Unit I

Introduction: Database and Database Users, Characteristics of the Database Approach, Actors on the scene, Workers behind the Scene, Advantages of using DBMS, Brief History. Database System Concepts and Architecture: Data Models, Schemas, and Instances, Three Schema Architecture and Data Independence, Database language and interfaces, the database system Environment, Centralized and Client/Server Architectures for DBMS, Classification of Database Management Systems.

What is Data

- **Data** can be defined as a representation of facts, concepts, or instructions in a formalized manner, which should be suitable for communication, interpretation, or processing by human or electronic machine.

- Data is represented with the help of characters such as alphabets (A-Z, a-z), digits (0-9) or special characters (+,-,/,*,<,>,= etc.) or images, videos.

What is Information

Information is organized or classified data, which has some meaningful values for the receiver. Information is the processed data on which decisions and actions are based.

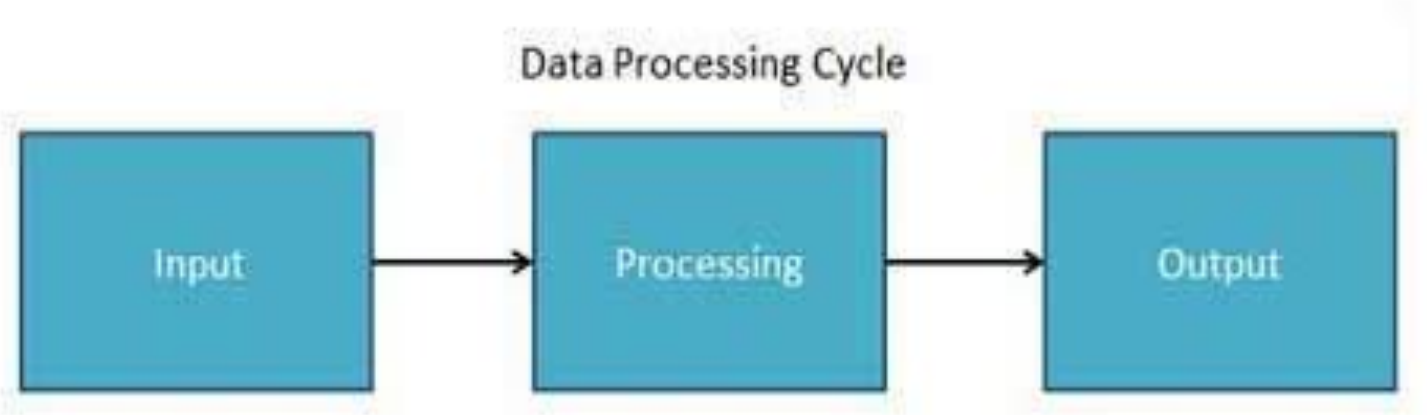
For the decision to be meaningful, the processed data must qualify for the following characteristics −

- **Timely** − Information should be available when required.
- **Accuracy** − Information should be accurate.
- **Completeness** − Information should be complete

Data Processing Cycle

- Data processing is the re-structuring or re-ordering of data by people or machine to increase their usefulness and add values for a particular purpose.

- Data processing consists of the following basic steps - input, processing, and output. These three steps constitute the data processing cycle.



Data Processing Cycle

Input − In this step, the input data is prepared in some convenient form for processing. The form will depend on the processing machine. For example, when electronic computers are used, the input data can be recorded on any one of the several types of input medium, such as magnetic disks, tapes, and so on.

Processing − In this step, the input data is changed to produce data in a more useful form. For example, pay-checks can be calculated from the time cards, or a summary of sales for the month can be calculated from the sales orders.

Output − At this stage, the result of the proceeding processing step is collected. The particular form of the output data depends on the use of the data. For example, output data may be pay-checks for employees.

