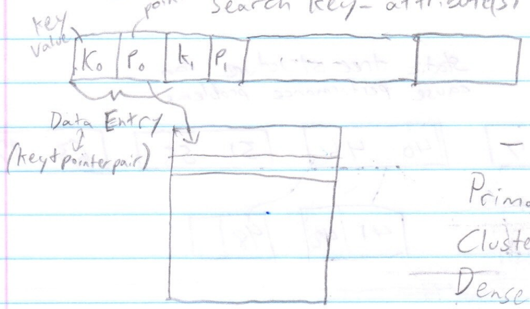
11/12 Notes

Indexing

Search key - attribute(s) of table table is built on



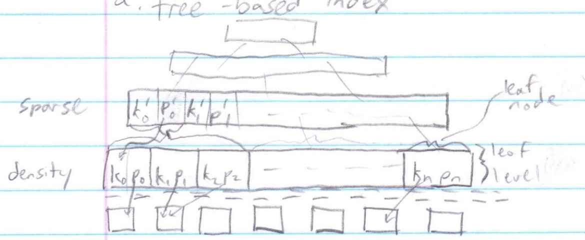
— covered last lecture —

Primary vs Secondary
Clustered vs Unclustered
Dense vs Sparse

1. Index types,
   a. tree-based
   b. hash-based
   c. bitmap-based

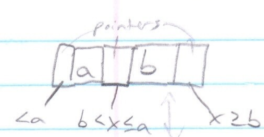
Performance measure: # of I/O's

a. tree-based index

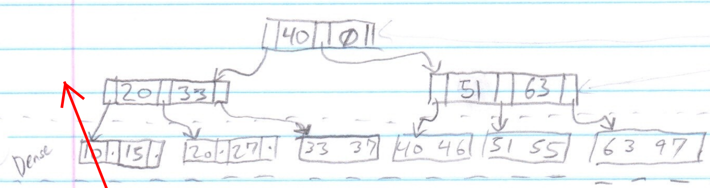


lower level of tree is ordered. index above is built to cover a range of indexes. Continued until top index is a single node

a-1. I SAM: Indexed Sequential Access Method



*f+1 pointers + f key values in each node *



10 15 20 27 33 37 40 46 51 55 63 97

Fanout: 2 (for this ex)

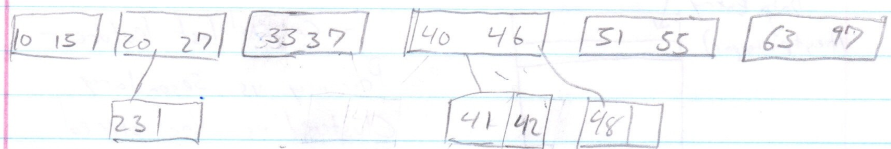
ISAM is a static structure in that all nodes above the leaf level will not be changed as a result of insertion/deletion. We just keep adding more pages (for insertion) or deleting entries (and pages) for deletion on the leaf level.

we could see an highly unbalanced tree, which leaves the logN performance expectation difficult to achieve

Fanout (f): # of data entries in each tree node

Insert: 23, 48, 41, 42

Static tree structure can cause performance problems



B^+-tree

1. d-degree - each node has [d, 2d] data entries
2. Fanout: the exact # of data entries in a node.

Each node has f key values and f+1 pointers

3. Leaf pages are tied together.

Typical B^+-tree in real databases

2d: 200

avg f: 130-140