TRIBHUVAN UNIVERSITY
SAMRIDDHI COLLEGE
**Lokanthali-16, Bhaktapur, Nepal**

**Bachelor of Science in**
**Computer Science & Information Technology**
**(B.Sc. CSIT)**
**Fourth Semester**

# Unit 2. *Database System-Concepts and Architecture*

by:

**Er. Bikal Adhikari**

(*B.E., M.Sc. Engg., M.Sc. ISE*)

Date:- 29th Nov., 2023

# UNIT OUTLINE

## Unit 2. Database System-Concepts and Architecture

**3 hours**

Data Models, Schemas, and Instances; Three-Schema Architecture and Data Independence; Database Languages and Interfaces; the Database System Environment; Centralized and Client/Server Architectures for DBMSs; Classification of Database Management Systems

# DATA MODELS

## ❑ Data abstraction refer to:

- Suppression of details of data organization and storage
- Highlighting of the essential features for an improved understanding of data

## ❑ Data model

- Collection of concepts that describe the structure of a database
- Provides means to achieve data abstraction
- **Basic operations**
    - Specify retrievals and updates on the database
- **Dynamic aspect** or **behavior** of a database application
    - Allows the database designer to specify a set of valid operations allowed on database objects (Compute_GPA)

# Categories of Data Models

- ❑ **High-level** or **conceptual data models**
  - – Close to the way many users perceive data
- ❑ **Low-level** or **physical data models**
  - – Describe the details of how data is stored on computer storage media (for Computer specialists)
- ❑ **Representational data models**
  - – Easily understood by end users
  - – Also similar to how data organized in computer storage

## ❑ Conceptual Data Models Use Concepts Such As

- **Entity**
  - Represents a real-world object or concept
    (an Employees)
- **Attribute**
  - Represents some property of interest
  - Further describes an entity (Employees' names or Salary)
- **Relationship** among two or more entities
  - Represents an association among the entities
    (A works-on relationship between an employee and a project)
  - **Entity-Relationship model**

❑ **Representational data models is used most frequently in traditional commercial DBMSs.**

- **Relational data model**
  - Used most frequently in traditional commercial DBMSs
- **Object data model**
  - New family of higher-level implementation data models
  - Closer to conceptual data models

