CST 204 Database Management Systems

B.Tech. CSE

Semester IV

Viswajyothi College of Engineering and Technology

Sindhu Jose, AP, CSE Dept

MODULE 1

Introduction & Entity Relationship (ER) Model

Sindhu Jose, AP, CSE Dept

Syllabus of Module 1

- Concept & Overview of Database Management Systems (DBMS). Characteristics of Database system,
- ▫ Database Users, structured, semi-structured and unstructured data. Data Models and Schema - Three Schema architecture. Database Languages, Database architectures and classification.
- ▫ ER model - Basic concepts, entity set & attributes, notations, Relationships and constraints,
- ▫ cardinality, participation, notations, weak entities, relationships of degree 3.

Data, Database & DBMS

- **Data**

  Known facts that can be recorded and have implicit meaning

- **Database**

  The collection of data

- **Database-management system (DBMS)**

  - is a collection of interrelated data and a set of programs to access those data.
  - General purpose software system that facilitates process of defining, constructing, manipulating, and sharing database

- The primary goal of a DBMS is to provide a way to store and retrieve database information that is both convenient and efficient
- Database systems are designed to manage large bodies of information.
- Management of data involves both storage of information and mechanisms for manipulation of information.
- The database system must ensure the safety of the information stored
- If data are to be shared among several users, the system must avoid possible anomalous results.

Database is divided in to **Traditional database** and **Modern database**

- **Traditional database** – has structured formatted data ie simple numeric or string data

  Eg: tables, relational database etc

- **Modern database** – stores images, audioclips, videostreams, geological information system(GIS) ,medical images etc

  Eg: Cassandra, Hadoop, Mongo DB.

Database implicit properties

- Universe of discourse(UoD) or Miniworld
  - Database represent some aspects of real world sometimes called miniworld or UoD
  - Changes to miniworld affects database
- A database is a logically coherent collection of data with some inherent meaning
- A database is designed, built and populated with data for specific purpose

Concept and Overview of DBMS

DBMS is a general purpose software system that facilitates process of

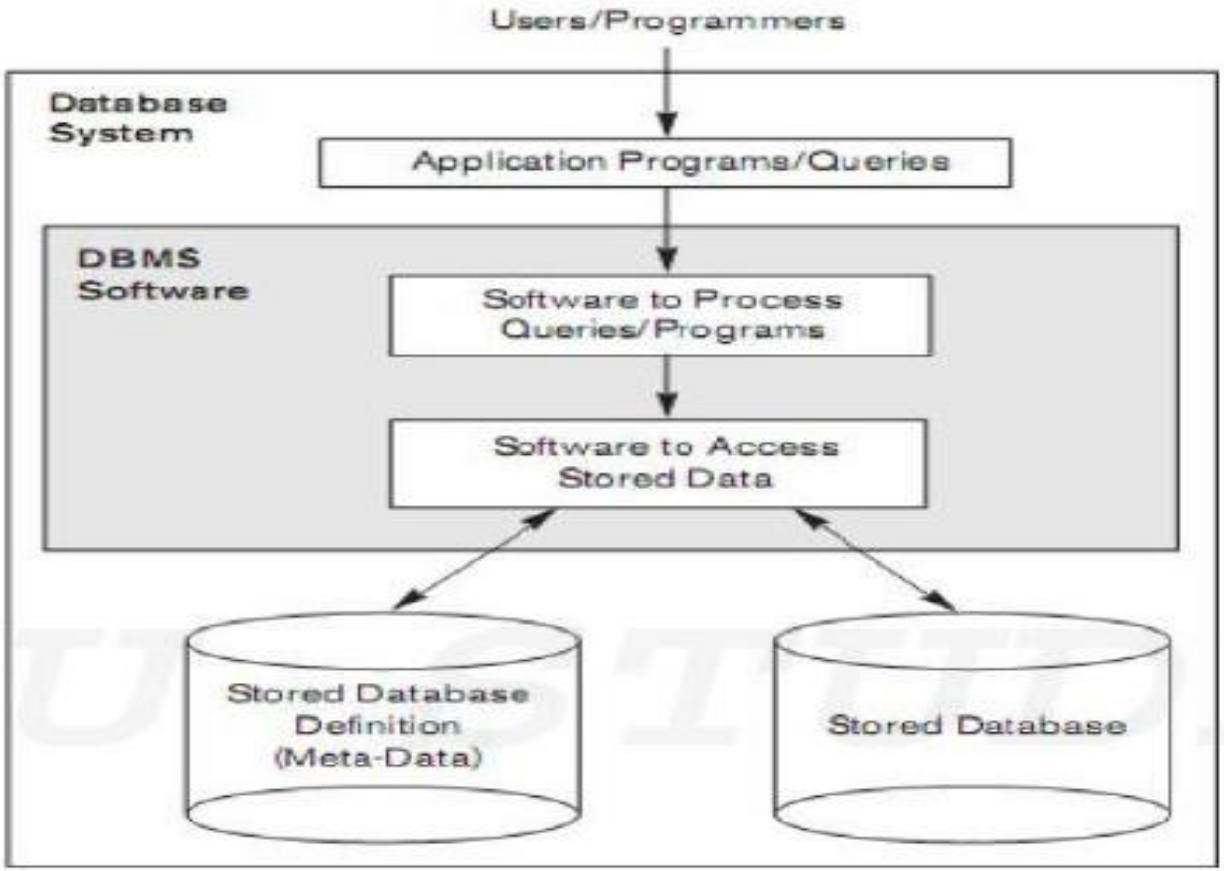
- **Defining** a database involves specifying the data types, structures, and constraints of the data to be stored in the database.The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called **meta-data.**
- **Constructing** the database is the process of storing the data on some storage medium that is controlled by the DBMS.
- **Manipulating** a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
- **Sharing** a database allows multiple users and programs to access the database simultaneously.

Two more additional functions provided by the DBMS include ***protecting*** *the database* and ***maintaining*** *it over a long period of time*.

- ***Protection :*** includes system protection against hardware or software malfunction (or crashes) and *security protection* against unauthorized or malicious access.

- ***Maintenance:*** include updation to improve performance or to meet user needs.

