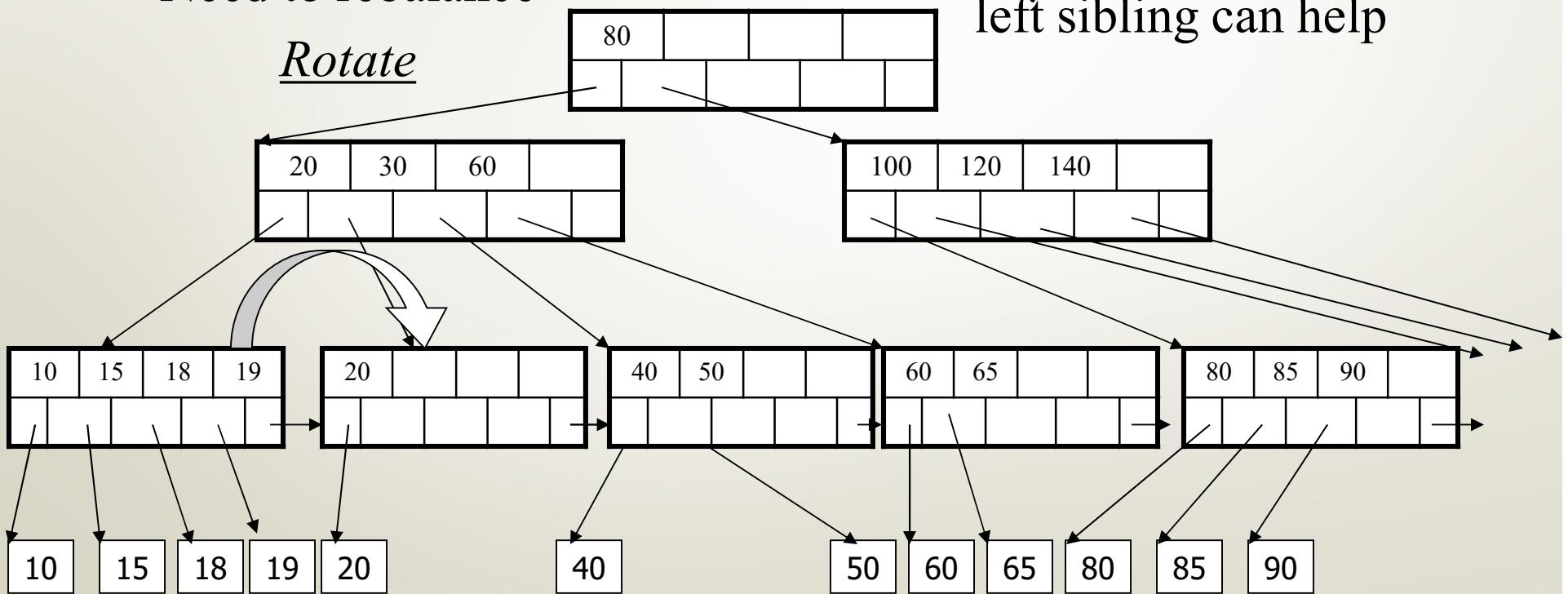
Now delete 25



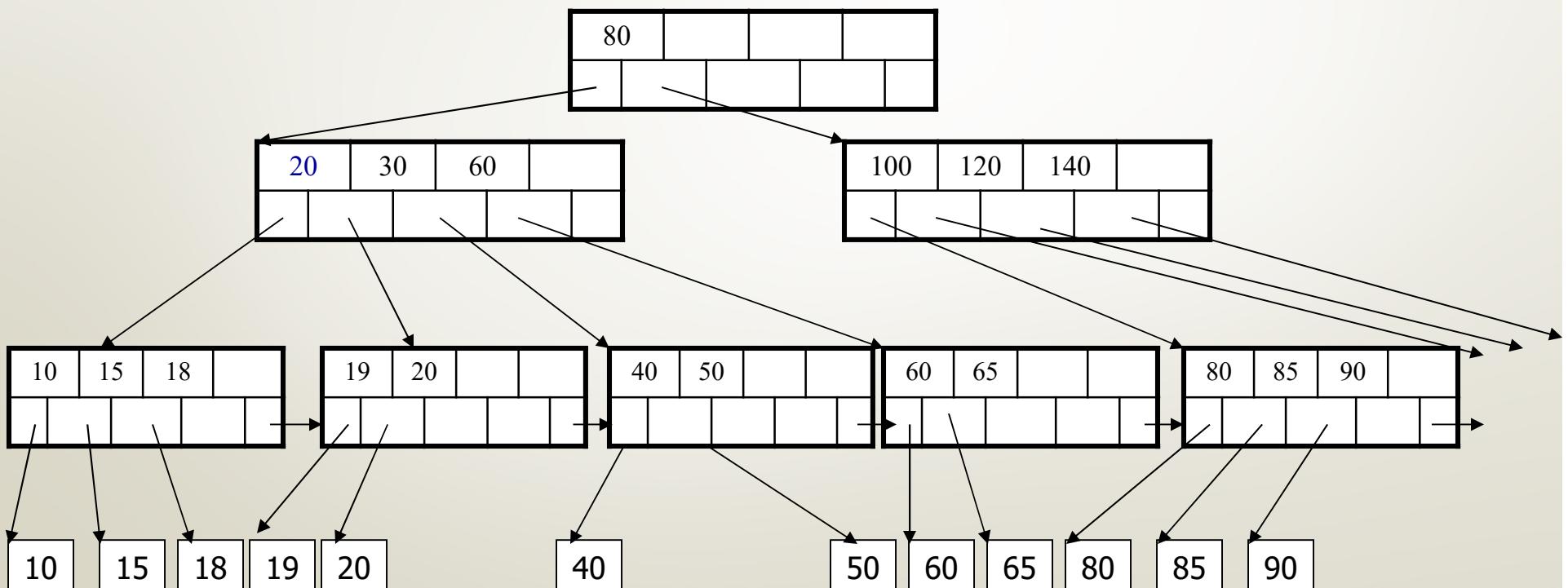
Deletion from a B+ Tree

After deleting 25