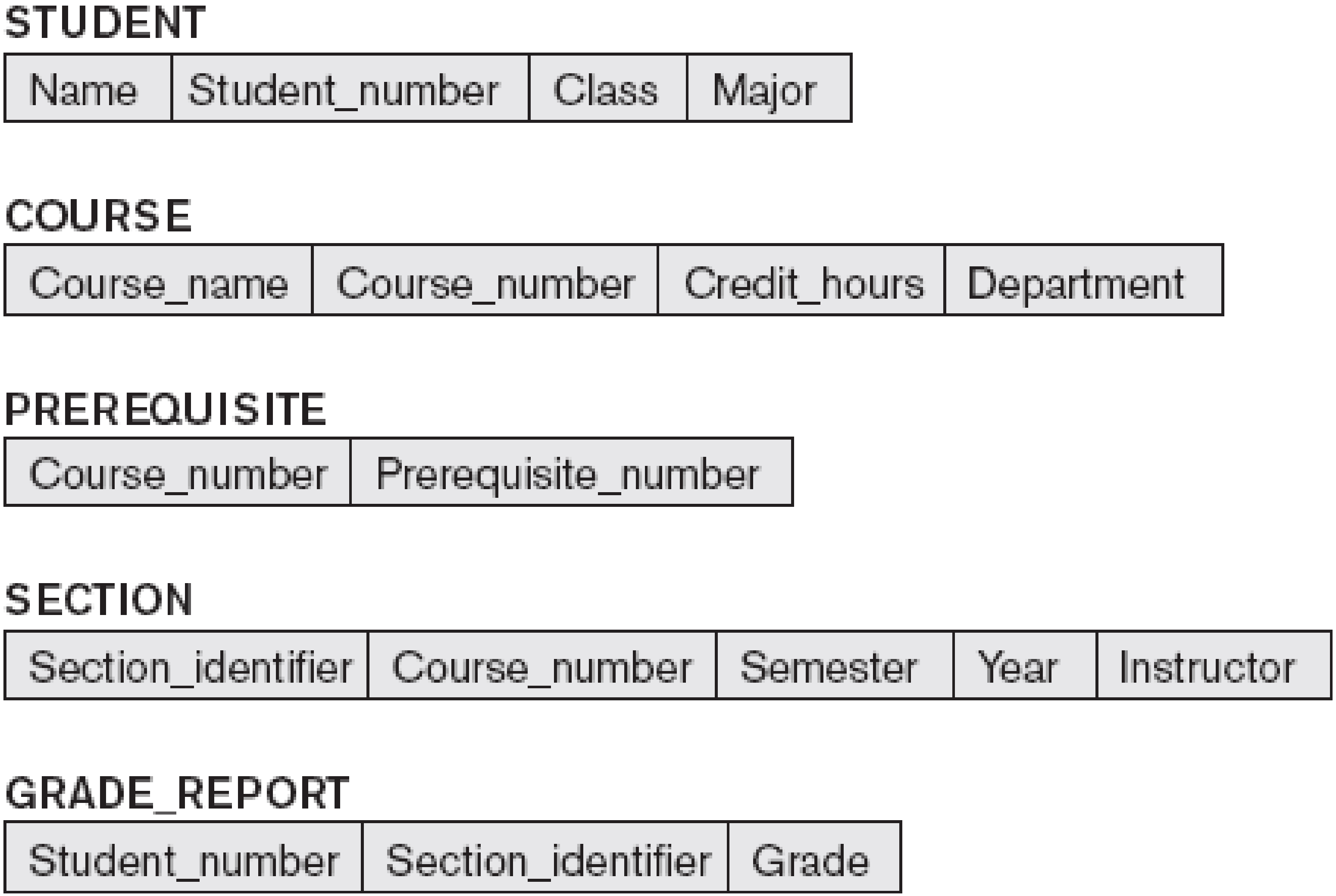
## ❑ Physical data models

- Describe how data is stored as files in the computer , (such as record format, record ordering and access path)
- **Access path**
  - Structure that makes the search for particular database records efficient
- **Index (Example of An Access Path)**
  - Allows direct access to data using an index term or a keyword (Index of any Books)

# Schemas, Instances or Database State

- **Database Schema:** The *description* of a database. It includes descriptions of the database structure and the constraints that should hold on the database.
- **Schema Diagram**: A diagrammatic display of (some aspects of) a database schema.
- **Schema Construct:** A component of the schema or an object within the schema, e.g., STUDENT, COURSE.
- **Database Instance:** The actual data stored in a database at a particular moment in time. Also called database state (or occurrence).

- ❑ **Distinction**
  - – The ***database schema*** changes very infrequently.
  - – The ***database state*** changes every time the database is updated.
- ❑ **Schema** is also called **intension**.
- ❑ **State** is also called **extension**.



