Introduction to the Basic Concepts

Motivation

- This course covers data processing from a computer science perspective:
  - Storage of *large amount of* data
  - Organization of data
  - Access to data
  - Processing of data

Data Structures vs File Structures

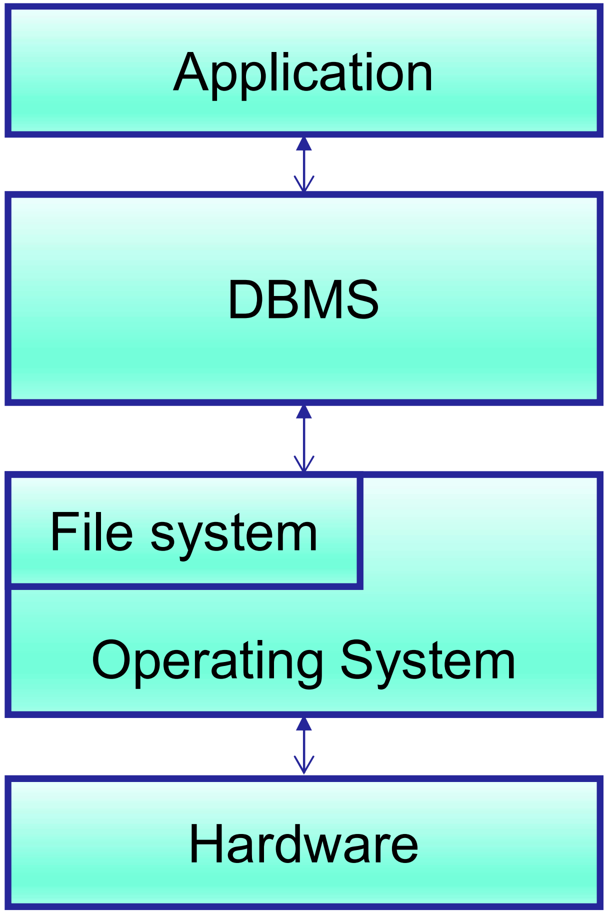
- Both involve:
  - **Representation** of Data

    +

  - Operations for **accessing** data

- Difference:
  - Data structures: deal with data in **main memory**
  - File structures: deal with data in **secondary storage**

Where do File Structures fit in Computing?



Computer Architecture

data is manipulated here

Main Memory (RAM)

- Semiconductors
- **Fast, expensive**, **volatile**, **small**

data transfer

data is stored here

Secondary Storage

- disks
- **Slow**, **cheap, stable, large**

Advantages

- Main memory is **fast**
- Secondary storage is **big** (because it is cheap)
- Secondary storage is **stable** (non-volatile) i.e. data is not lost during power failures

Disadvantages

- Main memory is **small**. Many databases are too large to fit in main memory (MM).
- Main memory is **volatile**, i.e. data is lost during power failures.
- Secondary storage is **slow** (10,000 times slower than MM)

How fast is main memory?

- Typical time for getting info from:

  Main memory: ~12 nanosec = 120 x 10^{-9} sec

  Magnetic disks: ~30 milisec = 30 x 10^{-3} sec

- An analogy keeping same time proportion as above:

  Looking at the index of a book : 20 sec

  versus

  Going to the library: 58 days

Normal Arrangement

- Secondary storage (SS) provides **reliable**, **long-term** storage for **large** volumes of data
- At any given time, we are usually interested in only a small portion of the data
- This data is loaded temporarily into main memory, where it can be rapidly manipulated and processed.
- As our interests shift, data is transferred automatically between MM and SS, so the data we are focused on is always in MM.

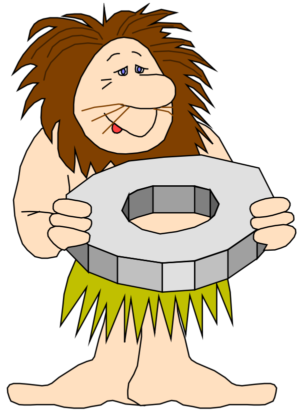
Goal of the file structures

- Minimize the number of trips to the disk in order to get desired information
- Grouping related information so that we are likely to get everything we need with only one trip to the disk.

