Simplest file structure is an unordered file
(heap file).

Can retrieve records from file using record id.
n/2 expected time for n records

An index is a data structure that organized records
to try to improve access speed.

Indexes return data that match a field/attribute
index is on (search key).

A data entry is a record in an index file.

What is stored in a data entry?

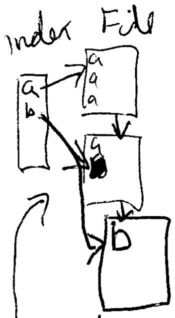
~~[illegible]~~ Some possibilities

① actual record → ~~data~~ heap is ordered

② <K, rid>

③ <K, rid-list>

Types of indexes



Clustered vs unclustered

↓ sorted by Key field

↓ unsorted

Primary

↓ index on a primary Key

Secondary

↓ any other

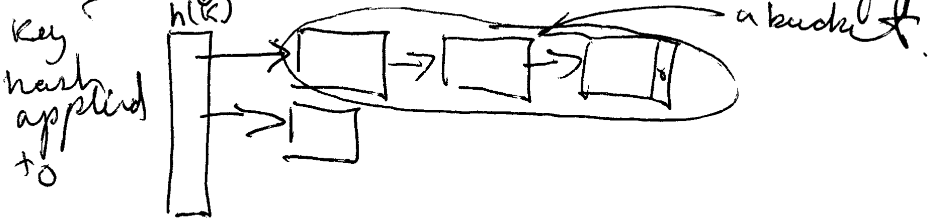
clustered index has one entry for each Key field
assume file sorted on field
with distinct blocks for field value

Data Structure Used for Indexing

Hash-Based Indexing

Records grouped in buckets by ~~a~~ hash $f^{\underline{n}}$.

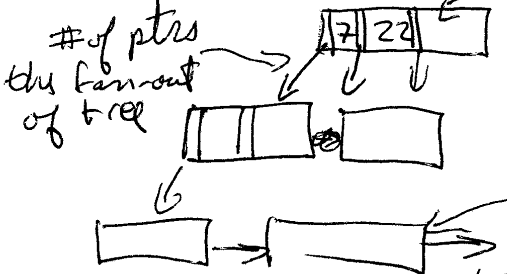
Buckets are a linked-list of pages



Tree-Based Indexing used

For ~~RBB~~ databases

Like Binary Search trees except # of ~~rec~~ records / ~~block~~ ~~[illegible]~~ many keys in a node rather just 2 ~~[illegible]~~ keys



data records ptd to key leaf of ~~tree~~

Comparisons of File Organizations

Operations will consider:

- Scan - fetch all records
- Search w/ equality
- Search w/ range selection
- Insert a record
- Delete a record

Variables

B = # of data pages (blocks) accesses

D = time to read or write a page

R = # of records / page

C - avg time to process a record] - 100 ns at time book written

H - time to calculate hash $f^{\underline{n}}$

Assumptions - no network ~~costs~~

no blocked disk accesses

Heap Files

Scan $B(D + RC)$

Search key $.5B(D + RC)$

Search Range $B(D + RC)$ — record could be anywhere in file

Insert $2D + C$ — just add to end of file

Delete ~~$B(D+RC)$~~ Search+ $C+D$ (no attempt to reclaim space ~~record id used to locate record~~ ~~allocation & add space~~.

Sorted Files

Scan $B(D+RC)$

Search key $D\log_2 B + C\log_2 R$

Search Range $D\log_2 B$ + # matches

Insert Search + $B(D+RC)$ ← read and ~~[illegible]~~ write data after insert

Delete Search + $B(D+RC)$

Clustered Files

File size typically 1.5 times usual unclustered

Scan $1.5B(D+RC)$

Search Key $D\log_{2} 1.5B + C\log_2 R$

Search w/ Range Search Key + # matches

Insert Search + D

Delete Search + D

Similar analysis for index cases