Storage and Indexing

File Organization (Heap & Sorted files)

File: A collection of **pages**, each containing a collection of **records**.

1. DBMS: support insert/delete/modify record; Read a particular record; scan all records (with some conditions)
2. Heap files: no particular order
3. Sorted Files: pages and records within pages are ordered by some condition
4. Index File Organisation

Heap (Unordered) Files

Simplest file structure, contains records in no particular order

As file grows and shrinks, disk pages are allocated and de-allocated => fastest for inserts

Sorted Files

Similar structure like heap files (pages and records), but pages and records are ordered

Fast for range queries, but hard for mainteneance: each insert potentially reshuffles records

How to choose?

Data is typically stored in pages on Hard Disks (HDD). To be able to process and analyze it-data needs to be brought to Memory (RAM).

(RAM is much more expensive than HDD)

DBMS model the cost of all operations

Cost = the number of page access (or disk I/O operations) $$1\ page\ access == 1\ I/O$$

Index Files & Indexes

Index: a data structure built on top of data pages used for efficient search. Contains a collection of data entries, and supports efficient retrieval of data records matching a given search condition.

Search Key Fields: The index is built over; **Any subset** of the fields of a relation can be the search key for an index

NOTE: Search key is not the same as key, i.e. no need to be unique

Index Classification

Clustered vs. Unclustered Index

Clustered: order of data records is the same as the order of index data entries

CLustering properties

1. A data file can have a clustered index on at most one search key combination

2. Cost of retrieving data records through index decreases greatly if index is clusted
3. Clustered indexes are more expensive to maintain (require file reorganization), but are really efficient for range search

Primary vs. Secondary Index

Primary: primary index includes the table's primary key

Secondary: is any other index

Properties

1. Primary index never contains duplicates
2. Secondary index may contain duplicates

Composite Search Keys

An index can be built over a combination of search keys

Hash-based Index

Represents index as a collection of buckets. Hash function maps the search key to the corresponding bucket.

$$h(r.search\_key) = bucket\ in\ which\ record\ r\ belongs$$

Good for equality selections.

Tree-based Index

Underlying data structure is a binary (B+) tree. Nodes contain pointers to lower levels (left for lower, right for higher). Leaves contain data entries sorted by search key values.

Good for range selections

Summary

1. Many alternative file organizations exist, each appropriate in some situation
2. if selection queries are frequent, sorting the file or building an index is important
3. Index is an additional data structure, introduced to quickly find entries with given key values. Hash-equality; Sorted files & tree-based indexes-range & equality (slower)
4. Files rarely kept sorted in practice since high cost; B+ tree index is better