Database System Environment



Characteristics of the Database Approach

1. Self describing nature of a database system
2. Insulation between programs and data, and data abstraction
3. Support of multiple views of the data
4. Sharing of data and multiuser transaction processing

Self-Describing Nature of a Database System

- Database system contains not only the database itself but also a complete definition or description of the database structure and constraints.
- This definition is stored in the DBMS catalog
- Information stored in the catalog is called metadata and it describes the structure of the primary database.

Student Database

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

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	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

Metadata

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
....	....	.....
....	....	.....

Insulation between Programs and Data, and Data Abstraction

- The structure of data files is stored in the DBMS catalog separately from the access programs.This property is called **program-data independence**
- An operation (also called a function or method) is specified in two parts.
- Interface
  - The interface (or signature) of an operation includes the operation name and the data types of its arguments (or parameters).
- Implementation
  - The implementation (or method) of the operation is specified separately and can be changed without affecting the interface.

- User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented. This may be termed **program-operation independence.**

- The characteristic that allows **program-data independence** and **program operation independence** is called **data abstraction.**

Support of Multiple Views of the Data

- A database has many users, each user may require a different perspective or view of the database.
- A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.

Sharing of Data and Multiuser Transaction Processing

- DBMS must include concurrency control software
  - to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct
- DBMS must enforce several transaction properties
  - Isolation property
  - Atomicity property

Isolation Property

- ensures that each transaction appears to execute in isolation from other transactions
- even though hundreds of transactions may be executing concurrently.

Atomicity Property

- ensures that either all the database operations in a transaction are executed or none are.
- Any mechanical or electrical device is subject to failure, and so is the computer system. In this case we have to ensure that data should be restored to a consistent state.
- For example an amount of Rs 50 has to be transferred from Account A to Account B.
- Let the amount has been debited from account A but have not been credited to Account B and in the meantime, some failure occurred.
  - So, it will lead to an inconsistent state.
  - So, we have to adopt a mechanism which ensures that either full transaction should be executed or no transaction should be executed i.e. the fund transfer should be atomic.

Advantages of DBMS

- Controlling Redundancy
- Restricting Unauthorized Access
- Providing Persistent Storage for Program Objects (Eg: Object-oriented database systems are compatible with programming languages such as C++ and Java, and the DBMS software automatically performs any necessary conversions.)
- Providing Storage Structures for Efficient Query Processing
- Providing Backup and Recovery
- Providing Multiple User Interfaces
- Representing Complex Relationship among Data
- Enforcing Integrity Constraints
- Permitting Inferencing and Actions using Rules

Disadvantages of DBMS

- Cost of Hardware & Software
- Cost of Data Conversion
- Cost of Staff Training
- Appointing Technical Staff
- Database Damage

Database Users

- People who work with a database can be categorized as
  - Actors on the scene
  - Workers behind the scene

ACTORS ON THE SCENE

The people whose jobs involve the day-to-day use of a large database are called as the actors on the scene.

1. Database Administrators
2. Database Designers
3. End Users
4. System Analyst and Application Programmers(Software engineers)

Database Administrator (DBA)

A person who has central control over the system is called a database administrator (DBA).

- In a database environment, the primary resource is the database itself, and the secondary resource is the DBMS and related software. Administering these resources is the responsibility of the database administrator (DBA).
- The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed.
- The DBA is accountable for problems such as security breaches and poor system response time.

Database Designers

- Database designers are responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data.
- It is the responsibility of database designers to communicate with all prospective database users in order to understand their requirements and to create a design that meets these requirements.
- Database designers typically interact with each potential group of users and develop views of the database that meet the data and processing requirements of these groups.

End Users

- Casual users
- Naive or parametric users
- Sophisticated users
- Standalone users

- **Casual end users :** occasionally access the database, but they may need different information each time.

- **Naive or parametric end users :**Their main job function revolves around constantly querying and updating the database, using standard types of queries and updates called **canned transactions** that have been carefully programmed and tested. The tasks that such users perform are varied:
  - Bank tellers check account balances and post withdrawals and deposits.
  - Reservation agents for airlines, hotels, and car rental companies check availability for a given request and make reservations.

- **Sophisticated end users :** include engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirements.

- **Standalone users :** maintain personal databases by using ready-made program packages that provide easy-to-use menu-based or graphics-based interfaces.

System Analysts and Application Programmers (Software Engineers)

- **System analysts** determine the requirements of end users, especially naive and parametric end users, and develop specifications for standard canned transactions that meet these requirements.
- **Application programmers** implement these specifications as programs; then they test, debug, document, and maintain these canned transactions. Such analysts and programmers—commonly referred to as **software developers or software engineers**

WORKERS BEHIND THE SCENE

- The people who work to maintain the database system environment but who are not actively interested in the database contents as part of their daily job are called as the workers behind the scene

1. DBMS system designers and implementers
2. Tool developers
3. Operators and maintenance personnel (system administration personnel)

- **DBMS system designers and implementers :** design and implement the DBMS modules and interfaces as a software package.
- **Tool developers :** design and implement tools—the software packages that facilitate database modeling and design, database system design, and improved performance.
- **Operators and maintenance personnel (system administration personnel) :** are responsible for the actual running and maintenance of the hardware and software environment for the database system.

Structured, Semi-structured and Unstructured data

Structured data

- Represented in a strict format
- It has been organized into a formatted repository that is typically a database.
- It concerns all data which can be stored in database SQL in a table with rows and columns
- *Example: Relational data*

Semi-Structured data

- Information that does not reside in a relational database but that have some organizational properties that make it easier to analyze
- With some process, you can store them in the relation database
- •*Example: XML data*

Unstructured data

- Data which is not organized in a predefined manner or does not have a predefined data model.
- It is not a good fit for a mainstream relational database.
- There are alternative platforms for storing and managing. It is increasingly prevalent in IT systems and is used by organizations in a variety of business intelligence and analytics applications.
- *Example:Word, PDF, Text, Media logs.*

FEATURES	STRUCTURED	SEMI STRUCTURED	UNSTRUCTURED
Format Type	Relational Database	HTML, XML, JSON	Binary, Character
Version Management	Rows, columns, tuples	Not as common – graph is possible	Whole data
Implementation	SQL	Anonymous nodes	-
Robustness	Robust	Limited robustness	-
Storage Requirement	Less	Significant	Large
Applications	DBMS, RDF, ERP system, Data Warehouse, Apache Parquet, Financial Data, Relational Table	Server Logs, Sensor Output	No SQL, Video, Audio, Social Media, Online Forums, MRI, Ultrasound

Data Models

- A collection of concepts that can be used to describe the structure of a database
- Structure of a database means the data types, relationships, and constraints that should hold on the data.
- Most data models also include a set of basic operations for specifying retrievals and updates on the database.