Need to rebalance

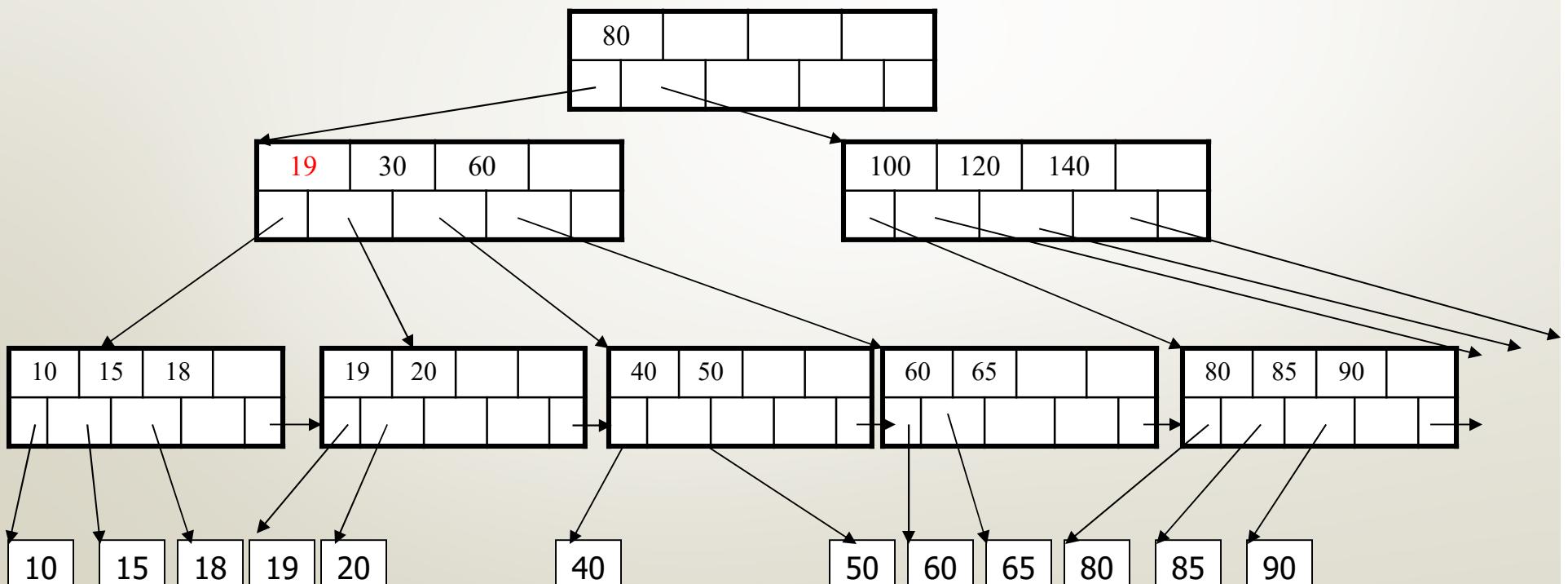
Rotation in general can involve
either sibling, but here only the
left sibling can help