**STUDENT**

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

**COURSE**

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

**PREREQUISITE**

| Course_number | Prerequisite_number |
|---|---|

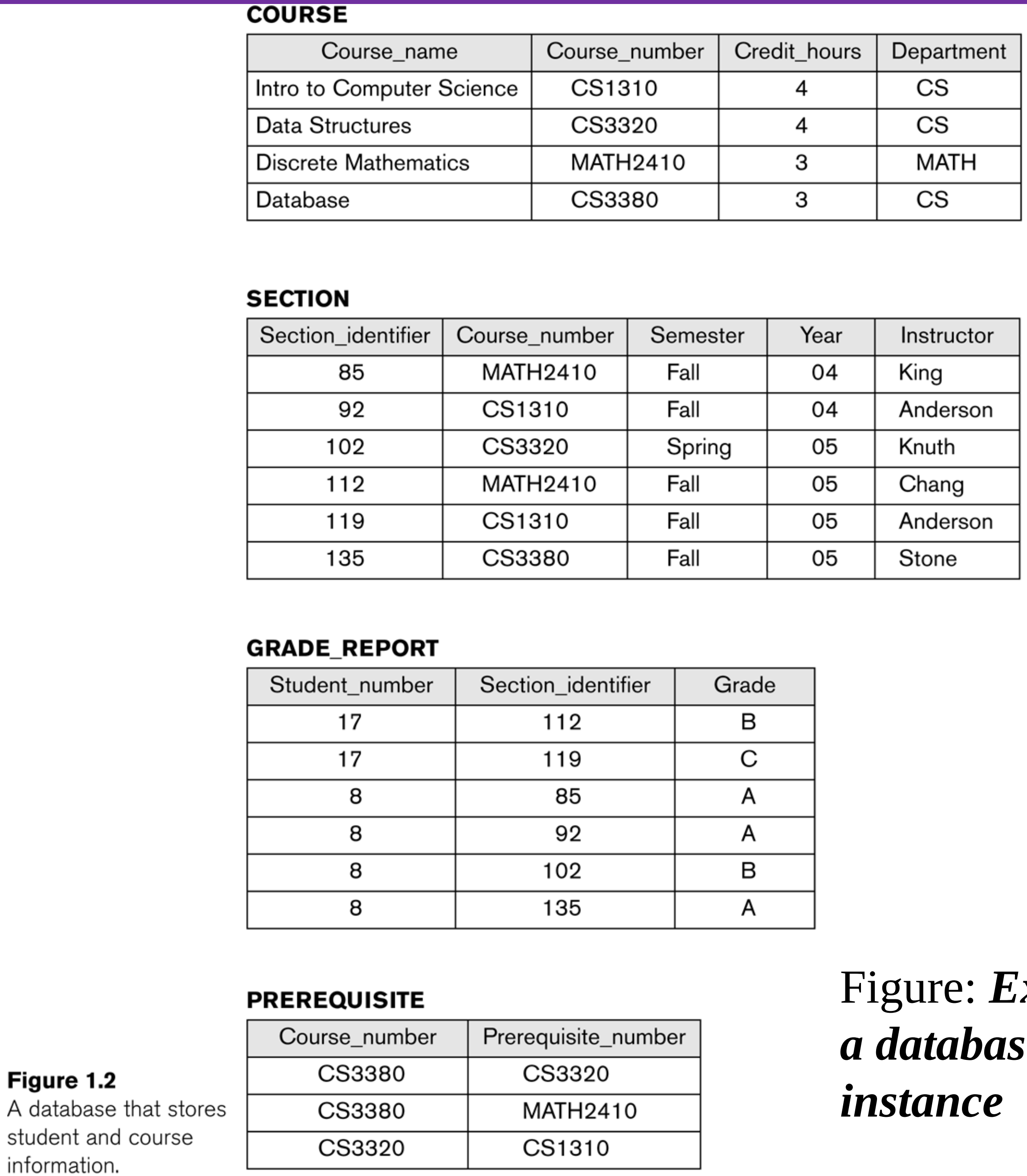
**SECTION**

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

**GRADE_REPORT**

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

Figure: ***Example of database schema***



**COURSE**

| Course_name | Course_number | Credit_hours | Department |
|---|---|---|---|
| Intro to Computer Science | CS1310 | 4 | CS |
| Data Structures | CS3320 | 4 | CS |
| Discrete Mathematics | MATH2410 | 3 | MATH |
| Database | CS3380 | 3 | CS |

**SECTION**

| Section_identifier | Course_number | Semester | Year | Instructor |
|---|---|---|---|---|
| 85 | MATH2410 | Fall | 04 | King |
| 92 | CS1310 | Fall | 04 | Anderson |
| 102 | CS3320 | Spring | 05 | Knuth |
| 112 | MATH2410 | Fall | 05 | Chang |
| 119 | CS1310 | Fall | 05 | Anderson |
| 135 | CS3380 | Fall | 05 | Stone |

**GRADE_REPORT**

| Student_number | Section_identifier | Grade |
|---|---|---|
| 17 | 112 | B |
| 17 | 119 | C |
| 8 | 85 | A |
| 8 | 92 | A |
| 8 | 102 | B |
| 8 | 135 | A |

**PREREQUISITE**

| Course_number | Prerequisite_number |
|---|---|
| CS3380 | CS3320 |
| CS3380 | MATH2410 |
| CS3320 | CS1310 |