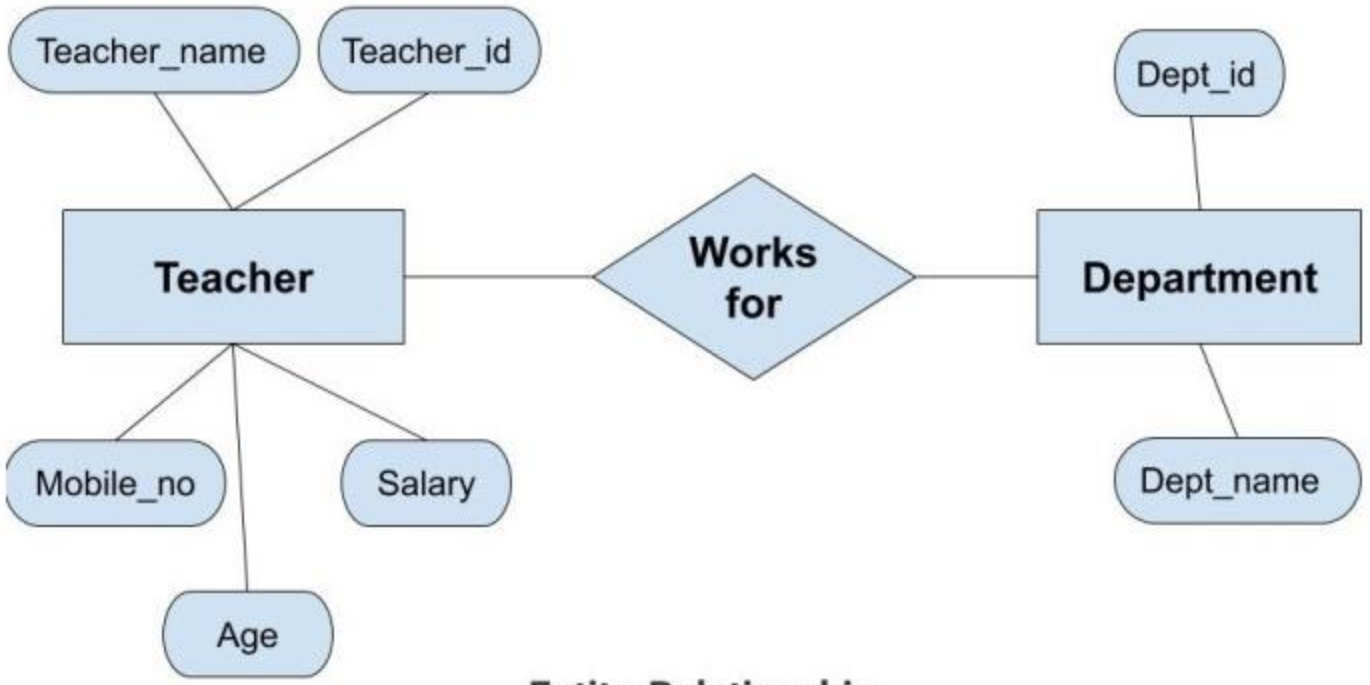
Categories of Data Models

- High-level or conceptual data models
- Low-level or physical data models
- Representational (or implementation) data models

High-level or conceptual data models

- Provide concepts that are close to the way many users perceive data .
- Use concepts such as entities, attributes, and relationships .
- An entity represents a real-world object or concept.(Eg: Teacher, Department in the figure)
- An attribute represents some property of interest that further describes an entity.(Eg: Teacher_name, Dept_id etc in the figure)
- A relationship among two or more entities represents an interaction among the entities (Eg: Works for in the figure)

Example:



Entity-Relationship Model

Low-level or physical data models

- Provide concepts that describe the details of how data is stored in the computer.
- Concepts provided by low-level data models are generally meant for computer specialists, not for typical end users.
- Describe how data is stored in the computer by representing information such as table formats, record formats, record orderings, and access paths.
- An access path is a structure that makes the search for particular database records efficient.

Representational (or implementation) data models

- It provide concepts that may be understood by end users.
- It hides some details of data storage but can be implemented on a computer system in a direct way.
- Used most frequently in traditional commercial DBMSs, and they include the widely-used **relational data model** the **network model, object oriented model** and **hierarchical model.**
- It is sometimes called **record-based data models**

Relational data model

- It is the most widely used model. In this model, the data is maintained in the form of a two-dimensional table. All the information is stored in the form of row and columns. The basic structure of a relational model is tables. So, the tables are also called *relations* in the relational model.

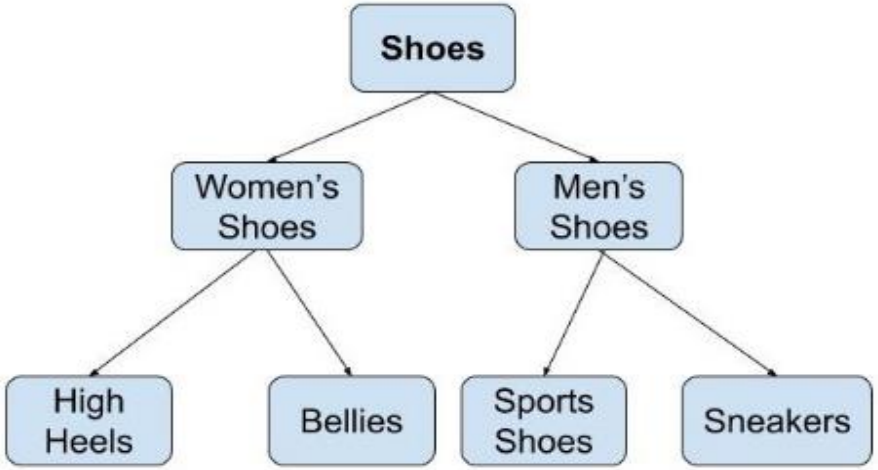
Example: In this example, we have an Employee table.