Difference between Data and Information

S.NO	DATA	INFORMATION
1	Data are the variables which helps to develop ideas/conclusions.	Informations are the meaningful of data.
2	Data are text and numerical values.	Information is refined form of actual data.
3	Data doesn't rely on Information.	While Information relies on Data.
4	Bits and Bytes are the measuring unit of data.	Information is measured in meaningful units like time, quantity, etc.
5	Data can be easily structured as the following: 1.Tabular data 2.Graph 3.Data tree	Information can also be structured as the following: 1.Language 2.Ideas 3.Thoughts
6	Data does not have any specific purpose	Information carries a meaning that has been assigned by interpreting data.
7	It is low-level knowledge.	It is the second level of knowledge.
8	Data does not directly helps in decision making.	Information directly helps in decision making.

What is Database

- The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently.

- It is also used to organize the data in the form of a table, schema, views, and reports, etc.

- **For example:** The college Database organizes the data about the admin, staff, students and faculty etc.

- Using the database, you can easily retrieve, insert, and delete the information.

Database Management System

- Database management system is a software which is used to manage the database. For example: MySQL, Oracle, etc are a very popular commercial database which is used in different applications.

- DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.

- It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

Characteristics / Why to Learn DBMS?

- **Real-world entity** − A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses the behavior and attributes too. For example, a school database may use students as an entity and their age as an attribute.

- **Relation-based tables** − DBMS allows entities and relations among them to form tables. A user can understand the architecture of a database just by looking at the table names.

- **Isolation of data and application** − A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.

Characteristics of DBMS

- **Less redundancy** − DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.

- **Consistency** − Consistency is a state where every relation in a database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state. A DBMS can provide greater consistency as compared to earlier forms of data storing applications like file-processing systems.

- **Query Language** − DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.

Advantages of DBMS

- **Controls database redundancy:** It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
- **Data sharing:** In DBMS, the authorized users of an organization can share the data among multiple users.
- **Easily Maintenance:** It can be easily maintainable due to the centralized nature of the database system.
- **Reduce time:** It reduces development time and maintenance need.
- **Backup:** It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
- **multiple user interface:** It provides different types of user interfaces like graphical user interfaces, application program interfaces

Disadvantages of DBMS

- **Cost of Hardware and Software:** It requires a high speed of data processor and large memory size to run DBMS software.

- **Size:** It occupies a large space of disks and large memory to run them efficiently.

- **Complexity:** Database system creates additional complexity and requirements.

- **Higher impact of failure:** Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

Simple example of a student database

RollNo	Name	Course	City
1	alok pandey	mca	varanasi
2	anil kumar	bca	delhi
3	amar singh	mtech	allahabad
4	barat sharma	mba	delhi
5	babita mishra	bba	kanpur
6	pramod sharma	bsc	jhansi
7	pranav singh	bsc	patna
10	shruti gupta	ba	delhi
11	pawan pandey	btech	noida
13	satish kumar	mtech	mumbai

Different types of Database Users / Actors on the Scene

1. **Database Administrator (DBA) :**

- Database Administrator (DBA) is a person/team who defines the schema and also controls the 3 levels of database.

- The DBA will then create a new account id and password for the user if he/she need to access the data base.

- DBA is also responsible for providing security to the data base and he allows only the authorized users to access/modify the data base.DBA also monitors the recovery and back up and provide technical support.

- The DBA has a DBA account in the DBMS which called a system or superuser account.

Different types of Database Users / Actors on the Scene

2. Naive / Parametric End Users :

- Parametric End Users are the unsophisticated who don't have any DBMS knowledge but they frequently use the data base applications in their daily life to get the desired results.
- For examples, Railway's ticket booking users are naive users. Clerks in any bank is a naive user because they don't have any DBMS knowledge but they still use the database and perform their given task.

3. System Analyst :

- System Analyst is a user who analyzes the requirements of parametric end users.
- They check whether all the requirements of end users are satisfied.

Different types of Database Users / Actors on the Scene

4. Sophisticated Users :

- Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database.

- They can develop their own data base applications according to their requirement. They don’t write the program code but they interact the data base by writing SQL queries directly through the query processor.

