

# Lesson 2:

# Database System Concepts & Architecture

---

CSC430/530 – DATABASE MANAGEMENT SYSTEMS

A solid blue horizontal bar at the bottom of the slide.

# OUTLINE

---

- Data model.
- Database schema and state.
- Data independence.
- DBMS languages, interfaces & system utilities.
- Centralized DBMS architecture.
- Two-tier & three-tier client/server DBMS architecture.
- DBMS classification.

# DATA MODEL: DEFINITIONS

---

- **Data model** is a set of concepts to describe:
  - **structure** of a database,
  - **operations** for manipulating these structures, and
  - certain **constraints** that the database should obey.
- Data model **structure & constraints**:
  - Database structure is defined by **constructs**.
  - Constraints specify **restrictions** on valid data that must be enforced at all times.
- Data model includes **set of operations** that govern database **retrievals** and **updates**.
  - **Basic operations**: *Insert, Delete, Update*.
  - **User-defined operations**: *compute\_student\_gpa, update\_inventory*.

# DATA MODELS: CATEGORIES

---

- **Categories of data models:**
  - **Conceptual** data models.
    - *High-level or semantic.*
    - Provide concepts that are close to the way **users perceive** data.
    - **Entities, attributes, relationships.**
  - **Implementation** data models.
    - Hide many details of **data storage**, but can be **implemented** on system directly.
    - User views and computer storage details.
  - **Physical** data models:
    - *Low-level or internal.*
    - Describe how data is **stored** as files on the computer.
    - Record formats, record orderings, and access paths.

# DATABASE SCHEMA & STATE (1)

- **Data model** consists of **two parts**:

- **Database schema** – description of the database.
  - Describes database **structure** and **constraints** that should hold on the database.
  - Specified during database **design** and **rarely** changed.
- **Database state** – snapshot of the database.
  - **Data** in the database at a particular moment in time.
  - Database **state** changes every time a **value** of **data item** in a record is **changed** or a **record** is **inserted** or **deleted**.

## STUDENT

Name	Student_number	Class	Major
------	----------------	-------	-------

## COURSE

Course_name	Course_number	Credit_hours	Department
-------------	---------------	--------------	------------

## PREREQUISITE

Course_number	Prerequisite_number
---------------	---------------------

## SECTION

Section_identifier	Course_number	Semester	Year	Instructor
--------------------	---------------	----------	------	------------

## GRADE\_REPORT

Student_number	Section_identifier	Grade
----------------	--------------------	-------