Emp_id	Emp_name	Job_name	Salary	Mobile_no	Dep_id	Project_id
AfterA001	John	Engineer	100000	9111037890	2	99
AfterA002	Adam	Analyst	50000	9587569214	3	100
AfterA003	Kande	Manager	890000	7895212355	2	65

EMPLOYEE TABLE

Hierarchical Model

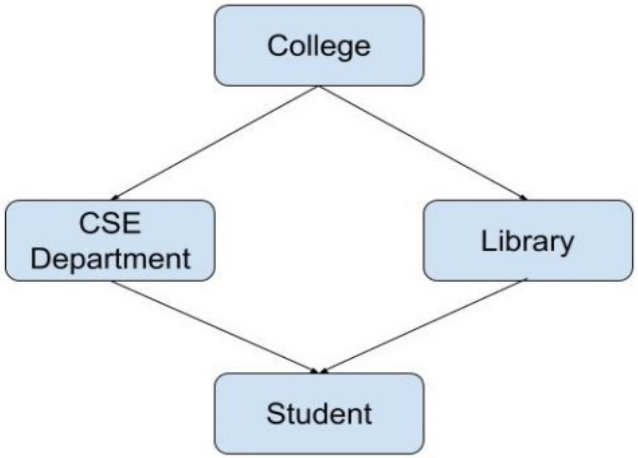
- Hierarchical Model was the first DBMS model. This model organises the data in the hierarchical tree structure. The hierarchy starts from the root which has root data and then it expands in the form of a tree adding child node to the parent node.



Hierarchical Model

Network Model

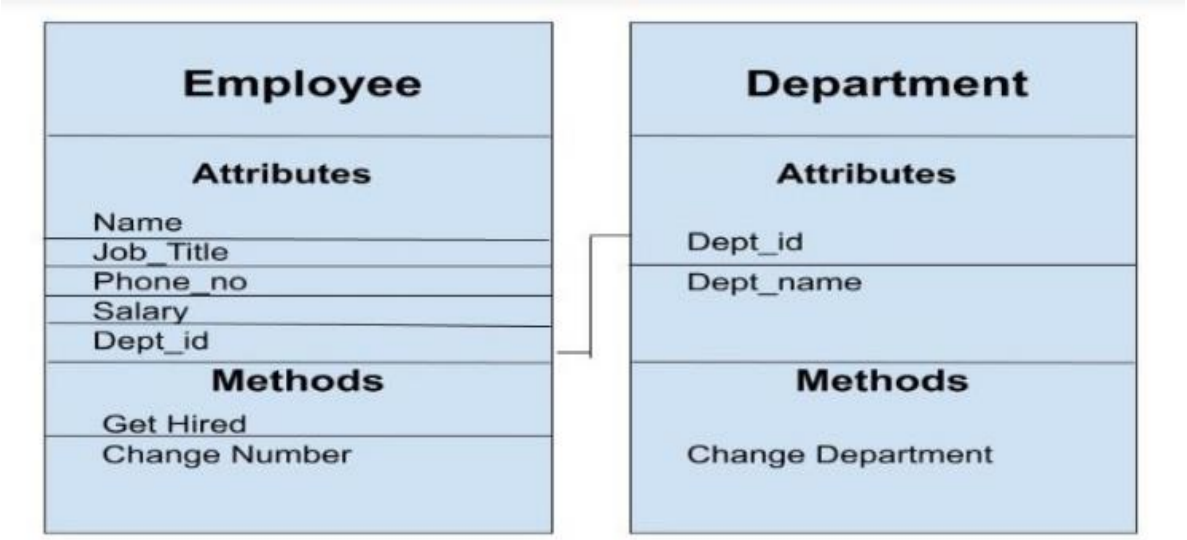
- This model is an extension of the hierarchical model. It was the most popular model before the relational model. This model is the same as the hierarchical model, the only difference is that a record can have more than one parent. It replaces the hierarchical tree with a graph.



Network Model

Object-Oriented Data Model

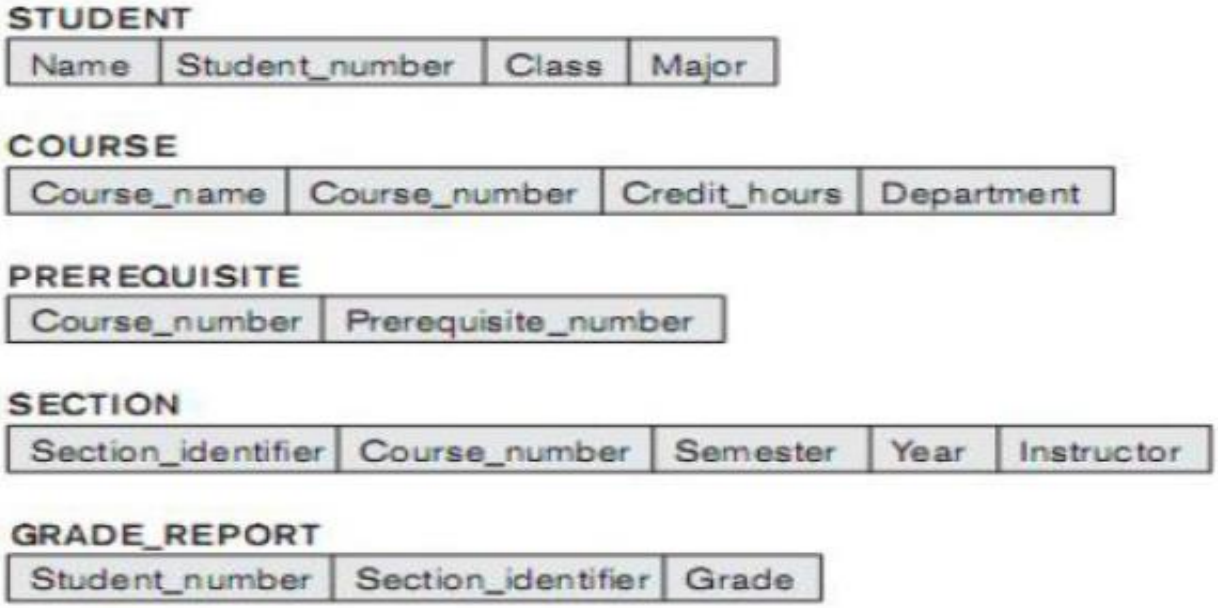
- The real-world problems are more closely represented through the object-oriented data model. In this model, both the data and relationship are present in a single structure known as an object.
- In the below example, we have two objects Employee and Department. All the data and relationships of each object are contained as a single unit.



