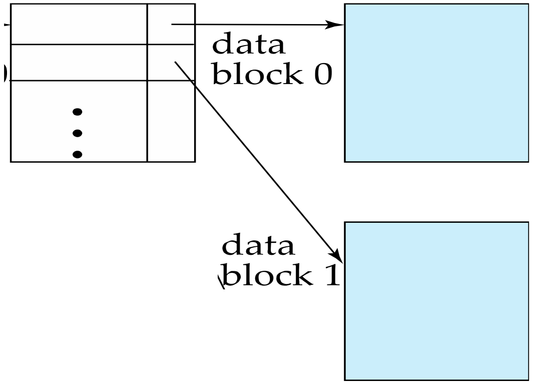
Chapter 11: Indexing

Edited by Radhika Sukapuram

Database System Concepts, 7th Ed.

Insertion of Records

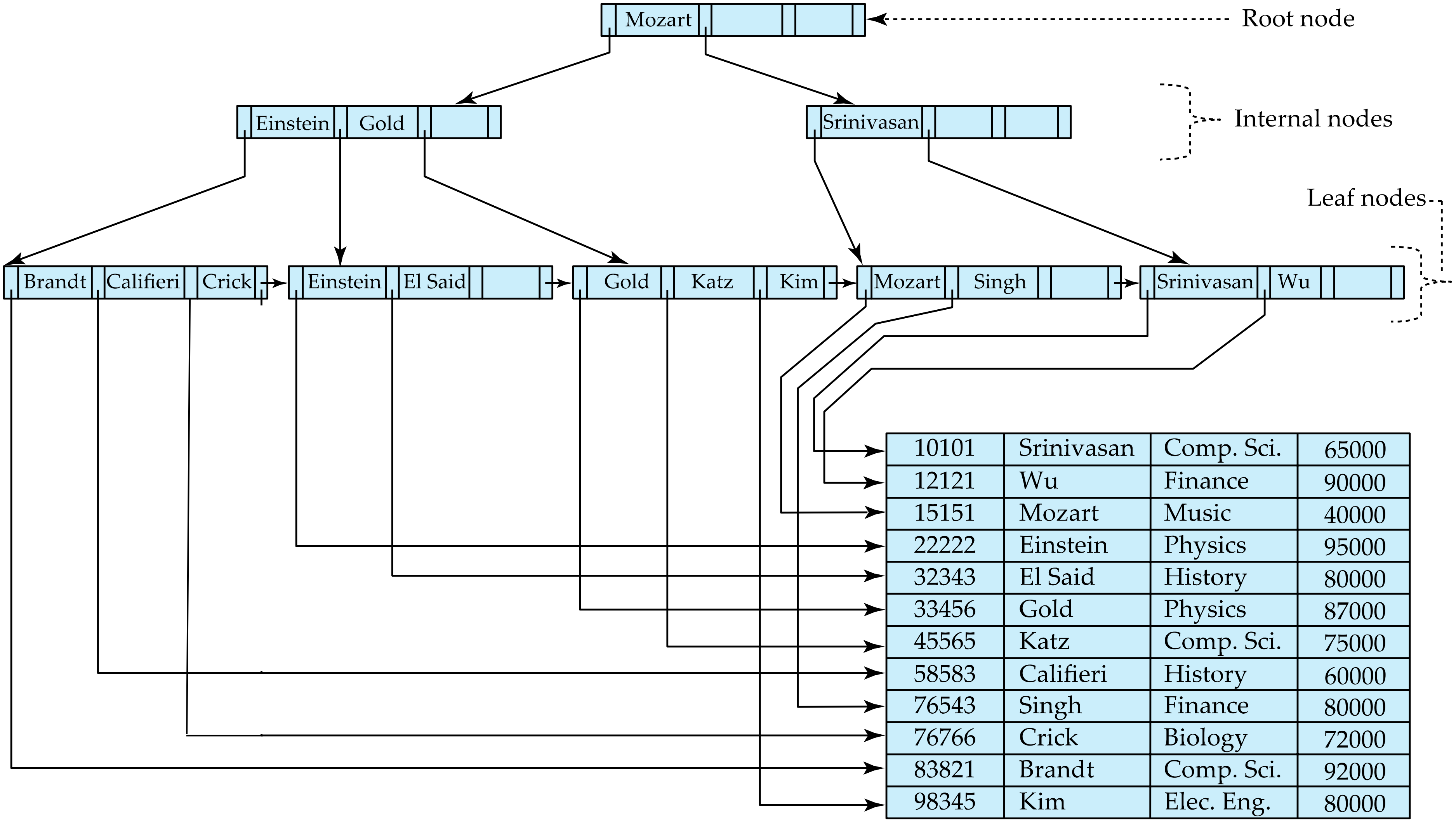
- Perform a lookup using the search-key value of the record to be inserted.
  - **Dense indices** – if the search-key value does not appear in the index
    - insert it.
  - **Sparse indices** – if index stores an entry for each block of the file, no change needs to be made to the index unless a new block is created.
    - If a new block is created, the first search-key value appearing in the new block is inserted into the index.