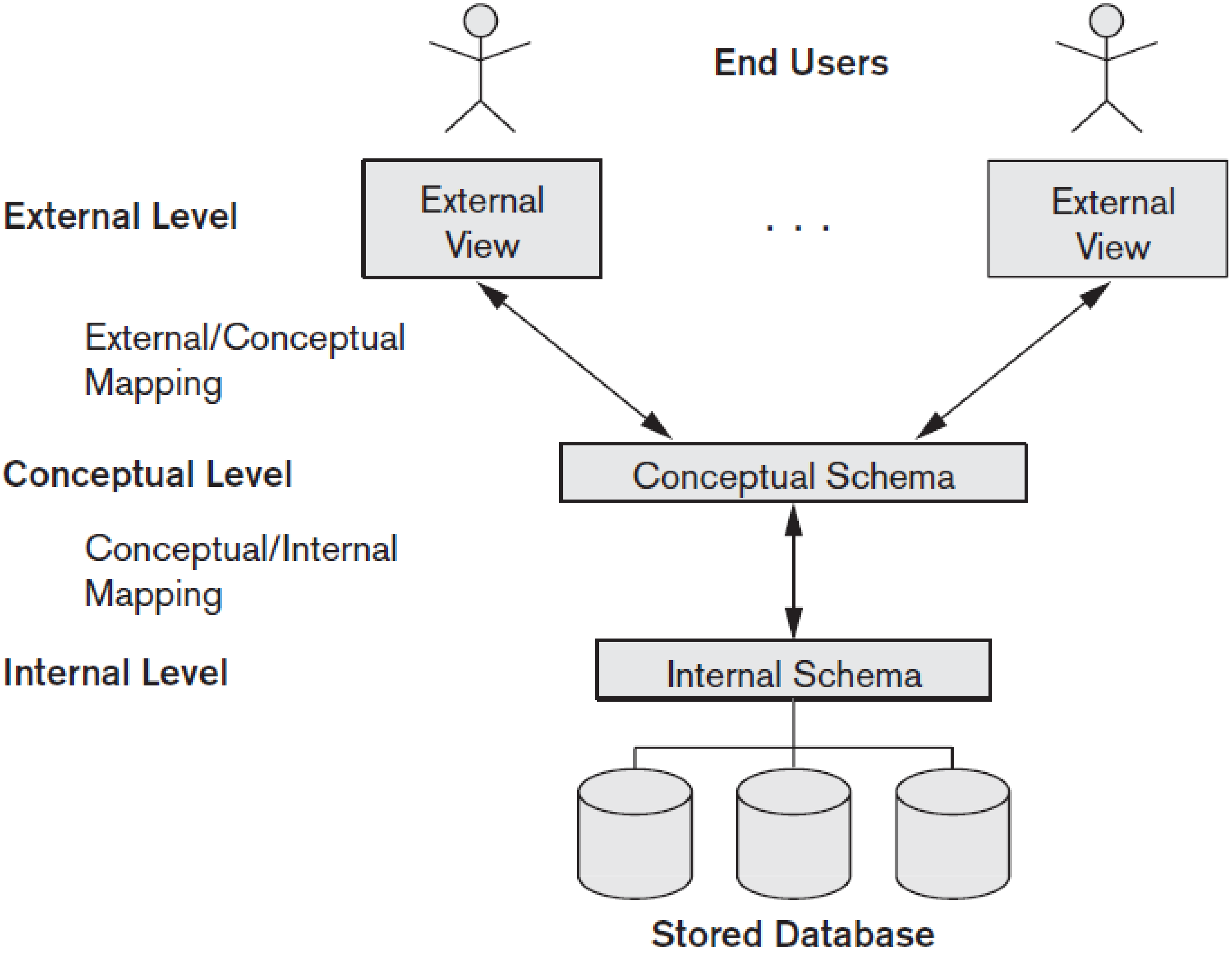
**Figure 1.2**
A database that stores student and course information.

Figure: ***Example of a database state or instance***

# Three-Schema Architecture and Data Independence

- ❑ **Internal level (Internal Schema)**
  - – Describes physical storage structure of the database
- ❑ **Conceptual level (Conceptual Schema)**
  - – Describes structure of the whole database for a community of users
- ❑ **External** or **view level**
  - – Describes part of the database that a particular user group is interested in

**Figure 2.2**
The three-schema architecture.



# DEFINE DATA INDEPENDENCE

- Data Independence: Capacity to change the schema at one level of a database system without having to change the schema at the next higher level
- Types:
  - **Logical** Data Independence:

    Capacity to change the conceptual schema without having to change external schema or application program.
  - **Physical** Data Independence:
  - **Capacity to change the internal schema without having to change conceptual schema.**

# DBMS LANGUAGES

## ❑ Data Definition Language (DDL)

- DDL stands for Data Definition Language.
- It is used to define database structure or pattern.
- It is used to create  schema, tables, indexes, constraints etc in the database.
- Using the DDL statements, we can create the skeleton of the database.
- It is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints etc.
- Here are some of the tasks that comes under DDL:
  - ○ Create: It is used to create objects in the database.
  - ○ Alter: It is used to alter the structure of the database
  - ○ Drop: It is used to delete objects from the database.
  - ○ Truncate: It is used to delete table in database instance.
  - ○ Rename: It is used to rename an object.
  - ○ Comment: It is used to comment on the data dictionary.
- Above commands are used to update database schema, that why they come under Data Definition Language.

## ❑ Data Manipulation Language (DML)

- DML stands for Data Manipulation Language.
- It is used for accessing and manipulating data in a database.
- Database language that enables insert, delete, update and retrieve data from database is called data manipulation language(DML).
- Query language is only a part of data manipulation language.
- DML statements are converted into equivalent low level statement by DML compiler.
- Structured Query Language supports following DML statements:
  - Select: It is used to retrieve data from tables.
  - Insert: It is used to insert data into a table.
  - Update: It is used to update existing data within a table.
  - Delete: It is used to delete records from a table.

## ❑ Data Control Language (DCL)

- DCL stands for Data Control Language.
- It is used to retrieve the stored or saved data.
- The DCL execution is transactional. It also has rollback parameters.
- But in oracle database the execution of DCL does not have the feature of rolling back.
- Here are some tasks that comes under DCL
  - Grant: It is used to give user access privilege to a database.
  - Revoke: It is used to take back permissions from the user.