Object_Oriented_Model

Schemas, Instances, and Database State

- The description of a database is called the **database schema,** which is specified during database design and is not expected to change frequently
- A displayed schema is called a **schema diagram.** We call each object in the schema a schema construct.
- In a given database state, each schema construct has its own current set of **instances**; for example, the STUDENT construct will contain the set of individual student entities (records) as its instances.
- The data in database at particular instant or moment of time is called **database state** or **snapshot**



Database schema

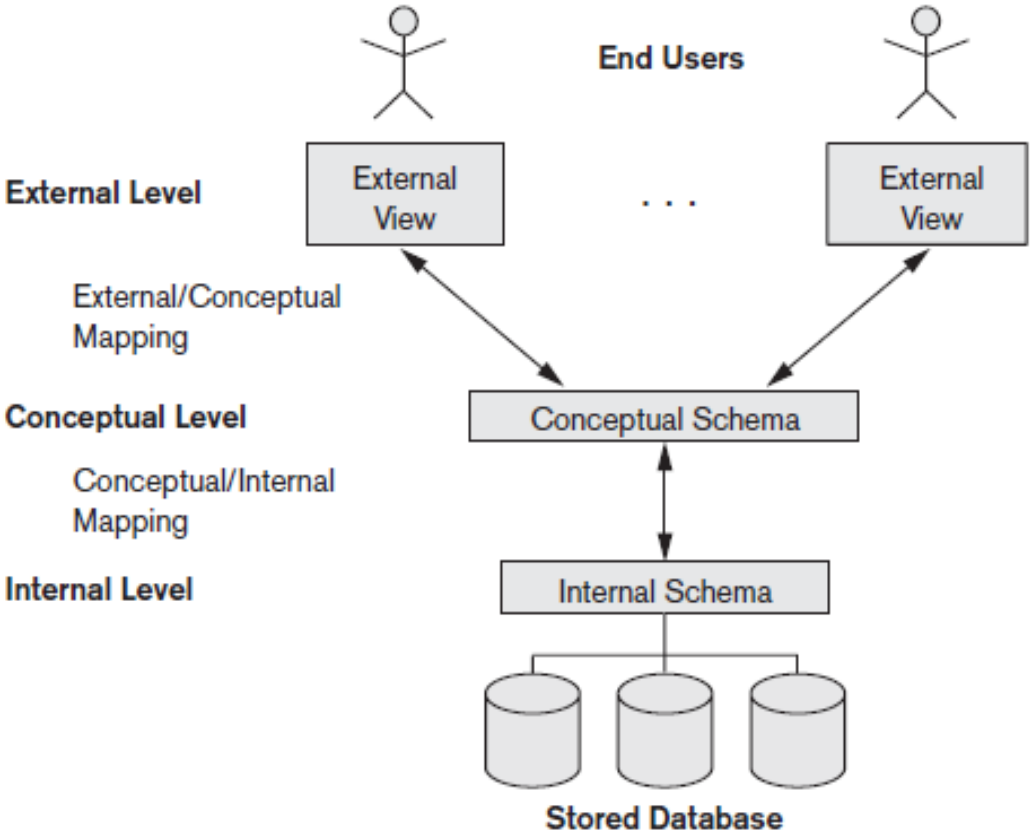
Name	Student_number	Class	Major
Ram	CS001	R4	CSE
Shyam	CS002	R4	CSE

Database state

- The schema is not supposed to change frequently, but it is not uncommon that changes occasionally need to be applied to the schema as the application requirements change. It is called **schema evolution.**

The Three Scheme Architecture

The three-schema architecture.



Internal level

- The **internal level** has an internal schema, which describes the physical storage structure of the database.

- The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

Conceptual level

- Describes the structure of the whole database for a community of users.
- The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints.
- The implementation conceptual schema is often based on a conceptual schema design in a high-level data model.

External or view level

- The **external** or **view level** includes a number of external schemas or user views.
- Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.
- Each external schema is typically implemented using a **representational data model**, possibly based on an external schema design in a high-level data model.

- In a DBMS based on the three-schema architecture, each user group refers only to its own external schema.
- Hence, the DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database.
- If the request is a database retrieval, the data extracted from the stored database must be reformatted to match the user's external view.

Mappings

- The processes of transforming requests and results between levels are called **mappings.**

- These mappings may be time-consuming, so some DBMSs—especially those that are meant to support small databases—do not support external views.

- A certain amount of mapping is necessary to transform requests between the conceptual and internal levels.

Data Independence

- The capacity to change the schema at one level of a database system without having to change the schema at the next higher level.

Two types of data independence:

- 1.Logical data independence
- 2.Physical data independence

Logical data independence

- Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. (Eg1: Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application program.)

Physical data independence

- Physical data independence is the capacity to change the internal schema without having to change the conceptual schema.
- Data independence occurs because when the schema is changed at some level, the schema at the next higher level remains unchanged; only the mapping between the two levels is changed. (Eg:Switching to different data structures or Change the location of database from say C Drive to D Drive).

Database Languages and Interfaces DBMS Languages

- **Data definition language (DDL)**, is used by the DBA and by database designers to define conceptual and internal schemas schemas.
- **Storage definition language (SDL)**, is used to specify the internal schema.
- **View definition language (VDL)**, to specify user views and their mappings to conceptual schema
- **Data Manipulation Language(DML)** is used for retrieval, insertion, deletion, and modification of the data

There are two main types of DMLs.