B$^+$-Tree Index Files

- Advantage of B$^+$-tree index files:
  - Automatically reorganizes itself with small, local, changes, in the face of insertions and deletions.
  - Performance degrades as index files grow in index-sequential files. Reorganization of entire index file is not required to maintain performance
- (Minor) disadvantage of B$^+$-trees:
  - extra insertion and deletion overhead, space overhead.
- Advantages of B$^+$-trees outweigh disadvantages
  - B$^+$-trees are used extensively

Example of B+-Tree of Degree 4



10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	80000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	60000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

B$^+$-Tree Index Files (Cont.)

A B$^+$-tree is a rooted tree satisfying the following properties:

- All paths from root to leaf are of the same length (balanced tree)
- Each node that is not a root or a leaf has between $\lceil n/2 \rceil$ and *n* children.
- A leaf node has between $\lceil (n-1)/2 \rceil$ and $n-1$ values
- Special cases:
  - If the root is not a leaf, it has at least 2 children.
  - If the root is a leaf (that is, there are no other nodes in the tree), it can have between 0 and ($n-1$) values.

  Note: *n-1* is the maximum number of search key values in a leaf node (n is called the degree of a tree)

B^+-Tree Node Structure

- Typical node

P_1	K_1	P_2	…	$P_{n\text{-}1}$	$K_{n\text{-}1}$	P_n

  - K_i are the search-key values
  - P_i are pointers to children (for non-leaf nodes) or pointers to records or buckets of records (for leaf nodes).
    - P_n of a leaf node points to the next leaf node
- The search-keys in a node are ordered

