


Storage and Indexing

Chapter 8


2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 1



Data on External Storage - Review

- File contains a (multi)set of records
 - Each record has a unique *RID*; sufficient to locate record on disk
- Data read and written to disk by *buffer manager*
 - Main unit of transfer is a *page*
- File made up of many pages; each page contains multiple records
 - Last class

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 2



File Organization

- Method of arranging *file of records* on disk
 - File typically stores a relation
- Many alternatives, with different strengths and weaknesses
 - *Heap (random order) files*: No particular order defined for records
 - *Sorted Files*: Records sorted based on one or more attributes
 - *Indexes*: Organize records using trees or hashing

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 3

Operations on File - Example

- Employees (Name, Age, Salary)
- Operations
 - Scan: Fetch all employees from disk
 - Equality Search: age = 21
 - Range Selection: age >= 18 AND age < 65
 - Insert a record
 - Delete a record
- *How do we organize the file to accomplish these tasks efficiently?*

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 4

Indexes

- A data structure that organizes data records on disk to optimize certain operations
- Speed up selections on the **search key** field of the index (denoted k)
 - Any subset of the fields of a relation can be the search key for an index on the relation.
 - **Search key** is **not** the same as **key** (minimal set of fields that uniquely identify a record in a relation).

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 5

Indexes

- Index contains a collection of **data entries**
- Data entry for search key k denoted k^*
- Alternative ways to store data entry k^* :
 - 1: Data entry k^* is an actual record (with search key k)
Index == File !
 - 2: Data entry k^* is (k, rid) pair, where rid is refers to a record with search key k
Actual data records stored in a different file
 - 3: Data entry k^* is $(k, \text{rid-list})$ pair, where rid-list refers to list of records with search key k
- Choice of alternative for data entries is orthogonal to the indexing technique used to locate data entries with a given key value k .
 - Examples include: B+ trees, hash-based structures

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 6

Indexes

Can search key k be a pair of values / attributes?

Can a file (relation) have more than one index?

Can a file (relation) have more than one index using data entry alternative 1?

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 7

Indexing Terminology

- **Primary vs. secondary:** If search key contains primary key, then called primary index
 - Otherwise, called secondary index
- **Clustered vs. unclustered:** If order of data records is the same as, or 'close to', order of data entries, then called clustered index.
 - Alternative 1 implies clustered, but not vice-versa.
 - Alternative 1 w/ clustered index also called a **clustered file**
 - A file can be clustered on at most one search key.
 - Cost of retrieving data records through index varies *greatly* based on whether the index is clustered or not!

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 8

Clustered vs. Unclustered Index

- Suppose: data records in heap file, index with Alt. 2
- To build **clustered** index, first sort the Heap file

Suppose you have an index on Age, and you want to do a range selection (e.g., Age >= 18)
Which index would you prefer? Why?

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 9

B+ Tree Indexes

Non-leaf Pages

Leaf Pages

- Leaf pages contain *data entries*, and are chained (prev & next)
- Non-leaf pages contain *index entries* and direct searches:

index entry

P_0	K_1	P_1	K_2	P_2	\dots	K_m	P_m
↓		↓		↓			↓

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 10

Example B+ Tree

Root

Entries < 17 Entries ≥ 17

- Find 28*? 29*? All > 15* and < 30*
- Insert/delete: Find data entry in leaf, then change it. Need to adjust parent sometimes.
 - And change sometimes bubbles up the tree

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 11

Hash-Based Indexes

- Good for equality selections.
 - Index is a collection of *buckets*. Bucket = *primary page* plus zero or more *overflow pages*.
 - *Hashing function h*: $h(r)$ = bucket in which record r belongs. h looks at the *search key* fields of r .

Primary bucket pages Overflow pages

Buckets contain:

- If Alternative (1) is used → data records
- Alternative 2 → <key, rid>
- Alternative 3 → <key, rid-list> pairs

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 12

Comparing File Organizations / Indexes - Example

- Employees (Name, Age, Salary)
- Operations
 - Scan: Fetch all employees from disk
 - Equality Search: age = 21
 - Range Selection: age >= 18 AND age < 65
 - Insert a record
 - Delete a record

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 13

Analysis of I/O Cost

- For simplicity, ignore CPU cost
- Important Factors for I/O Cost:
 - How many pages do we need to read?
 - Are I/Os sequential or non-sequential?
- Textbook contains a slightly different analysis (more detail, but ignores distinction between sequential & non-sequential I/O)

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 14

Heap File

- Scan: Read all pages in file
 - sequential
- Equality Search: Read all pages in file
 - sequential
 - *worst case*
- Range Selection: Read all pages in file
 - sequential
- Insert: Easy
- Delete: Searching + Easy

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 15

Clustered File (on Age)

- **Scan:** Slightly more than heap file due to page occupancy
- **Equality Search:** Traverse height of tree
 - non-sequential
- **Range Selection:**
 - Traverse height of tree (non-sequential)
 - Scan leaf records (sequential)
- **Insert:** Traverse height of tree (non-sequential) + write
- **Delete:** Traverse height of tree (non-sequential) + write

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 16

Heap File w/ Unclustered Tree Index (on Age)

- **Scan:** Same as heap file
- **Equality Search:** Traverse height of tree
 - Non-sequential
 - Can be more costly if many records satisfy equality
- **Range Selection:**
 - Traverse height of tree (non-sequential)
 - Scan leaf records (**non-sequential**)
 - Can be very expensive if many records satisfy range expression!!
- **Insert:** Update index + update heap file
- **Delete:** Update index + update heap file

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 17

Heap File w/ Unclustered Hash Index (on Age)

- **Scan:** Same as heap file
- **Equality Search:** Nearly constant
- **Range Selection:** Hash index no use!
 - Scan heap file
- **Insert:** Update index + update heap file
- **Delete:** Update index + update heap file

2/10/09 EECS 484: Database Management Systems, Kristen LeFevre 18



Announcements

- Optional Exercises (for Review): 8.1, 8.3, 8.11, 9.2, 9.3, 9.5, 9.7, 9.9, 9.12, 9.13, 9.14, 9.16, 9.17
- Sample Midterm posted on class website
- Exam Review -- Friday during discussion
- Midterm Exam -- Monday, in class