Deletion from a B+ Tree

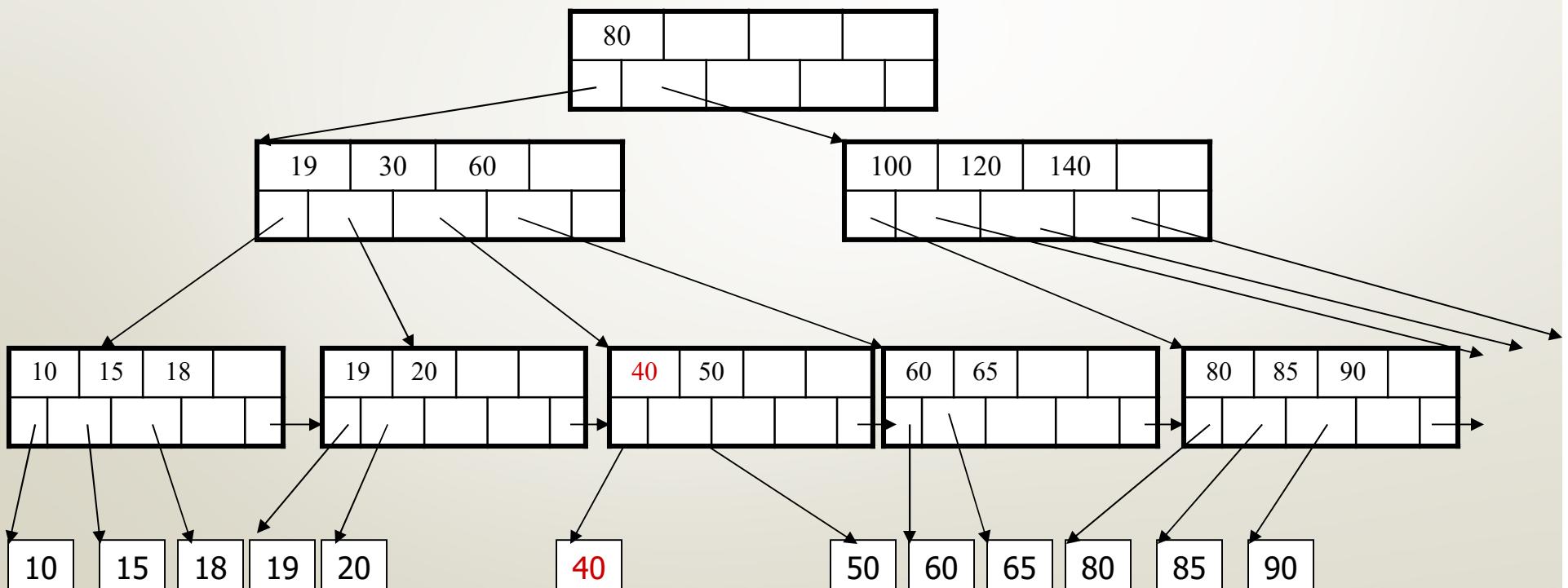


Deletion from a B+ Tree



Deletion from a B+ Tree

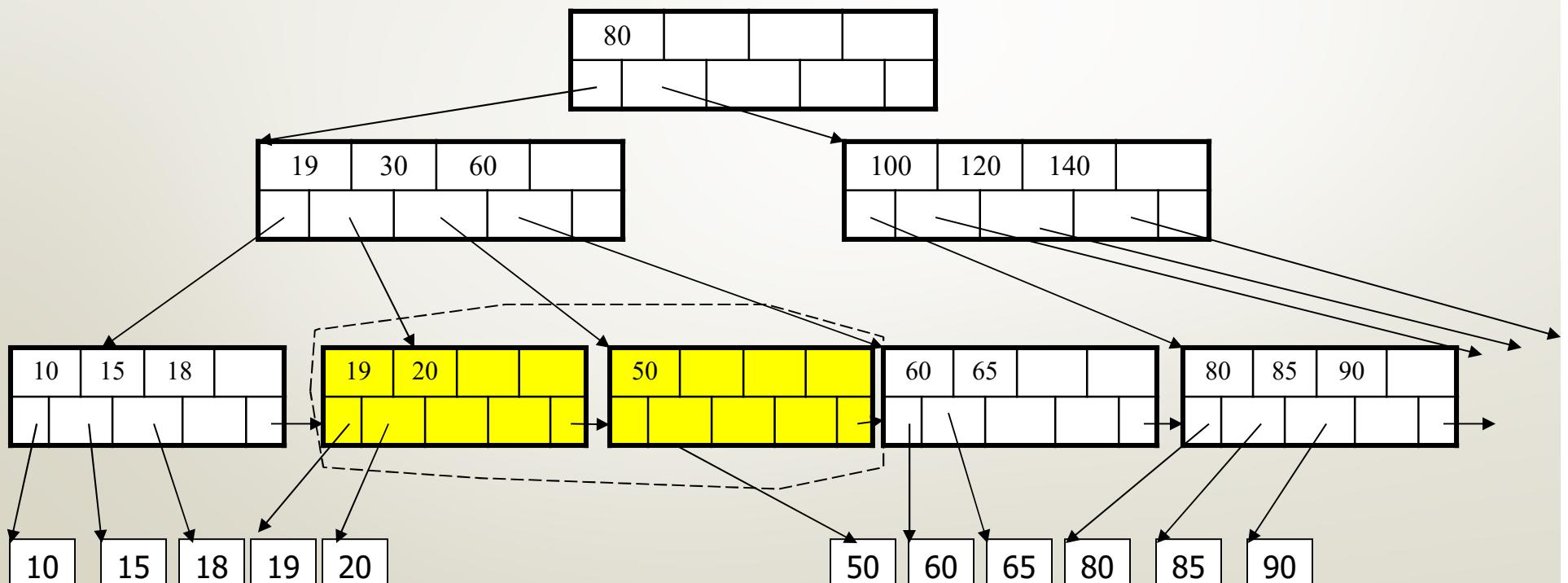
Now delete 40



Deletion from a B+ Tree