- **High-level** or **nonprocedural DML** can be used on its own to specify complex database operations concisely.
- **Low-level** or **procedural DML** must be embedded in a general-purpose programming language. This type of DML typically retrieves individual records or objects from the database and processes each separately. Therefore, it needs to use programming language constructs, such as looping, to retrieve and process each record from a set of records. Low-level DMLs are also called record-at-a-time DMLs

DBMS Structure

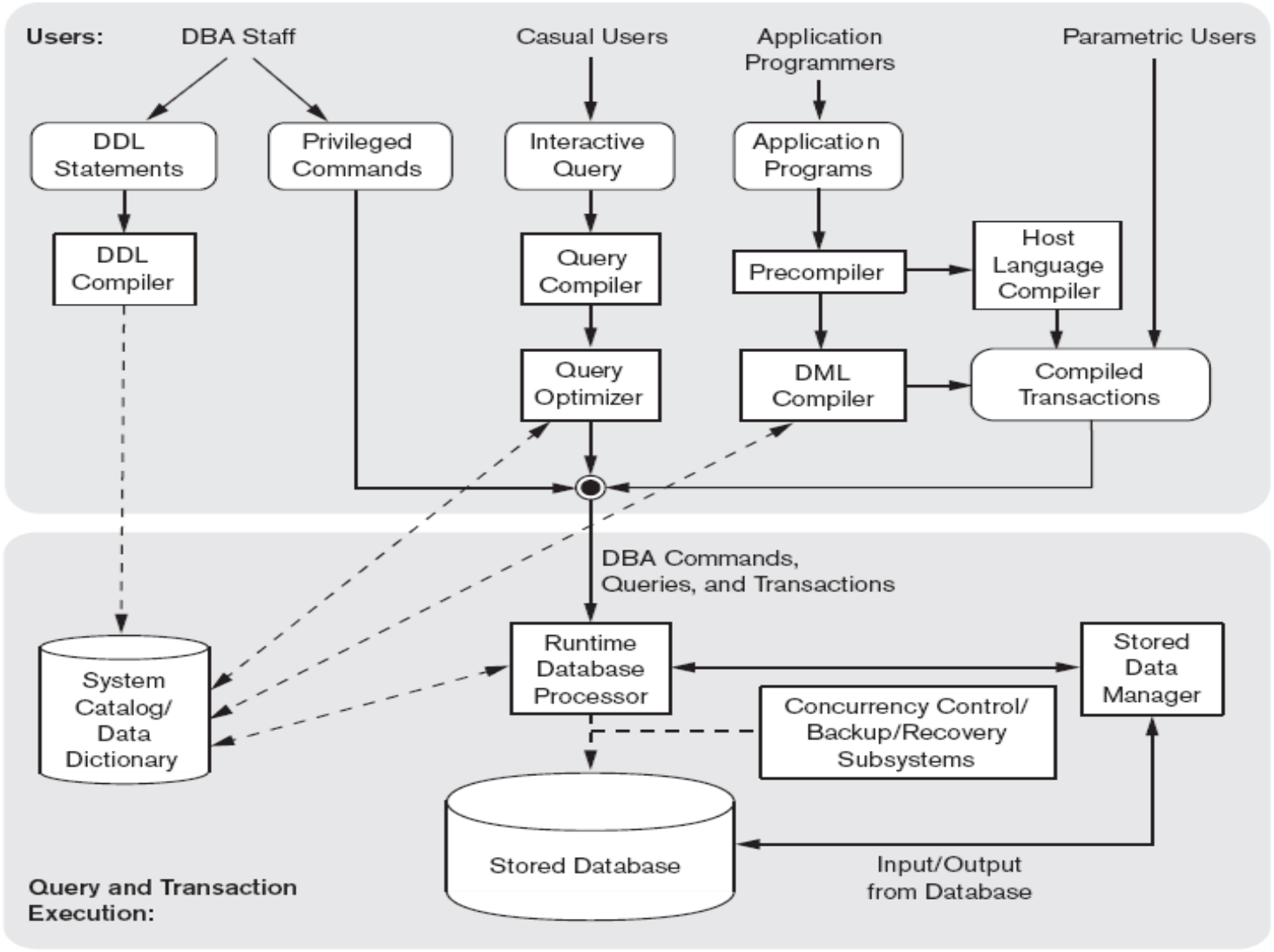


Figure 2.3
Component modules of a DBMS and their interactions.

Database Architecture

- The figure is divided into two parts.
  - The top part of the figure refers to the various users of the database environment and their interfaces.
  - The lower part shows the internals of the DBMS responsible for storage of data and processing of transactions.

1. Users

The top part shows interfaces for the DBA staff, casual users who work with interactive interfaces to formulate queries, application programmers who create programs using some host programming languages, and parametric users who do data entry work by supplying parameters to predefined transactions. The DBA staff works on defining the database and tuning it by making changes to its definition using the DDL and other privileged commands.

2. **DDL compiler :** The DDL compiler processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog.
3. **Query compiler :** The interactive queries are parsed and validated for correctness of the query syntax, the names of files and data elements, and so on by a query compiler that compiles them into an internal form
4. **Query optimizer :** After query compilation, the query optimizer is concerned with the rearrangement and possible reordering of operations, elimination of redundancies, and use of correct algorithms and indexes during execution.
5. **Precompiler :** Application programmers write programs in host languages such as Java, C, or C++ that are submitted to a precompiler. The **precompiler extracts DML commands** from an application program written in a host programming language.

6. **DML compiler :** It collects the DML commands from precompiler and compiles them to produce object code for database and the rest of the program is sent to the **host language compiler**.
7. **Compiled/Canned transactions:** The object codes for the DML commands and the rest of the program are linked, forming a canned transaction whose executable code includes calls to the runtime database processor. Canned transactions are executed repeatedly by parametric users, who simply supply the parameters to the transactions. Each execution is considered to be a separate transaction. An example is a bank withdrawal transaction where the account number and the amount may be supplied as parameters.

- In the lower part of Figure , the **runtime database processor** executes (1) the privileged commands, (2) the executable query plans, and (3) the canned transactions with runtime parameters.
- It works with the **system catalog** and may update it with statistics.
- It also works with the **stored data manager,** which in turn uses basic operating system services for carrying out low-level input/output (read/write) operations between the disk and main memory.
- **The runtime database processor** handles other aspects of data transfer, such as management of buffers in the main memory.
- The **concurrency control and backup and recovery systems** are integrated into the working of the runtime database processor for purposes of transaction management.

DBMS Architectures and Classification

DBMS architectures are of different types

- **Centralized DBMSs Architecture**
- **Basic Client/Server Architectures**
- **Two-Tier Client/Server Architectures for DBMS**
- **Three-Tier and n-Tier Architectures for Web Applications**

1. Centralized DBMSs Architecture

- Earlier architectures used mainframe computers.
- Accessed via computer terminals that doesn't have processing power and only provided display capabilities.
- Processing performed remotely on the computer system.
- Only the display information and controls are sent to the display terminals.
- Terminals were replaced with PCs and Workstations.
- All DBMS functionality, application program execution, and user interface processing carried out on one machine.