## ❑ Transaction Control Language (TCL)

- TCL stands for Transaction Control Language.
- TCL commands are used to manage transactions in a database.
- These commands are used to manage the changes made to the data in a table by DML statements.
- It also allows statements to be grouped together into logical transactions.
- Here are some tasks that comes under TCL:
  - Commit: It is used to save transaction on the database.
  - Rollback: It is used to restore the database to original since the last commit.

# DBMS INTERFACES

- ❑ **Menu-based interfaces for Web clients or browsing:**
  Present the user with lists of options(Menus), Pull-Down menus.
- ❑ **Forms-based interfaces:**
  Displays a form to each user, for naïve users.
- ❑ **Graphical user interfaces(GUI):**
  - Utilizes both Menus and Forms.
- ❑ **Natural language interfaces:**
  Accepts requests written in English.
- ❑ **Speech input and output:**
  Limited use of speech as an input query and speech as an answer to a question, inquiries for telephone directory, flight arrival.
- ❑ **Interfaces for parametric users:**
  Bank tellers have some set of operations.
- ❑ **Interfaces for the DBA**

# The Database System Environment

- **DBMS component modules**
  - Buffer management
  - Stored data manager
  - DDL Compiler
  - Interactive query interface
    - Query compiler
    - Query optimizer
  - Precompiler
  - Runtime database processor
  - System catalog
  - Concurrency control system
  - Backup and recovery system