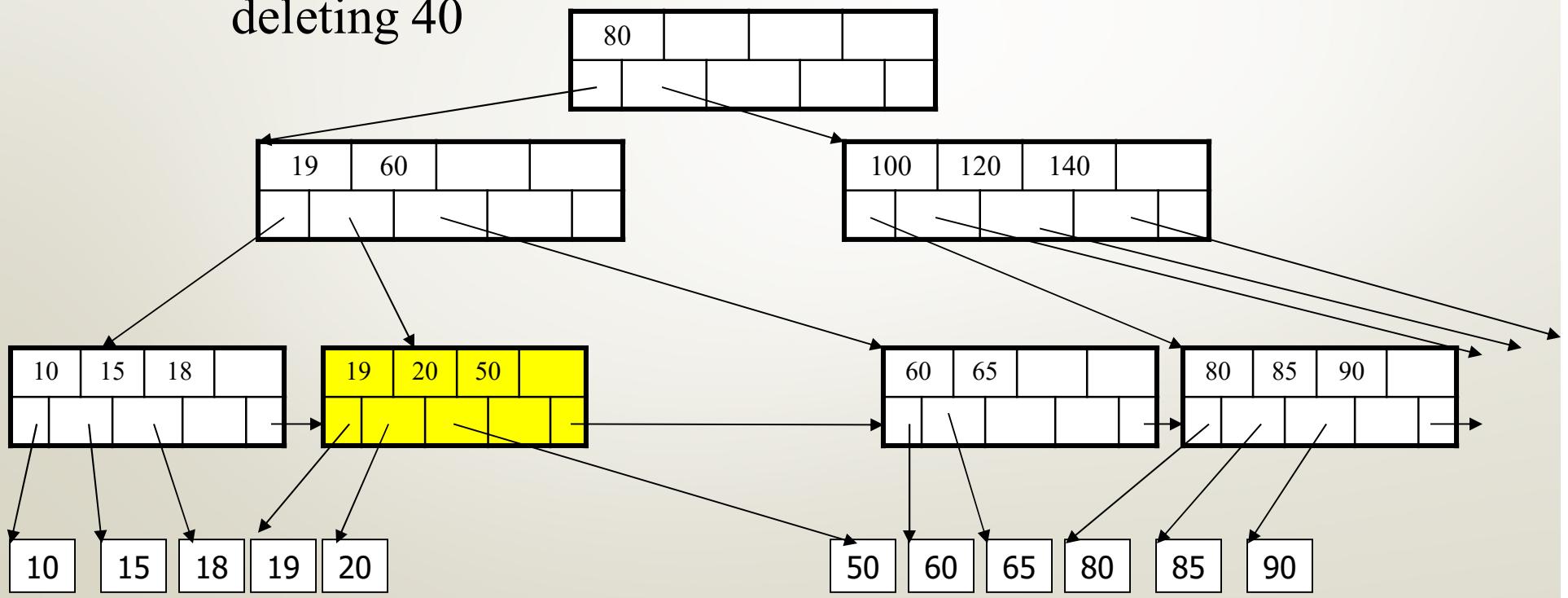
After deleting 40
Rotation not possible

Need to merge nodes

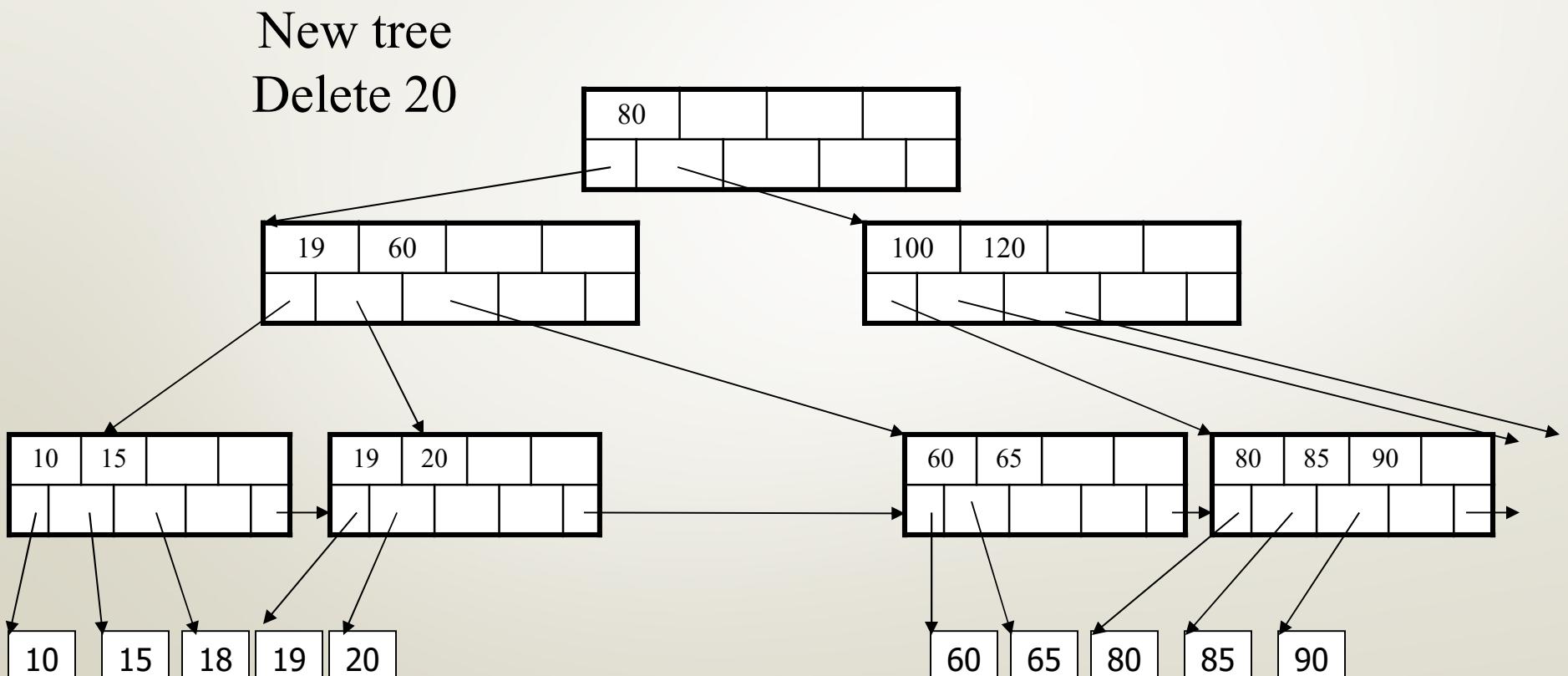


Deletion from a B+ Tree

Tree after
deleting 40