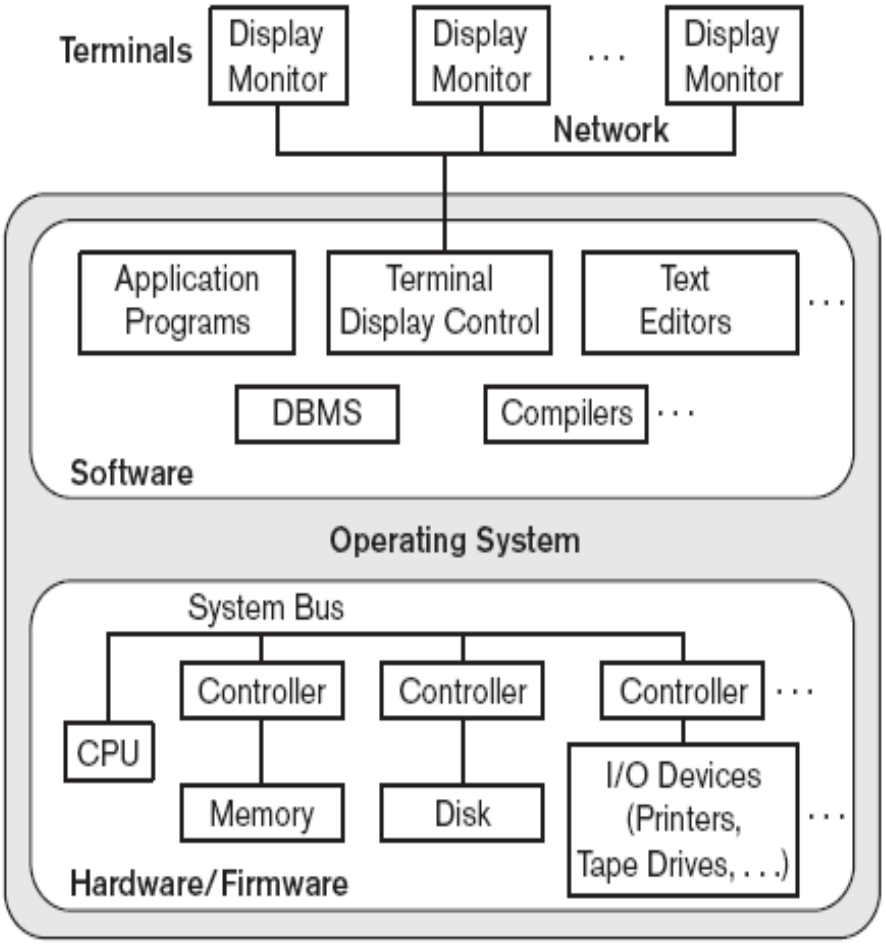


Figure 2.4
A physical centralized architecture.

2. Basic Client/Server Architectures

- Deal with computing environments in which a large number of PC's, workstations, servers and other software and equipments are connected via a network.
- Define **specialized servers** with specific functionalities
- Ex: File server- maintains the files of the client machines connected.
- Printer server- all print requests by the clients are forwarded to this machine.

Contd..

- A **client** in this framework is typically a user machine that provides user interface capabilities and local processing. When a client requires access to additional functionality such as database access, that does not exist at that machine, it connects to a server that provides the needed functionality.
- A **server** is a system containing both hardware and software that can provide services to the client machines, such as file access, printing, archiving, or database access.

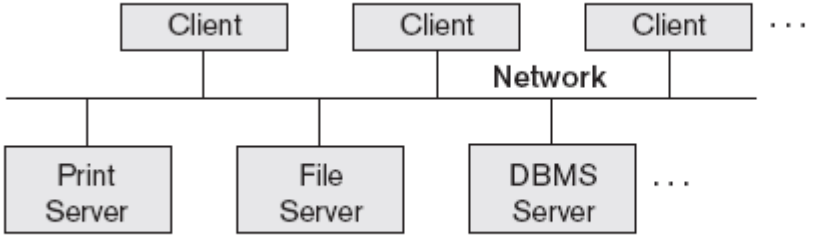


Figure 2.5
Logical two-tier client/server architecture.

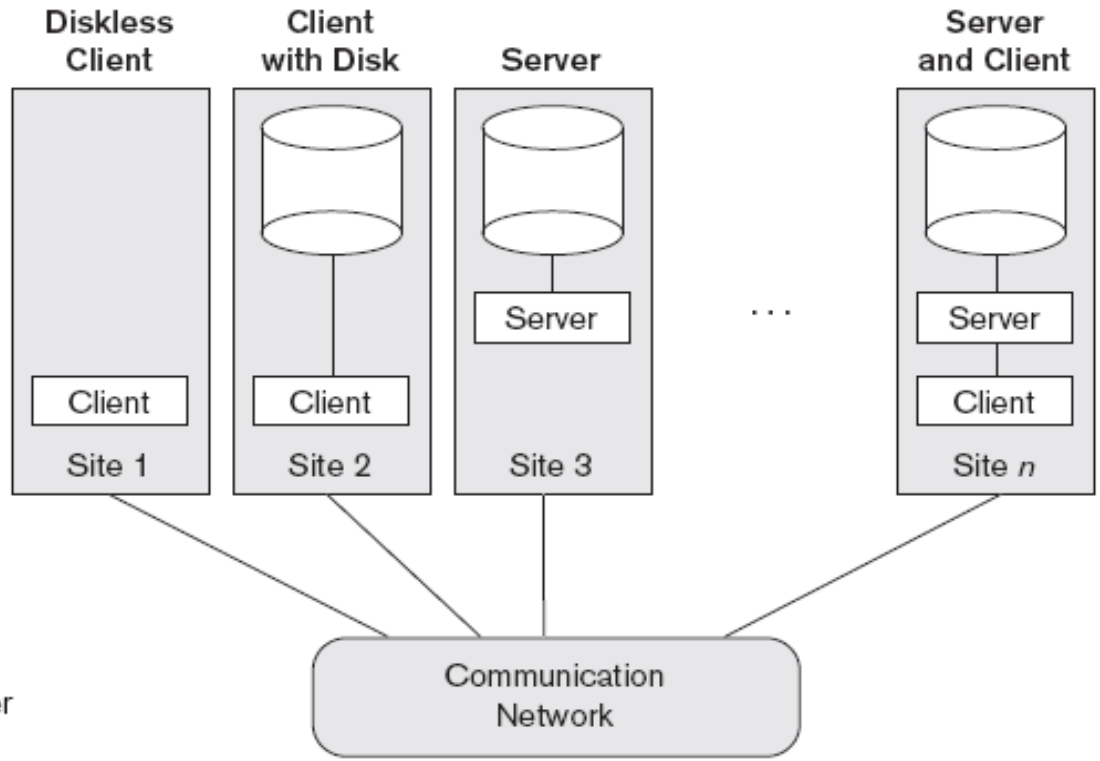


Figure 2.6
Physical two-tier client/server architecture.

