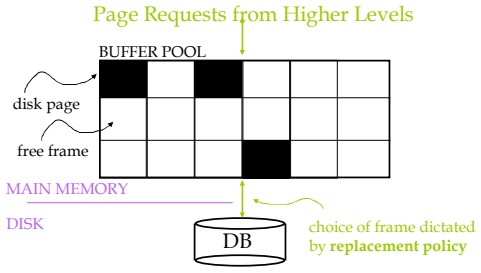
Disks and Files (Part 2)

(Chapter 9 of textbook)

Outline

- Disk technology and how to make disk read/writes faster (last time)
- Buffer management
- Storing "database files" on disk

Buffer Management in a DBMS



When a Page is Requested ...

- *If requests can be predicted (e.g., sequential scans) pages can be pre-fetched several pages at a time!*

More on Buffer Management

- Requestor of page must unpin it, and indicate whether page has been modified
- Page in pool may be requested many times
- CC & recovery may entail additional I/O when a frame is chosen for replacement.

In Class Exercise

- What happens if the buffer is full and all frames have pin count > 0?
- What happens if multiple transactions (users) want to access the same page?

In Class Exercise Solution

Buffer Replacement Policy

- Frame is chosen for replacement by a *replacement policy:*
  - Least-recently-used (LRU)
  - Clock
  - Many others
    - First-in-first-out (FIFO), Most-recently-used (MRU), Random

Buffer Replacement Policy (Contd.)

- Policy can have big impact on # of I/O's; depends on the *access pattern*.
- *Ex: Sequential flooding*

DBMS vs. OS File System

OS does disk space & buffer mgmt: why not let OS manage these tasks?

Outline

- Disk technology and how to make disk read/writes faster (last time)
- Buffer management
- Storing "database files" on disk

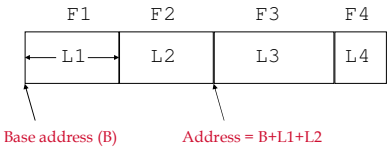
Files of Records

- Page or block is OK when doing I/O, but higher levels of DBMS operate on *records*, and *files of records*.
- FILE: A collection of pages, each containing a collection of records. Must support:

Things to decide

- Record format
- Page format
- File format

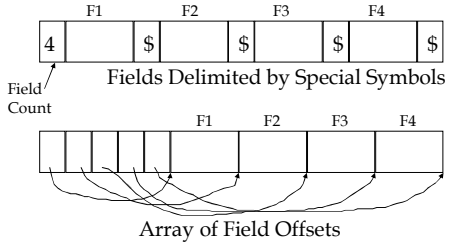
Record Formats: Fixed Length



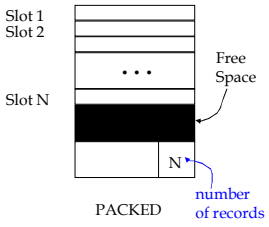
- Information about field types same for all records in a file; stored in *system catalogs.*
- Finding *i'th* field requires scan of record.

Record Formats: Variable Length

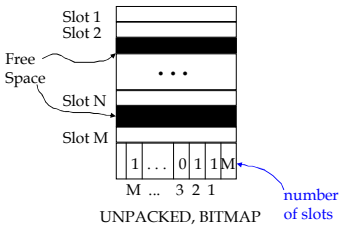
- Two alternative formats (# fields is fixed):



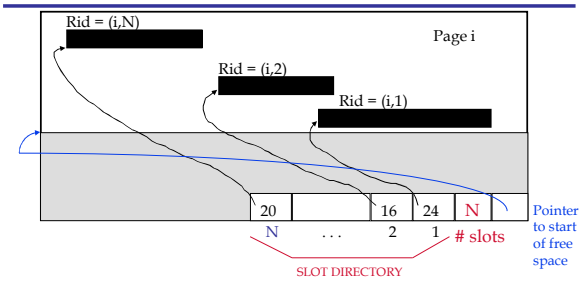
Page Formats: Fixed Length Records



Page Formats: Fixed Length Records



Page Formats: Variable Length Records

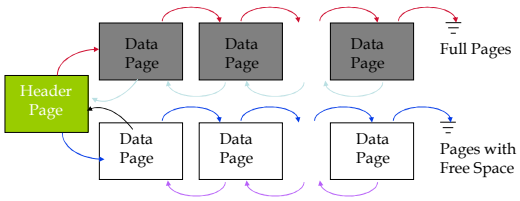


* *Can move records on page without changing rid; so, attractive for fixed-length records too.*

Unordered (Heap) Files

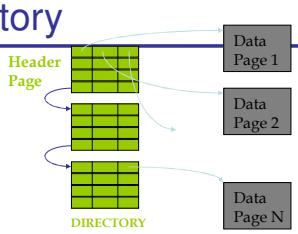
- Simplest file structure contains records in no particular order.
- As file grows and shrinks, disk pages are allocated and de-allocated.
- To support record level operations, we must:
  - keep track of the *pages* in a file
  - keep track of *free space* on pages
  - keep track of the *records* on a page
- There are many alternatives for keeping track of this.

Heap File Implemented as a List



- The header page id and Heap file name must be stored someplace.
- Each page contains 2 `pointers' plus data.

Heap File Using a Page Directory



- The entry for a page can include the number of free bytes on the page.
- The directory is a collection of pages; linked list implementation is just one alternative.
  - *Much smaller than linked list of all HF pages*!

Indexes

- A Heap file allows us to retrieve records:
  - by specifying the *rid*
    - *Usually <page id, slot number>, or some integer (need lookup table for corresponding page id and slot number)*
  - by scanning all records sequentially
- Sometimes, we want to retrieve records by specifying the *values in one or more fields*, e.g.,
  - Find all CS students with a gpa > 3
- Indexes are file structures that enable us to answer such value-based queries efficiently.

System Catalogs

- For each index:
  - structure (e.g., B+ tree) and search key fields
- For each relation:
  - name, file name, file structure (e.g., Heap file)
  - attribute name and type, for each attribute
  - index name, for each index
  - integrity constraints
- For each view:
  - view name and definition
- Plus statistics, authorization, buffer pool size, etc.

* *Catalogs are themselves stored as relations*!

Attr_Cat(attr_name, rel_name, type, position)

attr_name	rel_name	type	position
attr_name	Attribute_Cat	string	1
rel_name	Attribute_Cat	string	2
type	Attribute_Cat	string	3
position	Attribute_Cat	integer	4
sid	Students	string	1
name	Students	string	2
login	Students	string	3
age	Students	integer	4
gpa	Students	real	5
fid	Faculty	string	1
fname	Faculty	string	2
sal	Faculty	real	3

Summary

- Disks provide cheap, non-volatile storage
- Buffer manager brings pages into RAM
- DBMS vs. OS File Support
- Fixed and Variable length records
- Slotted page organization