5. Data Base Designers :

- Data Base Designers are the users who design the structure of data base which includes tables, indexes, views, constraints, triggers, stored procedures.

- He/she controls what data must be stored and how the data items to be related.

Different types of Database Users / Actors on the Scene

6. Application Program :

• Application Program are the back end programmers who writes the code for the application programs.

• They are the computer professionals. These programs could be written in Programming languages such as Visual Basic, Developer, C, FORTRAN, COBOL etc.

7. Casual Users / Temporary Users :

Casual Users are the users who occasionally use/access the data base but each time when they access the data base they require the new information, for example, Middle or higher level manager.

Different types of Database Users / Actors on the Scene

6. Application Program :

• Application Program are the back end programmers who writes the code for the application programs.

• They are the computer professionals. These programs could be written in Programming languages such as Visual Basic, Developer, C, FORTRAN, COBOL etc.

7. Casual Users / Temporary Users :

Casual Users are the users who occasionally use/access the data base but each time when they access the data base they require the new information, for example, Middle or higher level manager.

Workers behind the scene

- **DBMS designers and implementers:** Design and implement the DBMS software package itself.
- **Tool developers**: Design and implement tools that facilitate the use of the DBMS software. Tools include design tools, performance tools, special interfaces.
- **Operators and maintenance personnel**: Work on running and maintaining the hardware and software environment for the database system

History of DBMS

- 1960 - Charles Bachman designed first DBMS system
- 1970 - Codd introduced IBM'S Information Management System (IMS)
- 1976- Peter Chen coined and defined the Entity-relationship model also know as the ER model
- 1980 - Relational Model becomes a widely accepted database component
- 1985- Object-oriented DBMS develops.
- 1990s- Incorporation of object-orientation in relational DBMS.
- 1991- Microsoft ships MS access, a personal DBMS and that displaces all other personal DBMS products.
- 1995: First Internet database applications
- 1997: XML applied to database processing. Many vendors begin to integrate XML into DBMS products.

Difference between File System and DBMS

S.NO.	File System	DBMS
1.	File system is a software that manages and organizes the files in a storage medium within a computer.	DBMS is a software for managing the database.
2.	Redundant data can be present in a file system.	In DBMS there is no redundant data.
3.	It doesn't provide backup and recovery of data if it is lost.	It provides backup and recovery of data even if it is lost.
4.	There is no efficient query processing in file system.	Efficient query processing is there in DBMS.
5.	There is less data consistency in file system.	There is more data consistency because of the process of normalization.
6.	It is less complex as compared to DBMS.	It has more complexity in handling as compared to file system.
7.	File systems provide less security in comparison to DBMS.	DBMS has more security mechanisms as compared to file system.
8.	It is less expensive than DBMS.	It has a comparatively higher cost than a file system.

Single User and Multi User Database Systems

Single User

In these DBMS, at one time, only a single user can access the database. Hence, the user can use all the resources at all times.

All these systems are used for personal usage, such as personal computers experience. In this type of DBMS, both the physical and application layer can be used by the user.

Example – Personal Computers

Single User and Multi User Database Systems

Multi User

- These DBMSs supports two or more than two users accessing the database simultaneously.

- Multi-user systems contains all the mini-computers and mainframe computers.

- In mainframe computer, the database may exist on a single computer and in other computers, the database may be distributed in multiple computers.

- Multiple users can update data while working together simultaneously.
- **Example** –

Databases of Banks, insurance agencies, stock exchanges, supermarkets, etc.

Difference between Single User and Multi User Database Systems

Single User Database Systems	Multi User Database Systems
A DBMS is single-user if at most one user at a time can use the system.	A DBMS is multi-user if many/multi users can use the system and hence access the database concurrently.
Single-User DBMSs are mostly restricted to personal computer systems.	Most DBMSs are multi user, like databases of airline reservation systems, banking databases, etc.
Single user databases do not have multiprogramming thus, single CPU can only execute at most one process at a time.	Multiple users can access databases and use computer systems simultaneously because of the concept of Multiprogramming.
Example: Personal Computers.	Example: Databases of Banks, insurance agencies, stock exchanges, supermarkets, etc.