3. Two-Tier Client/Server Architectures for DBMS

- Server handles
  - Query and transaction functionality related to SQL processing.
  - Server is often called query server or transaction server because it provides these two functionalities.
- Client handles
  - User interface programs and application programs
  - For DBMS access, program establishes a connection to the server
- 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
  - Allows Java client programs to access one or more DBMSs through a standard interface

4. Three tier architecture

- Adds intermediate layer between client and the database server called Application server or Web server
- Improves database security by checking client's credentials before forwarding a request to the database server.
- User interface, application rules, and data access act as the three tiers.

➢ **Server**

- Runs application programs and stores business rules

➢ **Clients** contain GUIs and application-specific business rules.

➢ **Intermediate server**

- Accepts request from the client and process it.
- Sends database queries and commands to database server
- Acts as a channel for passing partially processed data from the database server to clients
- Further processed and filtered for GUI

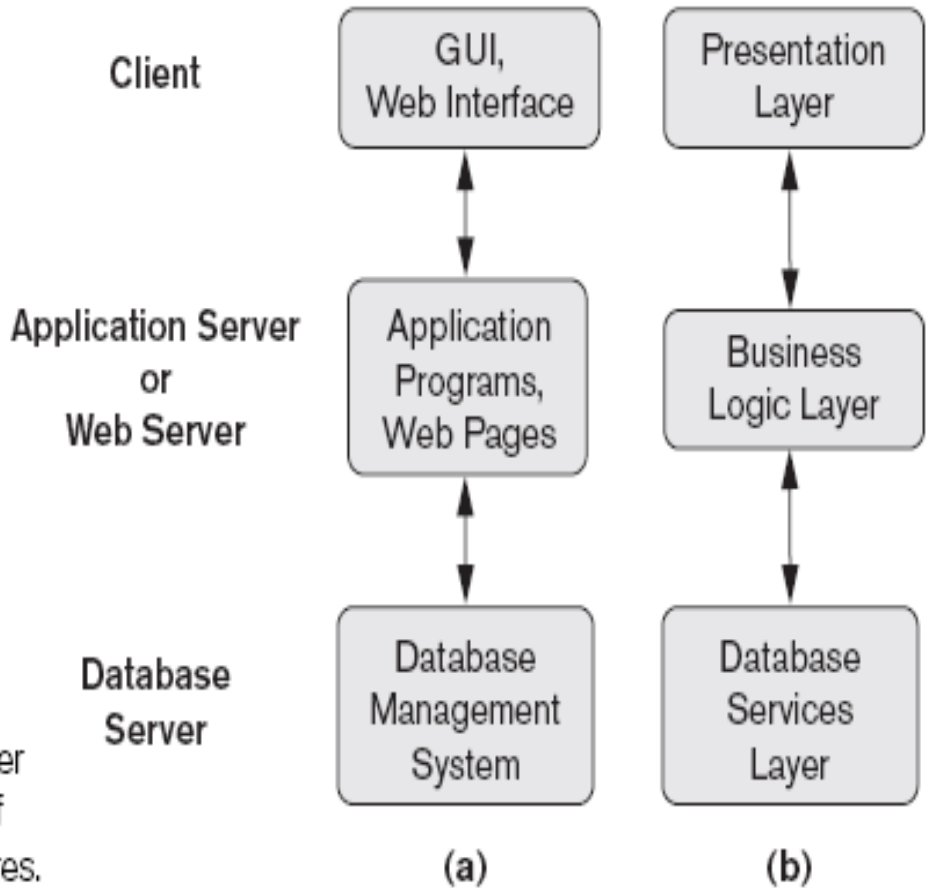


Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

- Presentation layer
  - Displays information to user and allows data entry
- Business logic layer
  - Handles intermediate rules and constraints before data is passed to user or DBMS.
  - Also acts as a web server that retrieves query results from the database server
  - Then formats it into dynamic web pages viewed by a web browser.
- Bottom layer
  - Includes data management services

N-tier

- Divide the layers between the user and the stored data further into finer components.
- Business logic layer is usually divided into multiple layers.

CLASSIFICATION OF DATABASE MANAGEMENT SYSTEMS

- DBMS is classified according to several criteria: data model, number of users, number of sites,

1. **Data model :** We can hence categorize DBMSs based on the data model: relational, object, object-relational, hierarchical, network etc.
2. **Number of users :** Single-user systems support only one user at a time and are mostly used with personal computers. Multiuser systems, which include the majority of DBMSs, support multiple users concurrently.
3. **Number of sites :** DBMS is centralized if the data is stored at a single computer site. A centralized DBMS can support multiple users, but the DBMS and the database themselves reside totally at a single computer site. A distributed DBMS (DDBMS) can have the actual database and DBMS software distributed over many sites, connected by a computer network.

Contd..

- **Homogeneous DBMS** use the same DBMS software at multiple sites. A recent trend is to develop software to access several autonomous preexisting databases stored under **Heterogeneous DBMS.**

4. **Types of access path :** DBMS can be general purpose or special purpose. When performance is a primary consideration, a special-purpose DBMS can be designed and built for a specific application; such a system cannot be used for other applications without major changes. Many airline reservations and telephone directory systems developed in the past are special purpose DBMSs. These fall into the category of online transaction processing (OLTP) systems, which must support a large number of concurrent transactions without imposing excessive delays.
5. Cost : Based on cost there are free DBMS softwares(MySQL) and paid DBMS softwares(Informix).

DBMS Interfaces

User-friendly interfaces provided by a DBMS may include the following:

1. Menu-Based Interfaces for Web Clients or Browsing.
2. Forms-Based Interfaces.
3. Graphical User Interfaces (GUI)
4. Natural Language Interfaces
5. Speech Input and Output
6. Interfaces for Parametric Users.