Schema Diagram of University Database

# DATABASE SCHEMA & STATE (2)

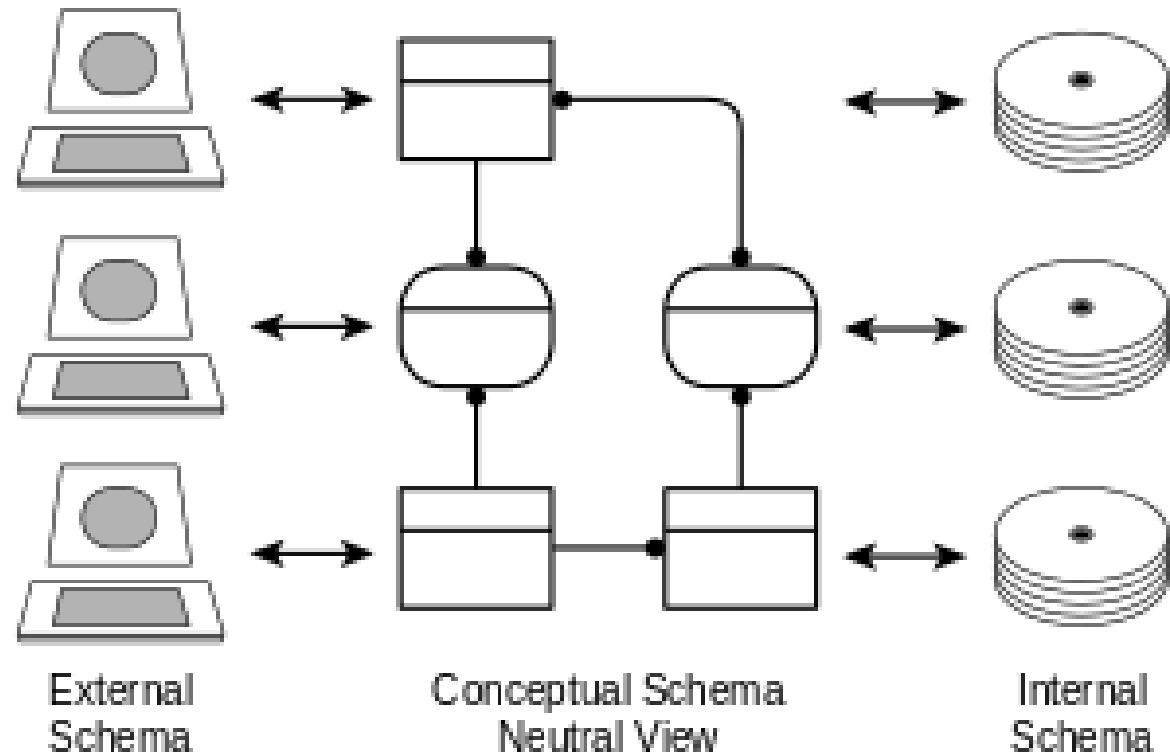
---

- **Distinction** between database **schema** & **state**:
  - The database schema changes very **infrequently** (schema evolution).
  - The database state changes **every time** the database is updated.
  - Schema = *intension*, state = *extension*.
- DBMS ensures **valid state** of database, based on the database **schema** (structure & constraints).

# THREE-SCHEMA ARCHITECTURE (1)

- **Three-schema architecture** supports database approach characteristics:

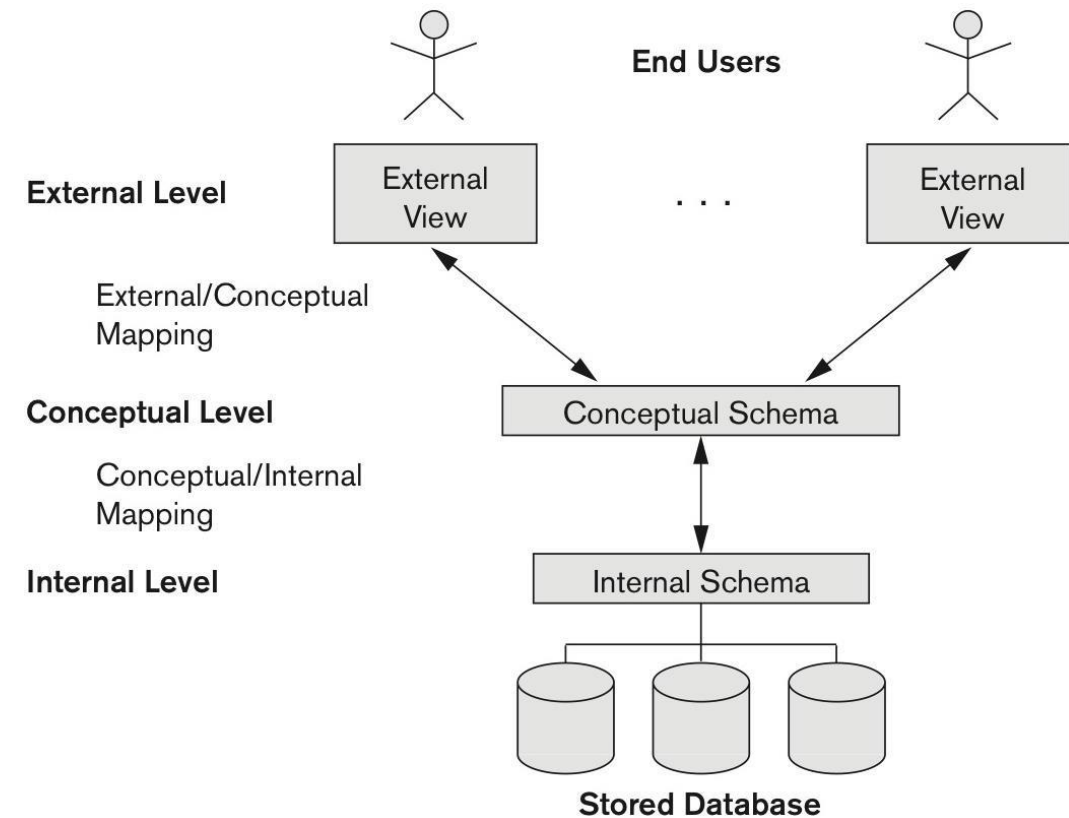
- **Self-describing** nature (DBMS catalog).
- Program-data **independence**.
- Support of data **multiple views**.



