

# CO527: Advanced Databases

## Storage and Indexing

### Lecture 01

June 16, 2016

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

# DBMS

## What, Why and When?

- What is a DBMS?
  - A piece of software to create and manage databases
- Why use a DBMS?
  - Compare with files vs DBMS
  - Features of a DBMS
- When not to use a DBMS?
  - DB is small with simple structure
  - Applications are simple, special purpose and relatively static
  - Concurrent, multi-user access to data is not required.

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

# DBMS Features

- Storing large amounts of data.
- Data independence.
- Fast access of data.
- Concurrent access to multiple users
- Security
- Reliability and Transaction Processing

# DBMS Features (Contd...)

How does a DBMS achieve its features? To understand this, we need to touch on a number of database topics.

- Data storage: Disks and Files
- Indexes
- Query Optimisation
- Transaction and Concurrency Control

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records



# Storage of Databases

- Most Databases are stored persistently on hard disks.
- This has major implications for DBMS design!
  - READ: transfer data from disk to RAM.
  - WRITE: transfer data from RAM to disk.

Both of these are high cost operations! So must be planned carefully!

# Why not store everything in main memory?

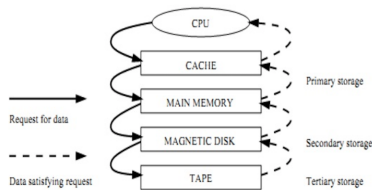
- cost and size
- Main memory is volatile: What's the problem?

# Why not store everything in main memory?

- cost and size
- Main memory is volatile: What's the problem?
- We want data to be saved between runs.

# Memory Hierarchy

- Typical storage hierarchy:
  - Cache and Main memory for currently used data (Primary storage)
  - Disks for the main database (Secondary storage)
  - Tapes for archiving older versions of the data (tertiary storage)



## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

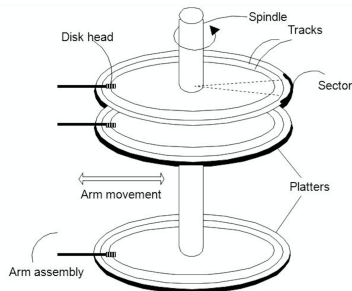
- Fixed-Length Records
- Variable-Length Records

# Storage Devices - Disks

- Secondary storage device of choice.
- Main advantage over tapes: random access vs. sequential
- Data is stored and retrieved in units called disk blocks or pages.
- Unlike RAM, time to retrieve a disk page varies depending upon location on disk.
  - Therefore, relative placement of pages on disk has major impact on DBMS performance.

# Disk Components

- The platters spin (say, 90 rps)
- The arm assembly is moved in or out to position a head on a desired track
- Tracks under heads make a *cylinder*
- *Block size* is a multiple of *sector size*



# Accessing a Disk Page

- Time to access (read/write) a disk block:
  - seek time (time required to move arms to position disk head on track)
  - rotational delay (time required to position the desired block under the head)
  - transfer time (time needed to transfer a data block)



# Disk Performance

- Seek time and rotational delay dominate.
  - Seek time is about 1-20msec
  - Rotational delay is about 0-10msec
  - Transfer rate is less than 1msec/4KB page
- Key to lower I/O cost:
  - reduce seek/rotation delays

# Arranging pages on Disk

- *Next* block concept:
  - blocks on same track, followed by
  - blocks on same cylinder, followed by
  - blocks on adjacent cylinder

# Arranging pages on Disk

- *Next* block concept:
  - blocks on same track, followed by
  - blocks on same cylinder, followed by
  - blocks on adjacent cylinder
- Blocks in a file should be arranged sequentially on disk to minimize seek and rotational delay.
- pre-fetching several pages at a time is a big win!

# Problem with Disks

- Disks are a potential bottleneck for performance
- Rate of performance increase
  - Disk access time is about 10% per year
  - Microprocessor >50% per year
- Disks are also not very reliable – Mean time to fail a disk is about 3 years.

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

# Redundant Arrays Independent Disks (RAID)

- A technology for parallelising disk access
- Redundant arrays of independent disks
- Disk Array: Arrangement of several disks that gives abstraction of a single, large disk.
  - Read/write concurrently to all disks.
- Goals: Increase performance.
- Two main techniques: Parallelism and Redundancy
  - Data striping: Data is partitioned; size of a partition is called the striping unit. partitions are distributed over several disks.
  - Redundancy: More disks  $\rightarrow$  more failures. Redundant information allows reconstruction of data if a disk fails.
- Data is distributed across the drives in one of several ways, referred to as RAID levels (0 through 6)

# Improving Performance with RAID

- Data Stripping: Higher transfer rates – bit-level or block-level
- Block-level:
  - Parallel serving for multiple independent requests decreases queuing time of I/O requests
  - Parallel serving for multiple block requests reduces response time
- The more the number of disks in an array, the larger the performance benefit.