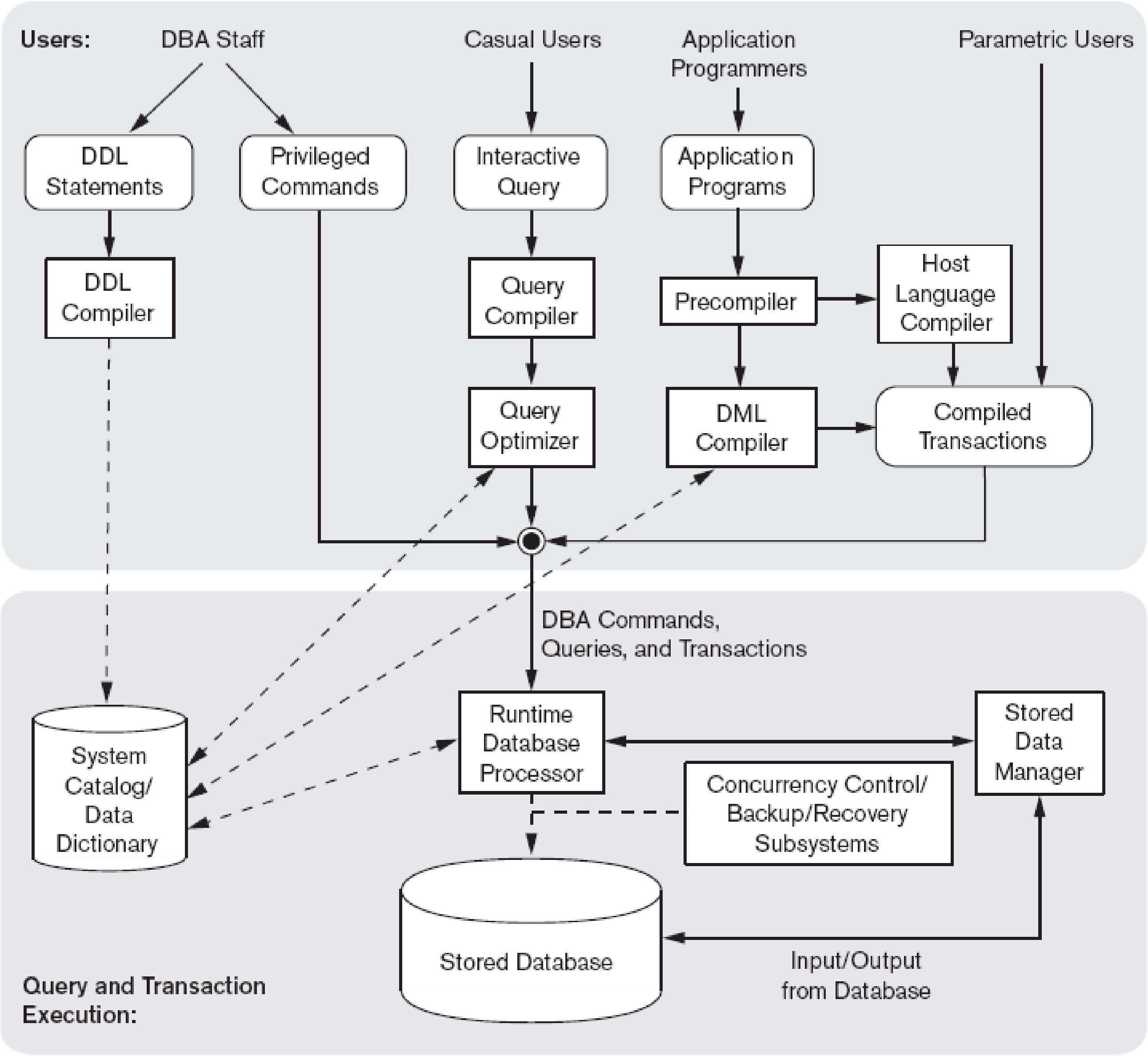


**Figure 2.3**
Component modules of a DBMS and their interactions.

# Database System Utilities

❑ **Helps DBA manage the database**

- **Loading:** Load existing data files
- **Backup:** Creates a backup copy of the database

**Database storage reorganization**

– Reorganize a set of database files into different file organizations

- **Performance monitoring**

– Monitors database usage and provides statistics to the DBA

# Tools, Application Environments, and Communications Facilities

- ❑ **CASE Tools**
  - – CASE (Computer-Aided Software Engineering) packages are software packages that include many tools that can be helpful when it comes to database design.
- ❑ **Data dictionary (data repository) system**
  - – Stores design decisions, usage standards, application program descriptions, and user information
- ❑ **Application development environments**

  Providing a tools to create a database.
- ❑ **Communications software**
  - – Allow for remote access to DB

# Centralized and Client/Server Architecture for DBMSs

## 1. Centralized DBMSs Architecture

– combines everything into single system including- DBMS software, hardware, application programs and user interface processing software.