Database

What is a database ?

Database



What is a database ?

- A collection of files storing related data.
- Models real-world enterprise (such as a university, hospital, library, etc.)

Examples of databases.

- METU’s students database, Amazon’s products database, THY airline reservation database, Isbank accounts db, Instragram postings db, Walmart payroll database

Database Management System

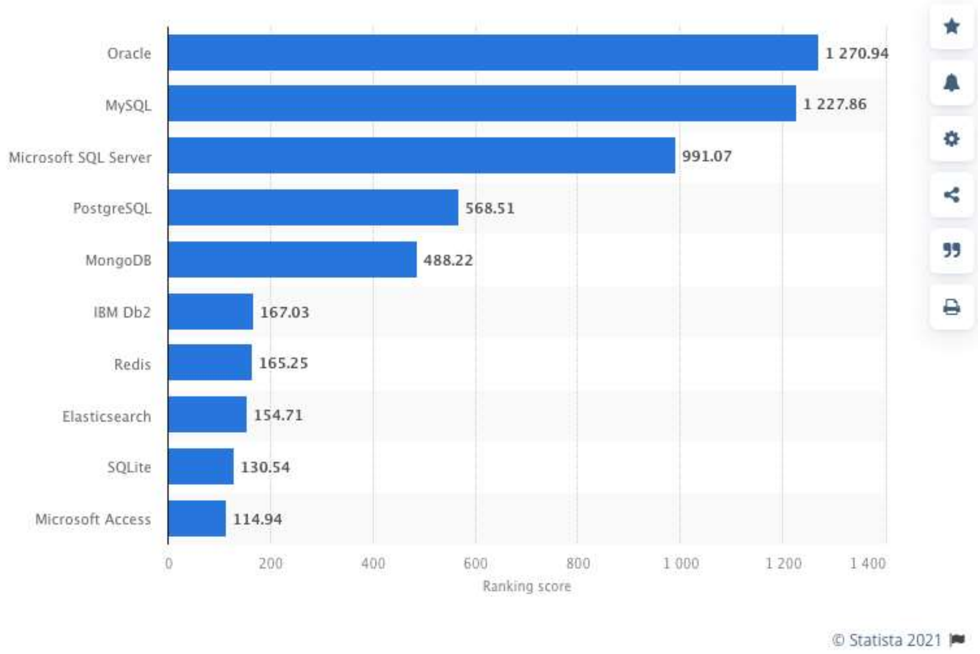
What is a DBMS ?

- A software package that allows us to **store** and **manage** efficiently a large database and allows it to persist over long periods of time.

Examples of DBMSs.

  - Oracle, IBM DB2, Microsoft SQL Server, Vertica
  - Open source: MySQL (Sun/Oracle), PostgreSQL

Database Management System



We will focus on ***Relational*** **DBMSs**

An Example: Online Bookseller

- What data do we need?
  - Data about books, customers, pending orders, order histories, trends, preferences, etc.
  - Data about sessions (clicks, pages, searches)
  - Note: data must be persistent!
  - Also note that data is large… won't fit all in memory
- What capabilities on the data do we need?
  - Insert/remove books, find books by author/title/etc.,
  - Analyze past order history, recommend books, …
  - Data must be accessed efficiently, by many users
  - Data must be safe from failures and malicious users

Required Data Management Functionality

1. Describe real-world enterprise in terms of stored data
2. Persistently store large (massive) datasets
3. Efficiently query & update
   - Must handle complex questions about data
   - Must handle sophisticated updates
   - Performance matters
4. Change structure (e.g., add attributes)
5. Concurrency control: enable simultaneous updates
6. Crash recovery
7. Security and integrity

DBMS Benefits

- Expensive to implement all these features inside the application.
- DBMS provides these features (and more)
- DBMS simplifies application development.

Key Data Management Concepts

- **Data models**: how to describe real-world data
    - Relational, XML, graph data (RDF)
- **Schema v.s. data**
- **Declarative query language**
    - Say what you want not how to get it
- **Data independence**

- **Query optimizer** and compiler
- **Transactions**: isolation and atomicity
- **Recovery**

For Ceng 352!