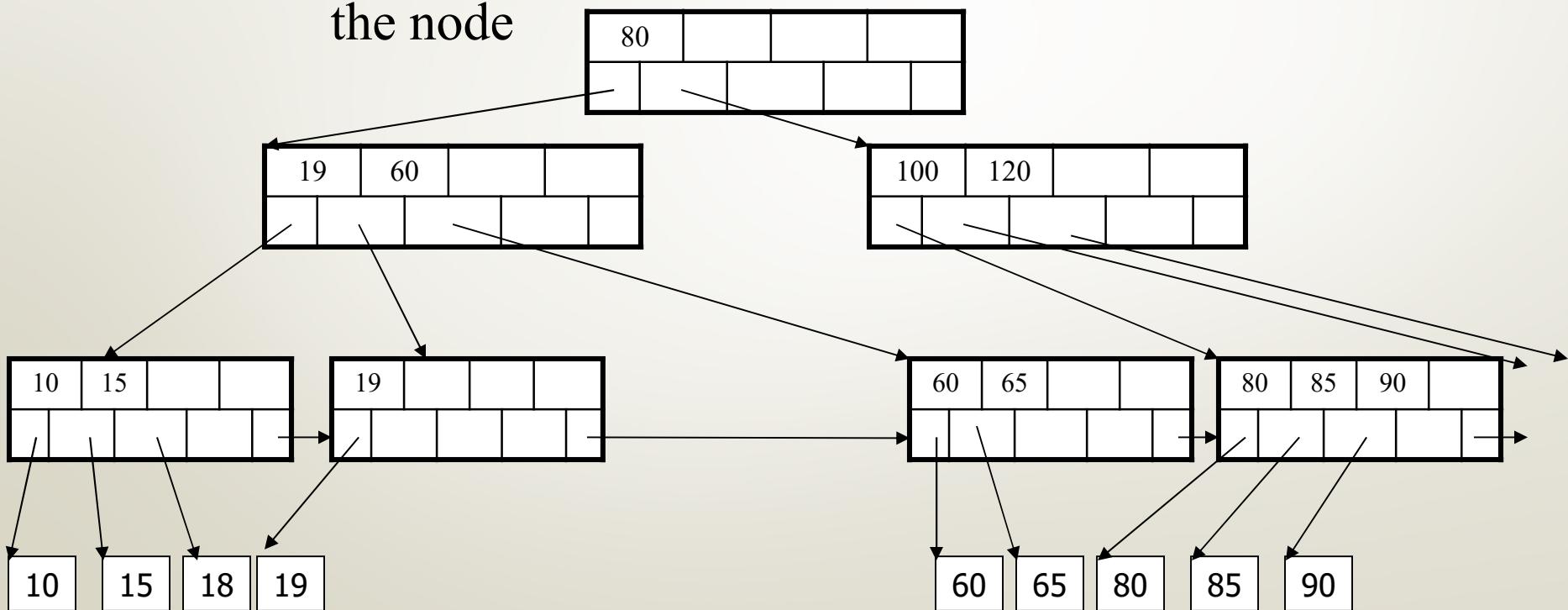


Deletion from a B+ Tree



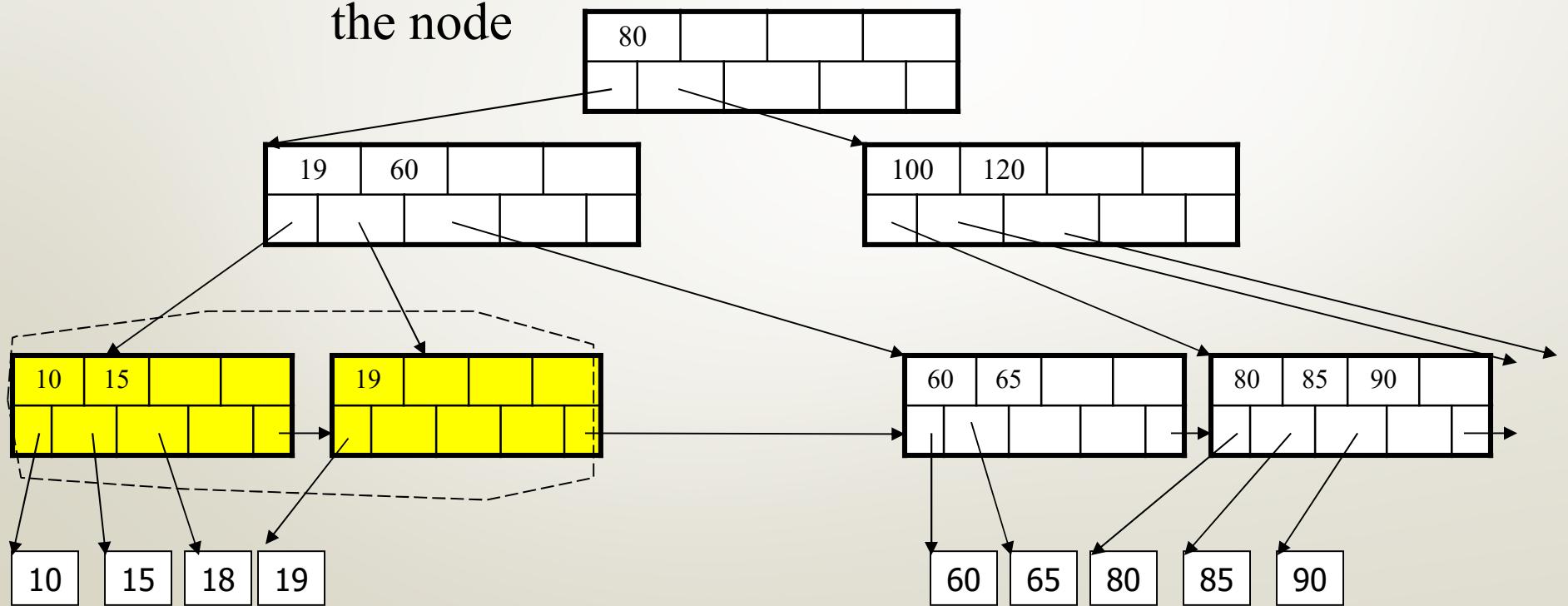
Deletion from a B+ Tree

Rotation is not possible
We have to delete
the node