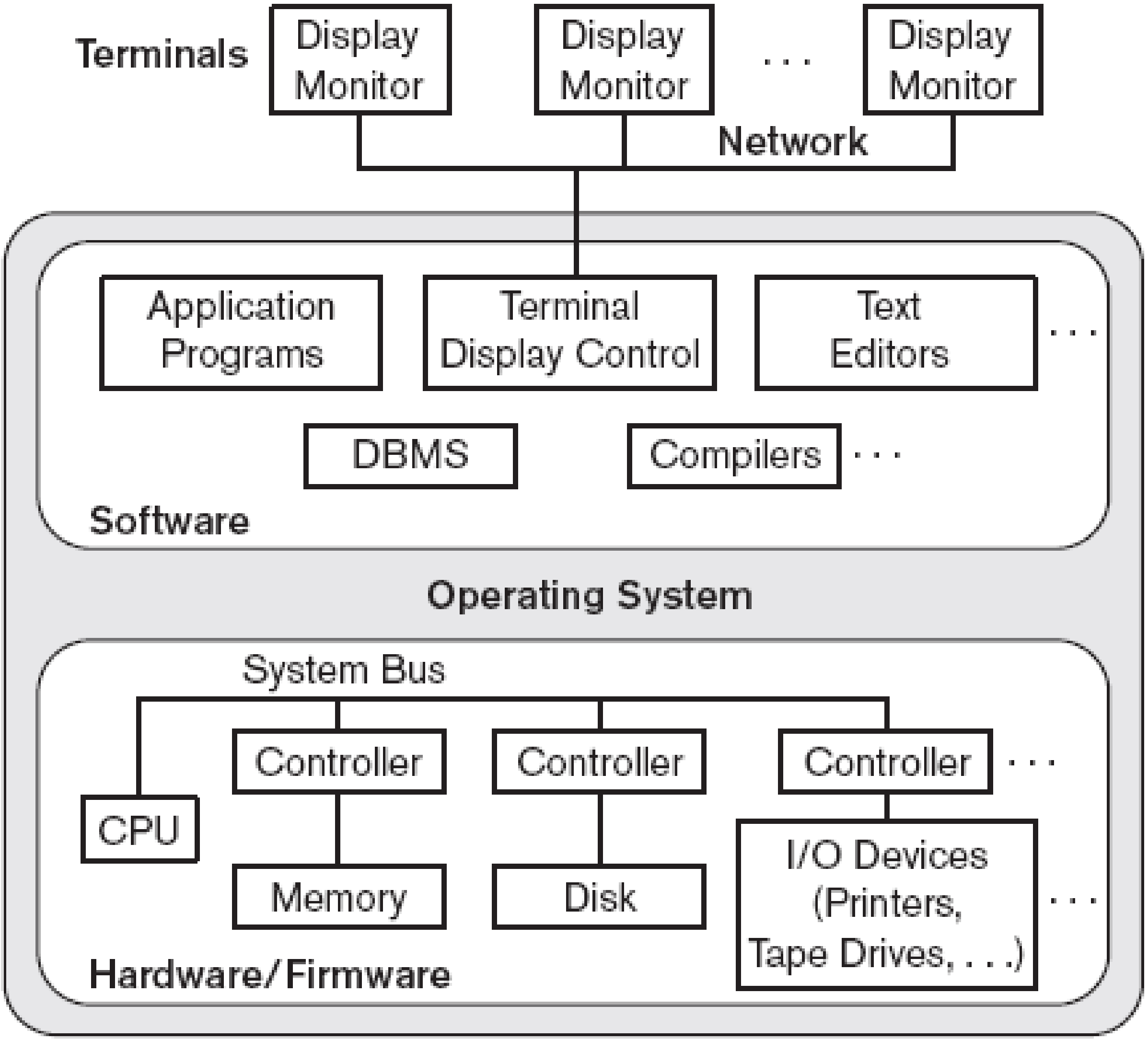


**Figure 2.4**
A physical centralized architecture.

- ❑ **Servers with specific functionalities**
  - **File server**
    - Maintains the files of the client machines.
  - **Printer server**
    - Connected to various printers; all print requests by the clients are forwarded to this machine
  - **Web servers** or **e-mail servers**
- ❑ **Client machines**
  - Provide user with:
    - Appropriate interfaces to utilize these servers
    - Local processing power to run local applications

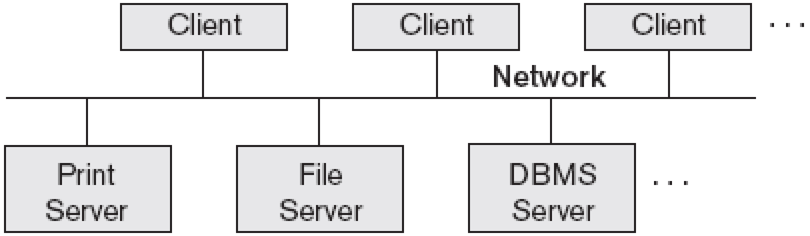


**Figure 2.5**
Logical two-tier client/server architecture.

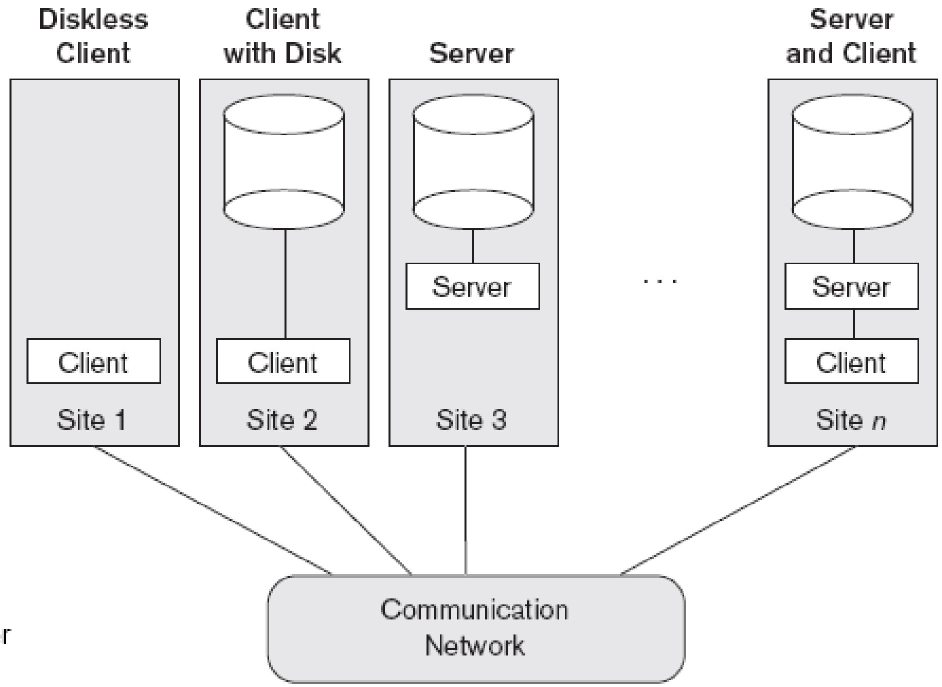


**Figure 2.6**
Physical two-tier client/server architecture.

- **Client**
  - User machine that provides user interface capabilities and local processing
- **Server**
  - System containing both hardware and software
  - Provides services to the client machines
    - Such as file access, printing, archiving, or database access

# Two-Tier Client/Server Architectures for DBMSs

- Server handles
  - Query and transaction functionality related to SQL processing
- Client handles
  - User interface programs and application programs

# Two-Tier Client/Server Architectures (cont'd.)

- ❑ **Open Database Connectivity (ODBC)**
  - – Provides application programming interface (API)
  - – Allows client-side programs to call the DBMS
    - Both client and server machines must have the necessary software installed
- ❑ **JDBC(Java$^{TM}$ database connectivity)**
  - – Allows Java client programs to access one or more DBMSs through a standard interface

