Indices: The Down Side

- Additional I/O to access index pages (except if index is small enough to fit in main memory)
- Index must be updated when table is modified.
- SQL-92 does not provide for creation or deletion of indices
  - Index on primary key generally created automatically
  - Vendor specific statements:
    - CREATE INDEX ind ON Transcript (CrsCode)
    - DROP INDEX ind

25

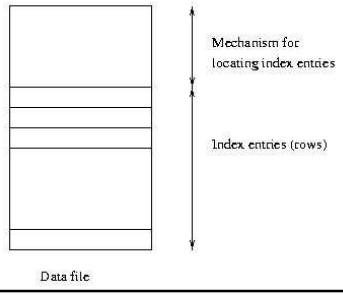
Clustered Index

- Clustered index: index entries and rows are ordered in the same way
  - An integrated storage structure is always clustered (since rows and index entries are the same)
  - The particular index structure (eg, hash, tree) dictates how the rows are organized in the storage structure
    - There can be at most one clustered index on a table
  - CREATE TABLE generally generally creates an integrated, clustered (main) index

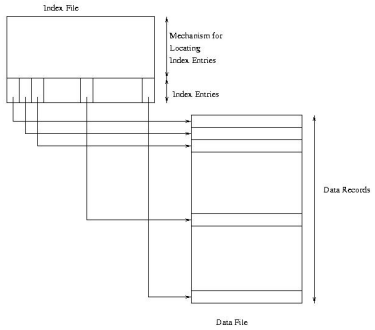
26

Clustered Main Index

Storage structure contains table and (main) index; rows are contained in index entries



Clustered Secondary Index



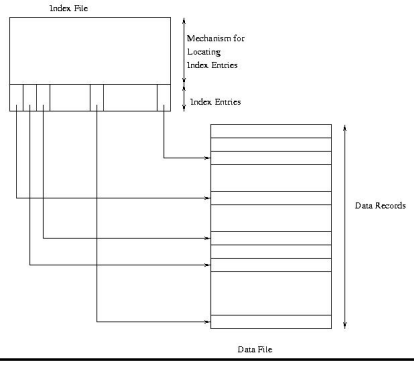
28

Unclustered Index

- Unclustered (secondary) index: index entries and rows are not ordered in the same way
- An secondary index might be clustered or unclustered with respect to the storage structure it references
  - It is generally unclustered (since the organization of rows in the storage structure depends on main index)
  - There can be many secondary indices on a table
  - Index created by CREATE INDEX is generally an unclustered, secondary index

29

Unclustered Secondary Index



Clustered Index

- Good for range searches when a range of search key values is requested
  - Use location mechanism to locate index entry at start of range
    - This locates first row.
  - Subsequent rows are stored in successive locations if index is clustered (not so if unclustered)
  - Minimizes page transfers and maximizes likelihood of cache hits

31

Example - Cost of Range Search

- Data file has 10,000 pages, 100 rows in range
- Page transfers for rows (assuming 20 rows/page):
  - Heap: 10,000 (entire file must be scanned)
  - File sorted on search key: $\log_2 10000 + (5 \text{ or } 6) \approx 19$
  - Unclustered index: ≤ 100
  - Clustered index: 5 or 6
- Page transfers for index entries (assuming 200 entries/page)
  - Heap and sorted: 0
  - Unclustered secondary index: 1 or 2 (all entries must be read)
  - Clustered secondary index: 1 (only first entry must be read)

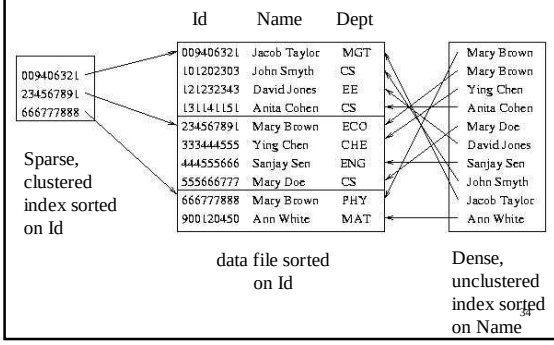
32

Sparse Vs Dense Index

- Dense index: index entry for each data record
  - Unclustered index must be dense
  - Secondary clustered index need not be dense
- Sparse index: index entry for each page of data file