Three-Schema Architecture Simplified View

# THREE-SCHEMA ARCHITECTURE (2)

- **Schemas** are defined in three levels:
  - **Internal schema.**
    - Describes details of physical data storage & accessing paths.
    - Uses physical data model.
  - **Conceptual schema.**
    - Describes structure of the whole database to the user.
    - Uses conceptual or implementation data models.
  - **External schema.**
    - Describes the specific part of the database to a user.
    - Uses same data models as conceptual schema.
- **Mappings** among schema levels are needed to transform requests and data.

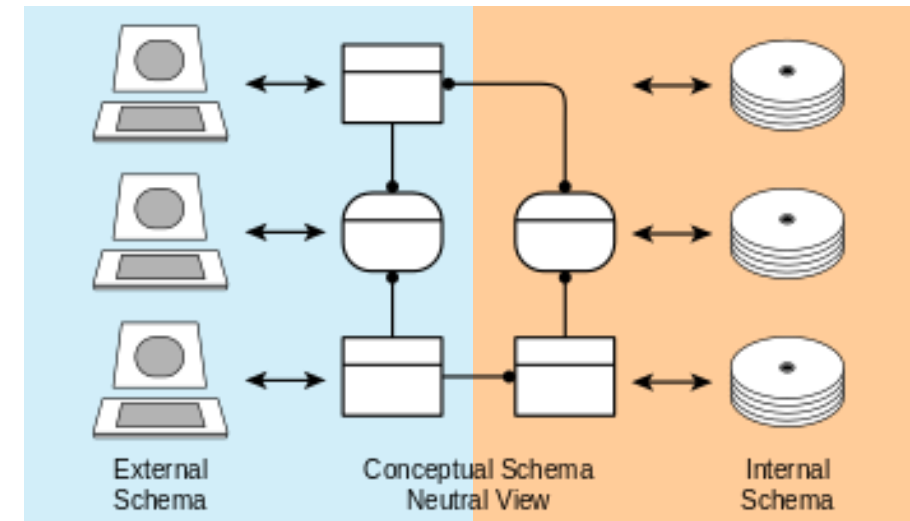


Three-Schema Architecture Detailed View



# DATA INDEPENDENCE

- **Data independence** – capacity to **change** the schema at **one level** of the database system without having to change the schema at the **next higher level**.
- **Two types** of data independence:
  - **Logical data independence:**
    - Change of **conceptual schema**  $\neq$  change of **external schemas**.
  - **Physical data independence:**
    - Change of **internal schema**  $\neq$  change of **conceptual schema**.
- **Change of schema** at a **lower-level** will only causes **change of mapping** to **higher-level** schemas.
  - **Main benefit:** no need to change application programs since they refer to the external schemas.



Three-Schema Architecture

# DBMS LANGUAGES: DDL

---

- Next step after DBMS design is the implementation of the **conceptual** and **internal** schemas and **mappings** between two.
  - Done by **data definition language** (DDL).
- **Data definition language** (DDL):
  - Used by the DBA and database designers to specify the **conceptual schema** of a database.
  - In addition, DDL is used to define **internal** and **external schemas**.