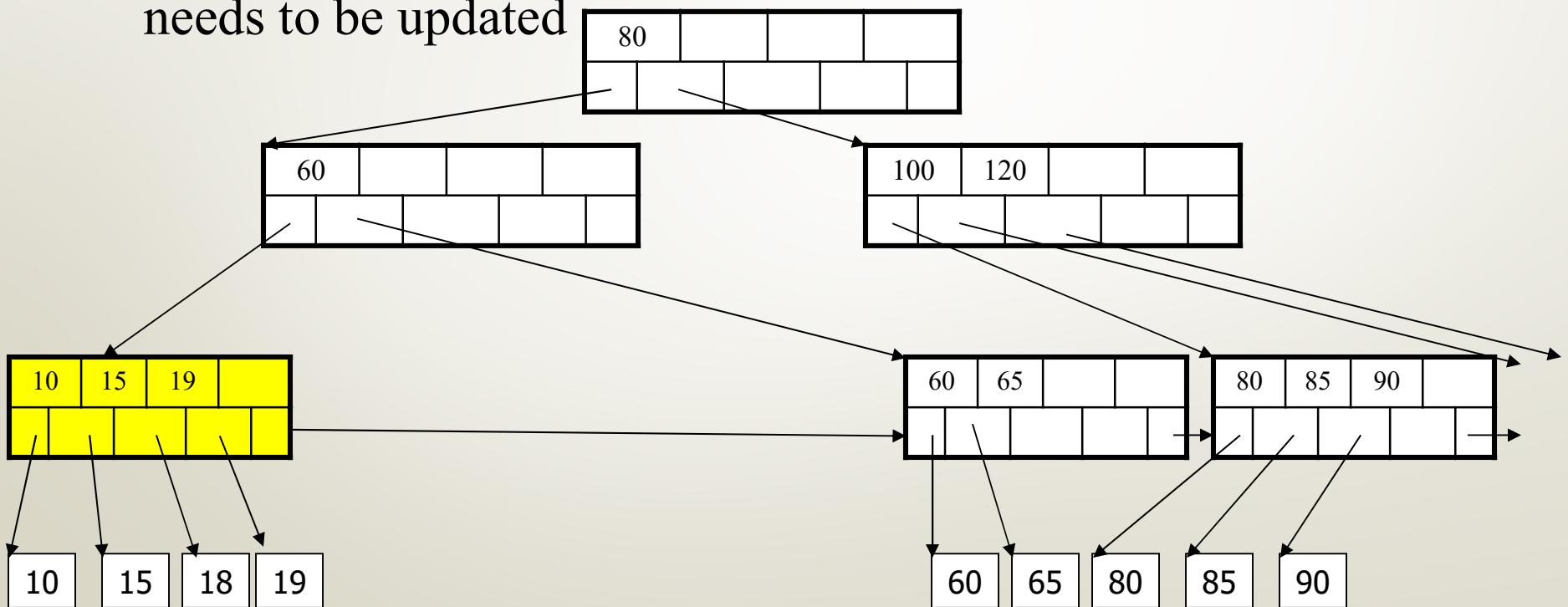
Deletion from a B+ Tree

Rotation is not possible
We have to delete
the node



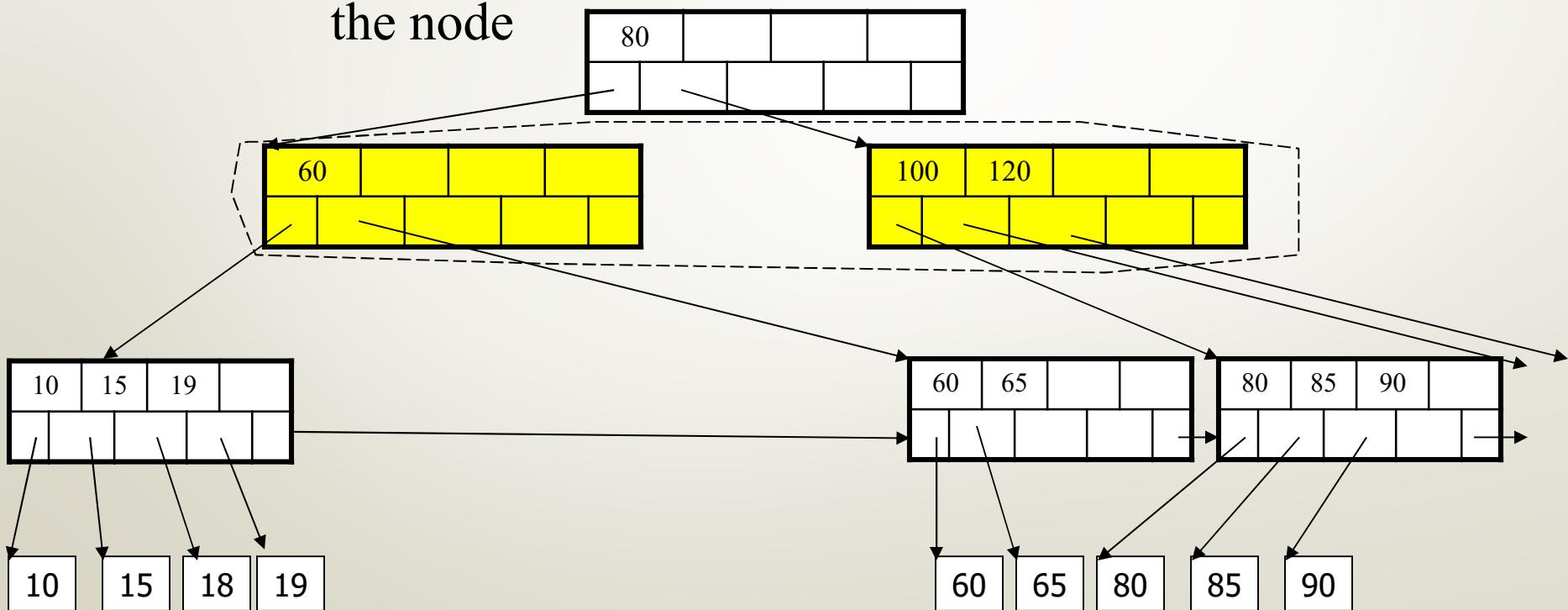
Deletion from a B+ Tree

The parent node