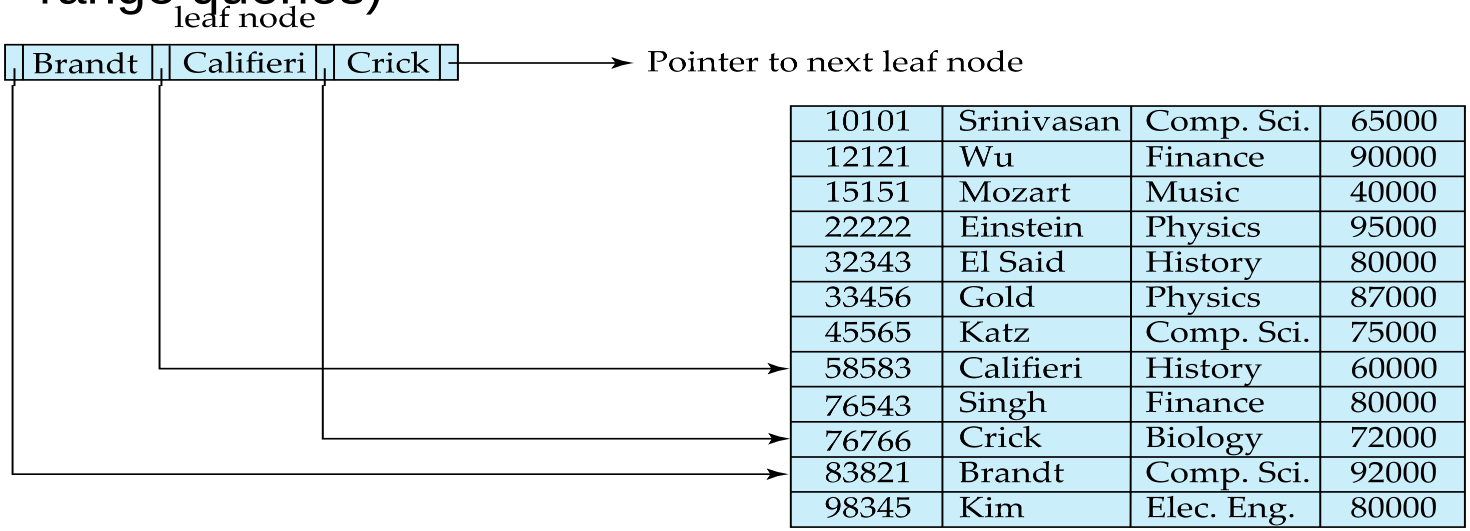
$$K_1 < K_2 < K_3 < . . . < K_{n-1}$$

(assume no duplicate keys)

Leaf Nodes in B$^+$-Trees

Properties of a leaf node:

- For $i = 1, 2, \ldots, n-1$, pointer P_i points to a file record with search-key value K_i,
- If L_i, L_j are leaf nodes and $i < j$, L_i's search-key values are less than or equal to L_j's search-key values
- P_n points to next leaf node in search-key order (for range queries)



10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	80000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	60000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

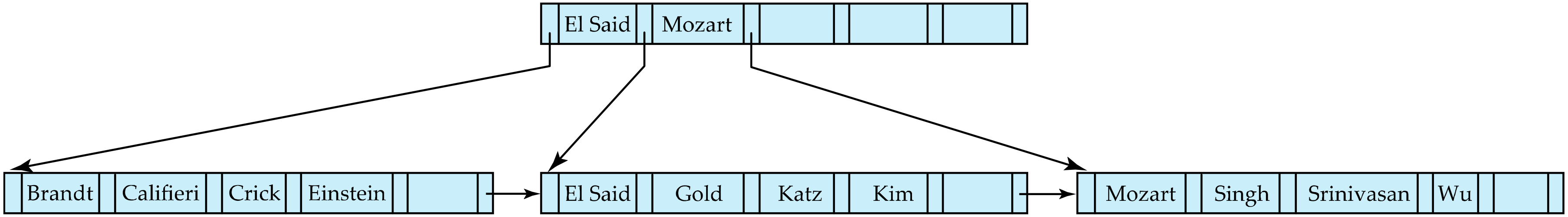
Non-Leaf Nodes in B$^+$-Trees

- Non leaf nodes form a multi-level sparse index on the leaf nodes. For a non-leaf node with *m* pointers:
  - All the search-keys in the subtree to which P_1 points are less than K_1
  - For $2 \leq i \leq n - 1$, all the search-keys in the subtree to which P_i points have values greater than or equal to K_{i-1} and less than K_i
  - All the search-keys in the subtree to which P_n points have values greater than or equal to K_{n-1}

P_1	K_1	P_2	$\dots$	P_{n-1}	K_{n-1}	P_n

Example of B$^+$-tree of Degree 6

- Leaf nodes must have between 3 and 5 values ($\lceil (n-1)/2 \rceil$ and n –1, with n = 6).
- Non-leaf nodes other than root must have between 3 and 6 children ($\lceil (n/2 \rceil$ and n with n =6).
- Root must have at least 2 children.



End of Chapter 14

Edited by Radhika Sukapuram

Database System Concepts, 7th Ed.