# DBMS LANGUAGES: DML

- Once the database schemas are compiled and database is populated with data, user must have means to **manipulate** the data.
  - Done by **data manipulation language** (DML).
- **Data manipulation language** (DML):
  - Used to specify database **retrievals** and **updates**.
  - **High-level** (*non-procedural*).
    - Specifies which data to retrieve, rather than how to retrieve it.
    - Operates on a set of data.
  - **Low-level** (*procedural*).
    - Specifies how to retrieve the data.
    - Operates on a single record.

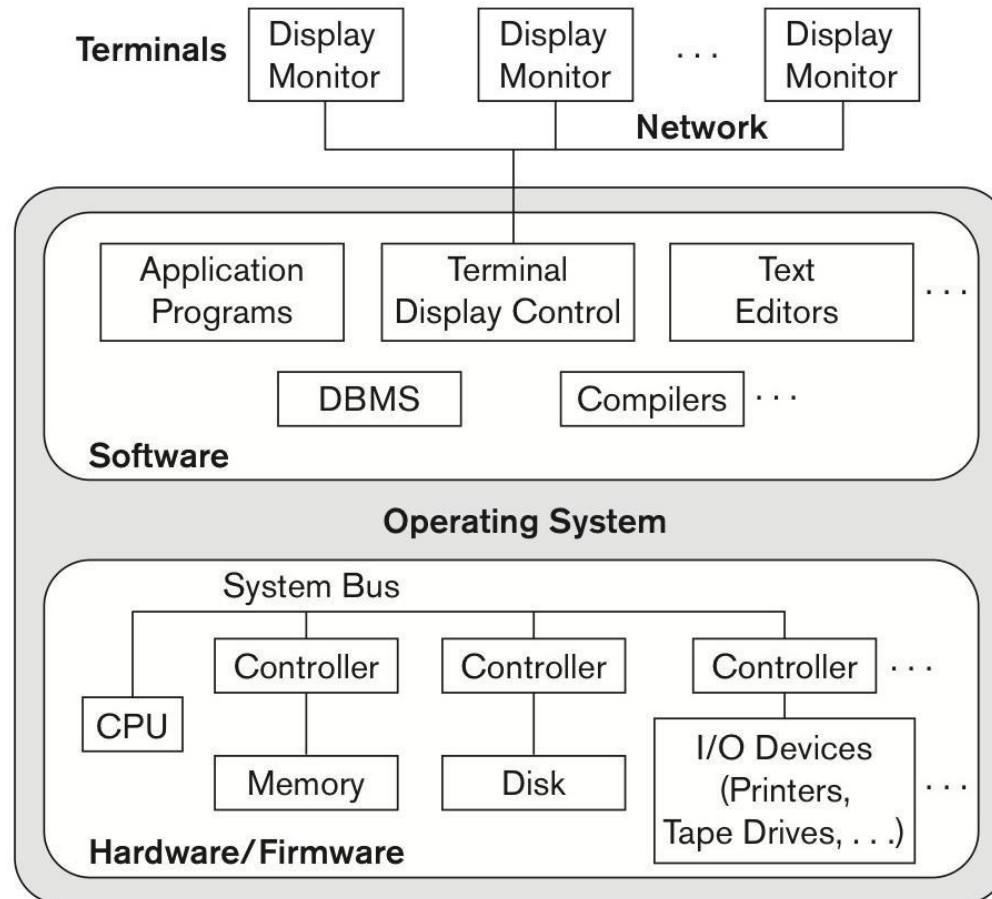
# DBMS SYSTEM UTILITIES

---

- **DBMS utilities** offer following functionality:
  - **Loading.**
    - Loading existing data files into DB or transferring data form one DB to another.
  - **Backup.**
    - Create a backup of database current state.
  - **Storage reorganization.**
    - Reorganize database files to improve performance
  - **Performance monitoring.**
    - Monitors database usage to check if reorganization is needed.
  - **Data dictionary** system.
    - Data dictionary stores meta-date, design decisions, usage standards, application program description and user information.
  - **Application development environments.**
    - Aid in database application development.