# Improving Reliability with RAID

- A single copy of data in a disk array  $\rightarrow$  loss of reliability
- Solution: Employ redundancy

Exercise 01: What are some disadvantages employing redundancy?



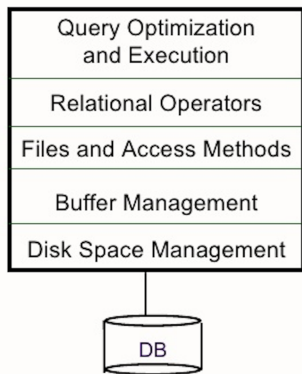
# Improving Reliability with RAID

## Employing Redundancy

- Mirroring or shadowing
  - Data is written redundantly to two identical physical disks
  - Data is read from the disk with shorter queuing, seek and rotational delays.
  - If a disk fails, the other disk is used until the first is repaired.
- Store extra information that is not normally needed.
  - Computing the redundant information can be done by using error-correcting codes (parity bits or Hamming codes)
  - Redundant information can be stored either on a small number of disks or can be distributed it uniformly across all disks.

# Structure of a DBMS

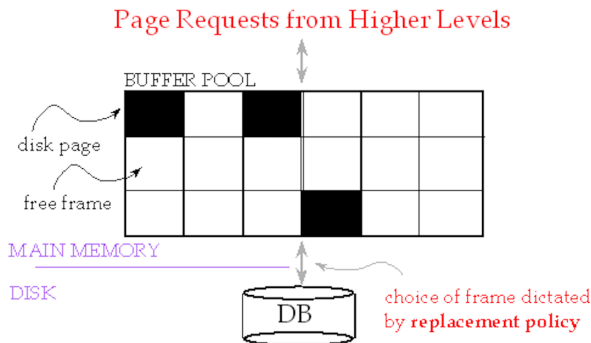
- A typical DBMS has a layered architecture.
- Lowest 4 layers must consider concurrency control and recovery.



# Disk Space Management

- Lowest layer of DBMS software manages space on disk.
- Higher levels call upon this layer to:
  - allocate/de-allocate a page
  - read/write a page
  - Page = 1 disk block
  - 1 page request = 1 disk block
- Request for a sequence of pages must be satisfied by allocating the pages sequentially on disk!

# Buffer Management in a DBMS



- Data must be in RAM for DBMS to operate on it.
- Table of  $\langle \text{frame\#}, \text{pageid} \rangle$  pairs is maintained.
- For each frame#, 2 variables (pin count: no of current users and dirty bit: page is modified)

# When a page is requested...

- If exists then done
- If requested page is not in pool:
  - Choose a frame for replacement
  - If frame is dirty, write it to disk → if the page has been modified
  - Read requested page into chosen frame
- *Pin* the page and return its address – Pin count is the number of current users

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

# Buffer Replacement Policy

- Frame is chosen for replacement by a *replacement* policy:
  - Least-Recently-Used (LRU), Clock, Most-Recently-Used (MRU), Random etc.
  - Policy can have a big impact on # of I/O's: depends on the *access pattern*.
  - *Sequential flooding* is a nasty situation caused by LRU. # *buffer frames* < # *pages in file* means each page request causes an I/O. MRU much better in this situation.

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

Several alternative exist, each ideal for some situations and not so good in others:

- Heap file –No particular order, suitable when typical access is a file scan retrieving all records.
- Ordered file –Ordered on a particular attribute (set of attributes) – Best if records must be retrieved in some order, or only a “range” of records is needed.
- Indexed file –Best for efficient searching. To be discussed later..

Which organization is best? How can you tell?



## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

# Operations to Compare

- Scan: Fetch all records from disk
- Equality Search
- Range selection
- Insert a record
- Delete a record

# Assumptions on our Analysis

- Heap Files:
  - Equality selection on key; exactly one match.
- Sorted Files:
  - Files compacted after deletions.

As a good approximation, we ignore CPU costs.

# Cost Model for our Analysis

- B: The number of data pages
- R: The number of records per page
- D: (Average) time to read or write disk page
- Measuring number of page I/O's ignores gains of pre-fetching a sequence of pages; thus, even I/O cost is only approximated.
- Average-case analysis; based on several simplistic assumptions.

