File Structures

07. Indexing – Part2

2020. Spring

Instructor: Joonho Kwon

jhkwon@pusan.ac.kr

Data Science Lab @ PNU

Slides are based on the textbook.

data science laboratory

Where are we?

- Plain Stream File
- Persistency → Buffer support → BufferFile
  - <incremental approach> Deriving BufferFile using various other classes

- Random Access → Index support → IndexedFile
  - <incremental approach> : Deriving TextIndexedFile using RecordFile and TextIndex

Outline

Too Large Index (1/2)

- On secondary storage (large linear index)
- Disadvantages
  - binary searching of the index requires several seeks(slower than a sorted file)
  - index rearrangement requires shifting or sorting records on second storage

- Alternatives (to be considered later)
  - hashed organization
  - tree-structured index (e.g. B-tree)

Too Large Index (2/2)

- Advantages over the use of a data file sorted by key even if the index is on the secondary storage
  - can use a binary search
  - sorting and maintaining the index is less expensive than doing the data file
  - can rearrange the keys without moving the data records if there are pinned records

Outline

Index by Multiple Keys (1/2)

DB-Schema = (ID-No, Title, Composer, Artist, Label)

- Query samples
  - Find the record with ID-NO “COL38358” (primary key - ID-No)
  - Find all the recordings of “Beethoven” (secondary key - composer)
  - Find all the recordings titled “Violin Concerto” (secondary key - title)

Index by Multiple Keys (2/2)

- Most people don't want to search only by primary key
- Secondary Key
  - can be duplicated
- Secondary Key Index
  - secondary key -> consult one additional index (primary key index)

Composer index

Secondary Key	Primary Key
BEETHOVEN	ANG3795
BEETHOVEN	DG139201
BEETHOVEN	DG18807
BEETHOVEN	RCA2626
COREA	WAR23699
DVORAK	COL31809
PROKOFIEV	LON2312
RIMSKY-KORSAKOV	MER75016
SPRINGSTEEN	COL38358
SWEET HONEY IN THE R	FF245

$$\begin{tikzcd}\boxed{\text{BEETHOVEN}} \arrow[r] & \boxed{\text{DG18807}}\end{tikzcd}$$

Secondary Index : Basic Operations (1/3)

- Record Addition
  - similar to the case of adding to primary index
  - secondary index is stored in canonical form
    - fixed length (so it can be truncated)
    - original name can be obtained from the data file

- can contain duplicate keys
- local ordering in the same key group

Secondary Index : Basic Operations (2/3)

- Record Deletion (2 cases)
  - Secondary index references directly record
    - delete both primary index and secondary index
    - rearrange both indexes
  - Secondary index references primary key
    - delete only primary index
    - advantage : fast

Secondary Index : Basic Operations (3/3)

- Record Updating
  - Secondary index references directly record
    - update all files containing record's location
  - Secondary index references primary key
    - affect secondary index only when either primary or secondary key is changed
    - when changes the secondary key
      - rearrange the secondary key index
    - when changes the primary key
      - update all reference field
      - may require reordering the secondary index
    - when confined to other fields
      - do not affect the secondary key index

Outline

Retrieval of Records

- Types
  - primary key access
  - secondary key access
  - combination of above

- Combination of keys
  - using secondary key index, it is easy
  - boolean operation (AND, OR)

Outline

- 7.1 What is an Index?
- 7.2 A Simple Index for Entry-Sequenced Files
- 7.4 Object-Oriented Support for Indexed, Entry-Sequenced Files of Data Objects
- 7.5 Indexes That Are Too Large to Hold in Memory
- 7.6 Indexing to Provide Access by Multiple Keys
- 7.7 Retrieval Using Combinations of Secondary Keys
- 7.8 Improving the Secondary Index Structure: Inverted Lists
  - skipped
- 7.9 Selective Indexes
- 7.10 Binding

Selective Indexes

- Selective Index
  - Index on a subset of records
- Selective index contains only some part of entire index
  - provide a selective view
  - useful when contents of a file fall into several categories
    - e.g. 20 < Age < 30 and $1000 < Salary

Index Binding (1/2)

- When to bind the key indexes to the physical address of its associated record?
- File construction time binding (Tight, in-the-data binding)
  - tight binding & faster access
  - the case of primary key
  - when secondary key is bound to that time
    - simpler and faster retrieval
    - reorganization of the data file results in modifications of all bound index files

Index Binding (2/2)

- Postpone binding until a record is actually retrieved (Retrieval-time binding)
  - minimal reorganization & safe approach
  - mostly for secondary key

- Tight, in-the-data binding is good when
  - static, little or no changes
  - rapid performance during retrieval
  - mass-produced, read-only optical disk

Q&A