Relational Data Model

- The **relational model** is the most widely used model today.
  - Main concept: **relation**, basically a table with rows and columns.
  - Every relation has a **schema**, which describes the columns, or fields.
    - For a University database, schema of the relation Student:

Student(sid: string, **name**: string, **login**: string, **age**: integer, **gpa**:real)

Instance of Student Relation

Student Table

Sid	Name	Login	Age	Gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8

Example: University Database Schema

Student(**sid**: string, **name**: string, **login**: string,
age: integer, **gpa**:real)

Course(**cid**: string, **cname**:string, **credits**:integer)

Enrolled(**sid**:string, **cid**:string, grade:string)

- This is a conceptual schema.

Levels of Abstraction

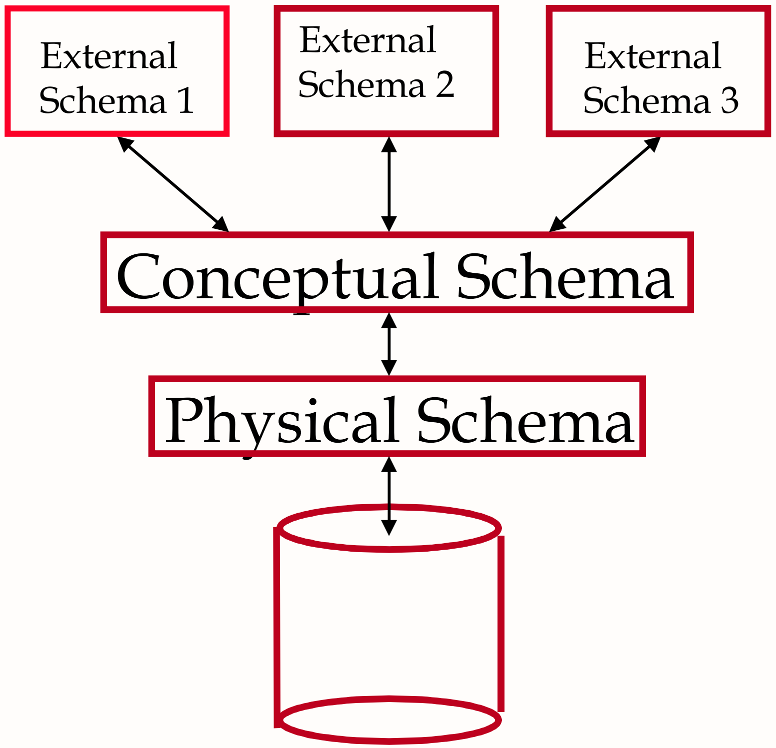
- **Conceptual Schema:** used to model the enterprise.
  - Students(sid: string, name: string, login: string, age: integer, gpa:real)
  - Courses(cid: string, cname:string, credits:integer)
  - Enrolled(sid:string, cid:string, grade:string)

- **Physical schema:** used to describe the file structures.
  - E.g. Student is a sequential file, Course is an indexed file

- **External Schema (View):** a derived schema out of the conceptual schema.
  - E.g. Course_info(cid:string,enrollment:integer)

Levels of Abstraction (cont.)

- *Many* **external schemata**, *Single* **conceptual (logical) schema** and *Single* **physical schema**.
  - External schemata describe how users see the data.
  - Conceptual schema defines logical structure
  - Physical schema describes the files and indexes used.



- Schemas are defined using a data definition language (DDL)
- Data is modified/queried using a data manipulation language (DML)

Data Independence

Applications are insulated from how data is structured and stored.

- **<u>Logical data independence</u>:** The ability to change the logical (conceptual) schema without changing the External schema (User View)
  - Can change schema w/o affecting apps
- **<u>Physical data independence</u>:** The ability to change the physical schema without changing the logical schema is.
  - Can change how data is stored on disk without maintenance to applications
- One of the most important benefits of using a DBMS!

Structure of a DBMS

- A typical DBMS has a layered architecture.
- This is one of several possible architectures; each system has its own variations.