# CENTRALIZED DBMS ARCHITECTURE

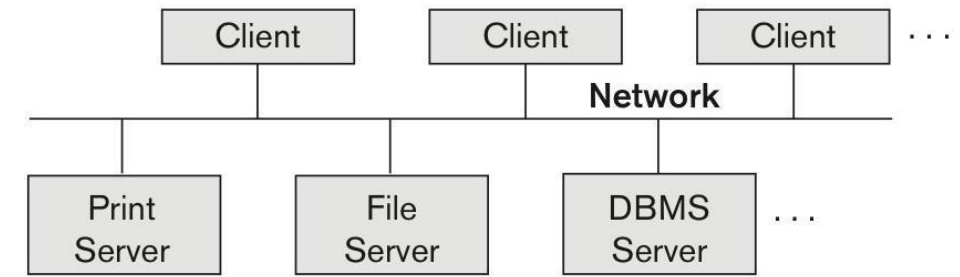
- **Centralized DBMS** architecture combines all elements into a **single system**.
  - DBMS software, hardware, application programs, and user interface processing software.



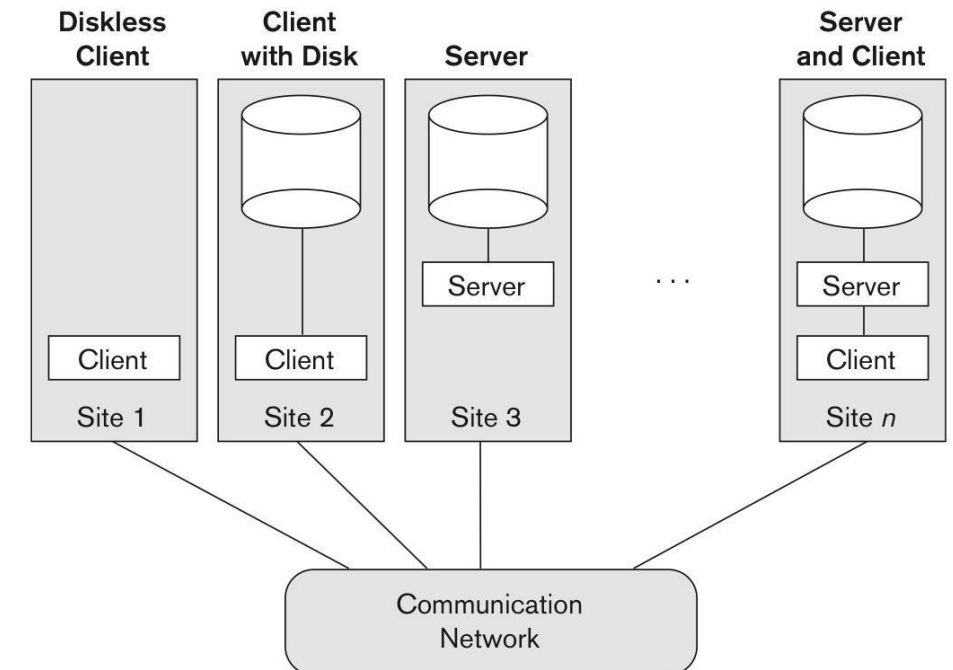
Centralized Architecture

# TWO-TIER CLIENT/SERVER ARCHITECTURE (1)

- **Basic two-tier client/server architecture:**
  - Specialized **servers** with specialized functions.
    - Print server.
    - File server.
    - **DBMS server.**
    - Web server.
    - Email server.
  - **Clients** can access the specialized servers as needed.
    - When client requires access to additional functionality (**database access**) it connects to the server.