needs to be updated



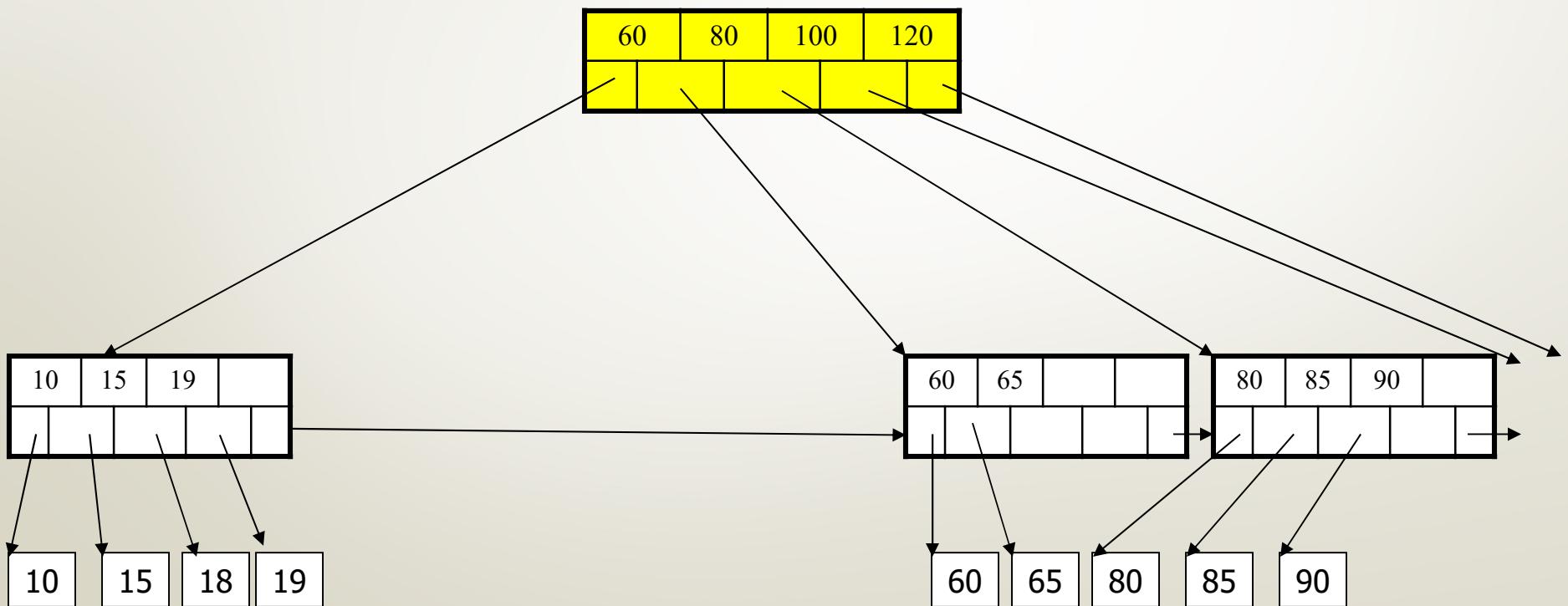
Deletion from a B+ Tree

Rotation is not possible
We have to delete
the node



Deletion from a B+ Tree

Final tree



Advantages of B+Trees

- Balanced → Uniform space utilization
 - Predictable organization Can we do better?
 - Predictable time (logarithmic);
unbalanced can be linear in worst case
- Good for range queries