# Cost of Operations

	Scan	Equality	Range	Insert	Delete
Heap					
Sorted					

# Cost of Operations

	(a) Scan	(b) Equality	(c ) Range	(d) Insert	(e) Delete
(1) Heap	BD	0.5BD	BD	2D	Search +D
(2) Sorted	BD	$D \log_2 B$	$D(\log_2 B + \text{\# pgs with match recs})$	Search + BD	Search +BD

- DBMS sees data as a collection of records
- Hence, for fixed-length records, pages can be considered as a collection of *slots*
- Each slot contains a record
- Each record contains a record id

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

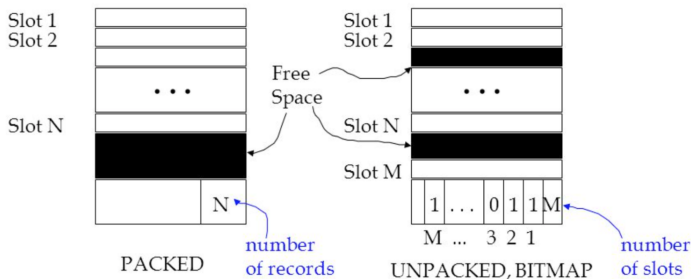
- File Formats
- Cost of Operations

## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records



# Page Formats: Fixed Length Records



- $RecordId = \langle pageId, slot \# \rangle$ . In PACKED version, moving records for free space management changes record id; may not be acceptable

# Outline

## 1 DBMS

- What, Why and When?
- DBMS Features

## 2 Data storage: Disks and Files

- Storage of Databases
- Storage Devices
- RAID

## 3 File Organisations

- File Formats
- Cost of Operations

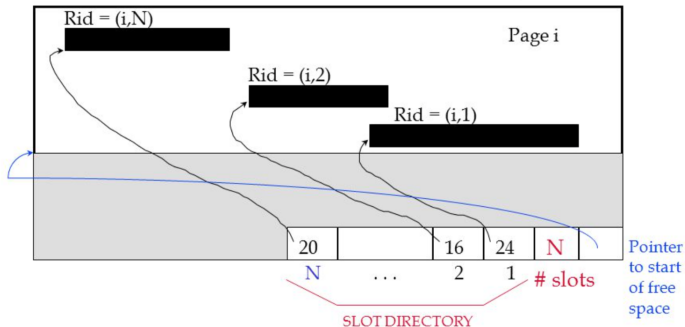
## 4 Page Formats

- Fixed-Length Records
- Variable-Length Records

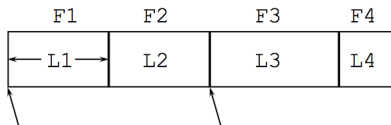
# Page Formats: Variable Length Records

- Cannot divide the page into fixed-length slots
- When a new record is to be inserted, we have to find an empty slot of just the right length.
- If not, waste space for smaller records
- In variable-length records, having contiguous free space needs is highly desirable (avoid free blocks in the middle of pages)
- So, the ability to move records on a page becomes very important

# Page Formats: Variable Length Records



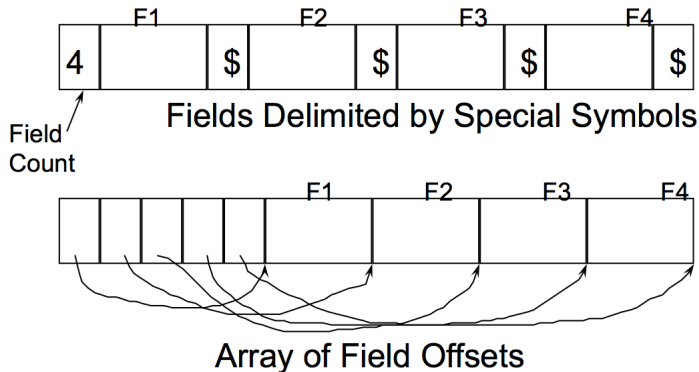
# Record Formats: Fixed Length Records



Base address (B)     $\text{Address} = B + L1 + L2$

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

# Record Formats: Variable Length Records



- Second offers direct access to i'th field.


## Exercise 02

Consider a disk with the following characteristics: block size  $B = 512$  bytes; number of blocks per track = 20; number of tracks per surface = 400. A disk pack consists of 15 double-sided disks.

- 1 What is the total capacity of a track?
- 2 How many cylinders are there?
- 3 What is the total capacity of a cylinder?
- 4 Suppose that the disk drive rotates the disk pack at a speed of 2400 rpm (revolutions per minute); what are the transfer rate (tr) in bytes/msec and the block transfer time (btt) in msec? What is the average rotational delay (rd) in msec?
- 5 Suppose that the average seek time is 30 msec. How much time does it take (on the average) in msec to locate and transfer a single block, given its block address?

# References

 Elmasri and Navathe  
*Fundamentals of Database systems 6th edition.*

 Ramakrishnan and Gehrke  
*Database Management systems.*