Logical Two-Tier Client/Server Architecture



Physical Two-Tier Client/Server Architecture

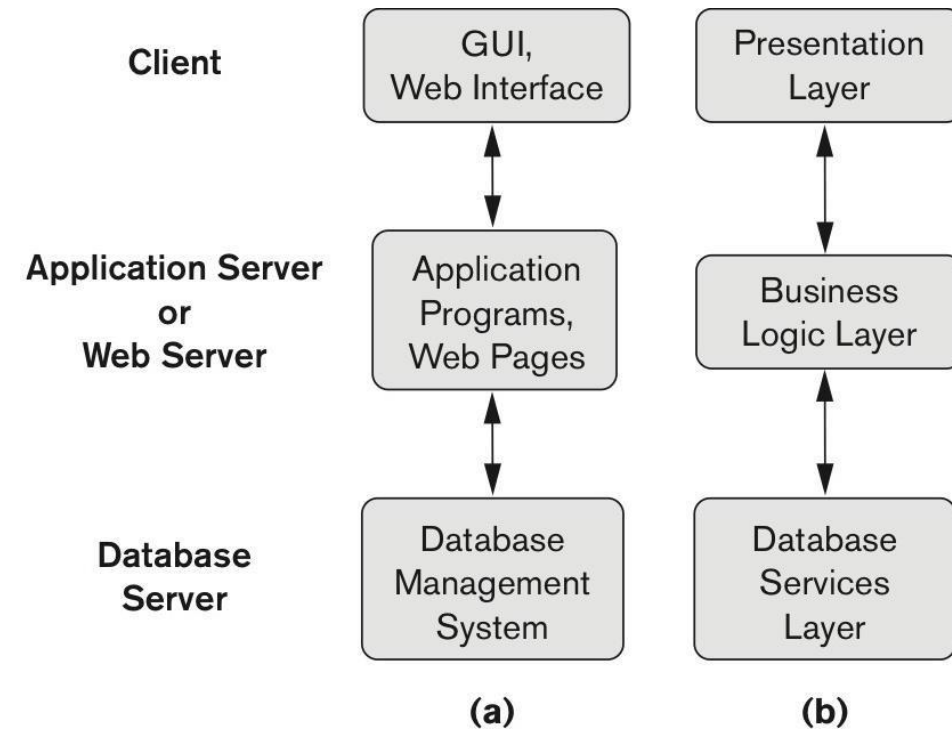
# TWO-TIER CLIENT/SERVER ARCHITECTURE (2)

---

- **Two-tier client/server DBMS architecture:**
  - **User interface** and **application programs** are stored on the **client** side.
  - **Query & transaction functionality** related to **SQL processing** is performed by the **server** (SQL server).
- When **DBMS access** required, **program** (client) establishes a connection to the **DBMS** (server).
  - Once connection is established the **client** can communicate with the **DBMS**.
  - Query results are sent back to **client** program which processes and displays the results.

# THREE-TIER CLIENT/SERVER ARCHITECTURE

- **Three-tier** is a common **architecture** for **web applications**.
  - **Intermediate layer** between **client** and **database server**.
- **Clients** → **user interface & web browsers**.
- **Middle tier** → **application server** or **web server**.
  - Runs apps or stores business rules (*procedures, constraints*) that are used to access the data from server.
  - In addition, used for security by checking client credentials.



Three-Tier Client/Server Architecture



# DBMS CLASSIFICATION

- DBMS are **classified** by following **characteristic**:

- **Data model.**

- Hierarchical.
- Network.
- **Relational.**
- Object.
- Document.
- Graph.
- Key-value.

- **Number of users.**

- Single-user.
- Multi-user.

- **Number of sites.**

- Centralized (single site).
- Distributed (multiple sites).
  - Homogeneous (same DBMS on all sites).
  - Heterogeneous (different DBMS on sites).

- **Cost.**

- Open source (free).
  - MySQL, PostgreSQL.
- Commercial (\$\$\$).

# SUMMARY

---

- Data model definition and categories.
- Database schema and state.
- Three-schema architecture and mappings.
- Logical and physical data independence.
- DBMS languages.
- Two-tier and three-tier DBMS architecture.
- DBMS classification.