33

Sparse Vs. Dense Index



Sparse, clustered index sorted on Id

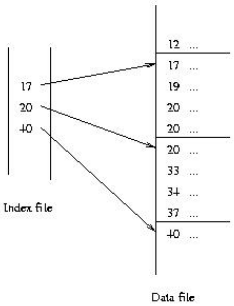
data file sorted on Id

Dense, unclustered index sorted on Name

34

Sparse Index

Search key should be candidate key of data file (else additional measures required)



Multiple Attribute Search Key

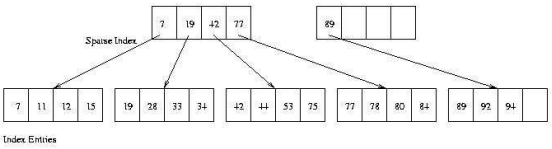
- CREATE INDEX Inx ON Tbl (att1, att2)
- Search key is a *sequence* of attributes; index entries are lexically ordered
- Supports finer granularity equality search:
  - "Find row with value (A1, A2) "
- Supports range search (tree index only):
  - "Find rows with values between (A1, A2) and (A1′, A2′) "
- Supports partial key searches (tree index only):
  - Find rows with values of att1 between A1 and A1′
  - But not "Find rows with values of att2 between A2 and A2′ "

36

Locating an Index Entry

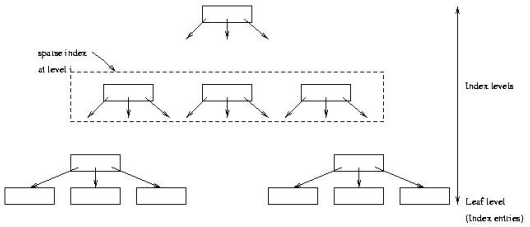
- Use scan (index entries not sorted)
- Use binary search (index entries sorted)
  - If Q pages of index entries, then $log_2 Q$ page transfers (which is a big improvement over binary search of the data pages of a F page data file since $F >> Q$)
- Use multilevel index: Sparse index on sorted list of index entries

Two-Level Index



- *Separator level* is a sparse index over pages of index entries
- *Leaf level* contains index entries
- Cost of searching the separator level << cost of searching index level since separator level is sparse
-Cost or retrieving row once index entry is found is 0 (if integrated) or 1 (if not)

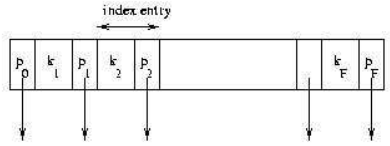
Multilevel Index



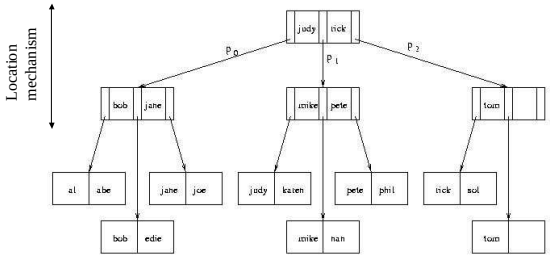
- Search cost = number of levels in tree
- If Φ is the fanout of a separator page, cost is $log_\Phi Q + 1$
- Example: if Φ = 100 and Q = 10,000, cost = 3
  (reduced to 2 if root is kept in main memory)

Index Sequential Access Method

- Generally an integrated storage structure
  – Clustered, index entries contain rows
- Separator entry = (k_i , p_i); k_i is a search key value; p_i is a pointer to a lower level page
- k_i separates set of search key values in the two subtrees pointed at by p_{i-1} and p_i



Index Sequential Access Method

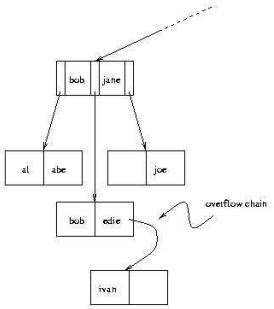


Index Sequential Access Method

- The index is static:
  – Once the separator levels have been constructed, they never change
  – Number and position of leaf pages in file stays fixed
- Good for equality and range searches
  – Leaf pages stored sequentially in file when storage structure is created to support range searches
    - if, in addition, pages are positioned on disk to support a scan, a range search can be very fast (static nature of index makes this possible)
- Supports multiple attribute search keys and partial key searches

Overflow Chain

- Contents of leaf pages change
- Row deletion yields empty slot in leaf page
- Row insertion can result in overflow leaf page and ultimately overflow chain
  - Chain might be long, unsorted, scattered on disk
  - ISAM can be inefficient if table is dynamic



43