# Three-Tier and n-Tier Architectures for Web Applications

- **Application server** or **Web server**
  - Adds intermediate layer between client and the database server
  - Runs application programs and stores business rules
- N-tier
  - Divide the layers between the user and the stored data further into finer components

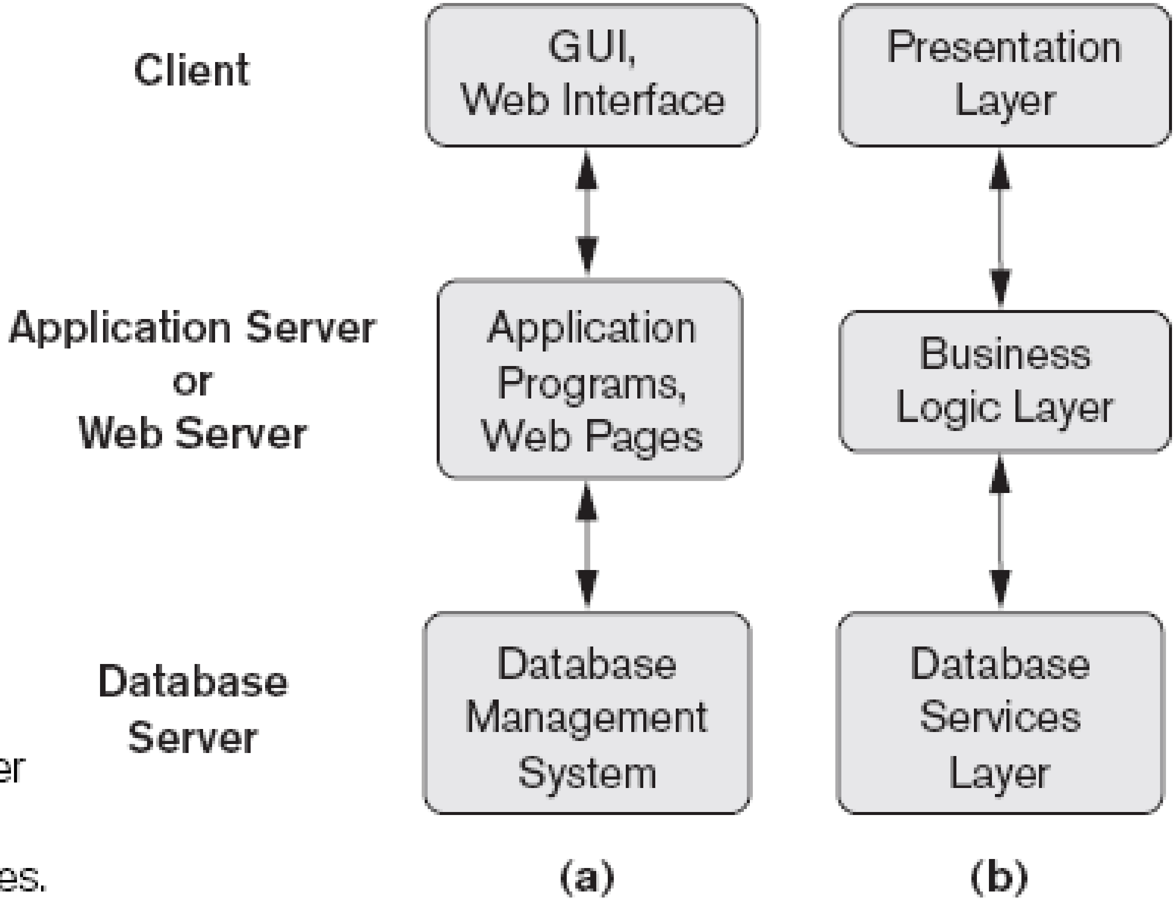


**Figure 2.7**
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

# CLASSIFICATION OF DBMS

## 1. Data model

- **Relational**
- **Object**
- **Hierarchical** and **network** (legacy)
- **Native XML DBMS**

## 2. Number of users

- **Single-user (Same time)**
- **Multiuser (Same time)**

## 3. Number of sites

- **Centralized**
- **Distributed**
  - **Homogeneous (Same DBMS software at multiple site)**
  - **Heterogeneous**

## 4. Cost

- Open source (Free: MYSQL)
- Different types of licensing

# SUMMARY

- ❑ Concepts used in database systems
- ❑ Main categories of data models
- ❑ Types of languages supported by DMBSs
- ❑ Interfaces provided by the DBMS
- ❑ DBMS classification criteria:
  - – Data model, number of users, number of sties, access paths, cost

# *Thank You !!!*