Data Model, Schema and Instance

- **Data modeling**

- **Data modeling (data modelling)** is the process of creating a data model for the data to be stored in a database.

- This data model is a conceptual representation of Data objects, the associations between different data objects, and the rules.

- Data modeling helps in the visual representation of data and enforces business rules, regulatory compliances, and government policies on the data.

- Data Models ensure consistency in naming conventions, default values, semantics, security while ensuring quality of the data.

Data Model

Data Model

- The **Data Model** is defined as an abstract model that organizes data description, data semantics, and consistency constraints of data.

- The data model emphasizes on what data is needed and how it should be organized instead of what operations will be performed on data.

- Data Model is like an architect's building plan, which helps to build conceptual models and set a relationship between data items.

Types of Data Model

1. Conceptual Data Model (High Level)

- A **Conceptual Data Model** is an organized view of database concepts and their relationships.

- The purpose of creating a conceptual data model is to establish entities, their attributes, and relationships.

- In this data modeling level, there is hardly any detail available on the actual database structure.

- Business stakeholders and data architects typically create a conceptual data model.

Types of Data Model

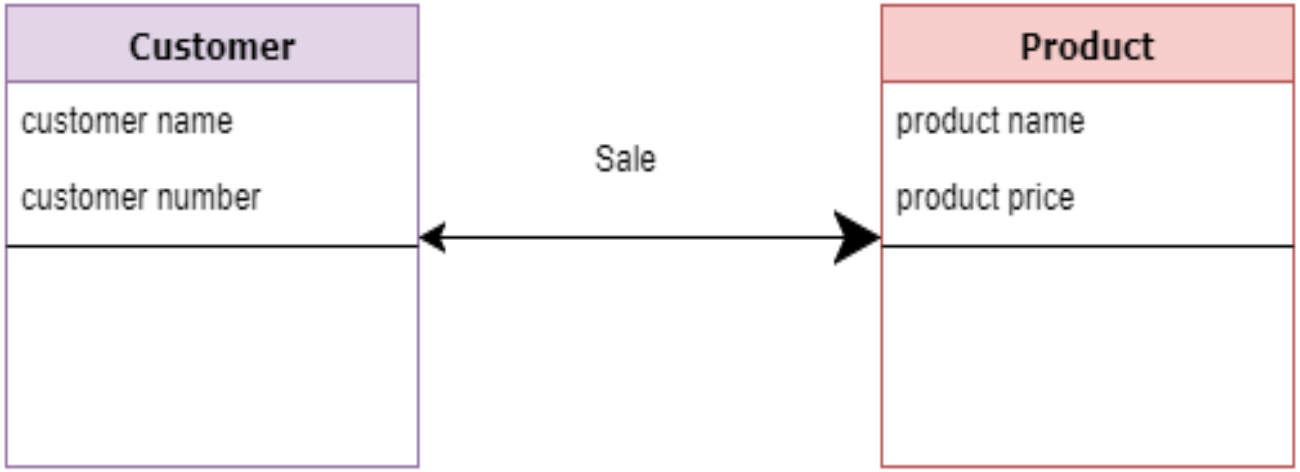
The 3 basic tenants of Conceptual Data Model are

- **Entity**: A real-world thing
- **Attribute**: Characteristics or properties of an entity
- **Relationship**: Dependency or association between two entities

Types of Data Model

Conceptual Data Model Example

- Customer and Product are two entities. Customer number and name are attributes of the Customer entity
- Product name and price are attributes of product entity
- Sale is the relationship between the customer and product



Conceptual Data Model

Types of Data Model

2. Logical Data Model

• The **Logical Data Model** is used to define the structure of data elements and to set relationships between them.

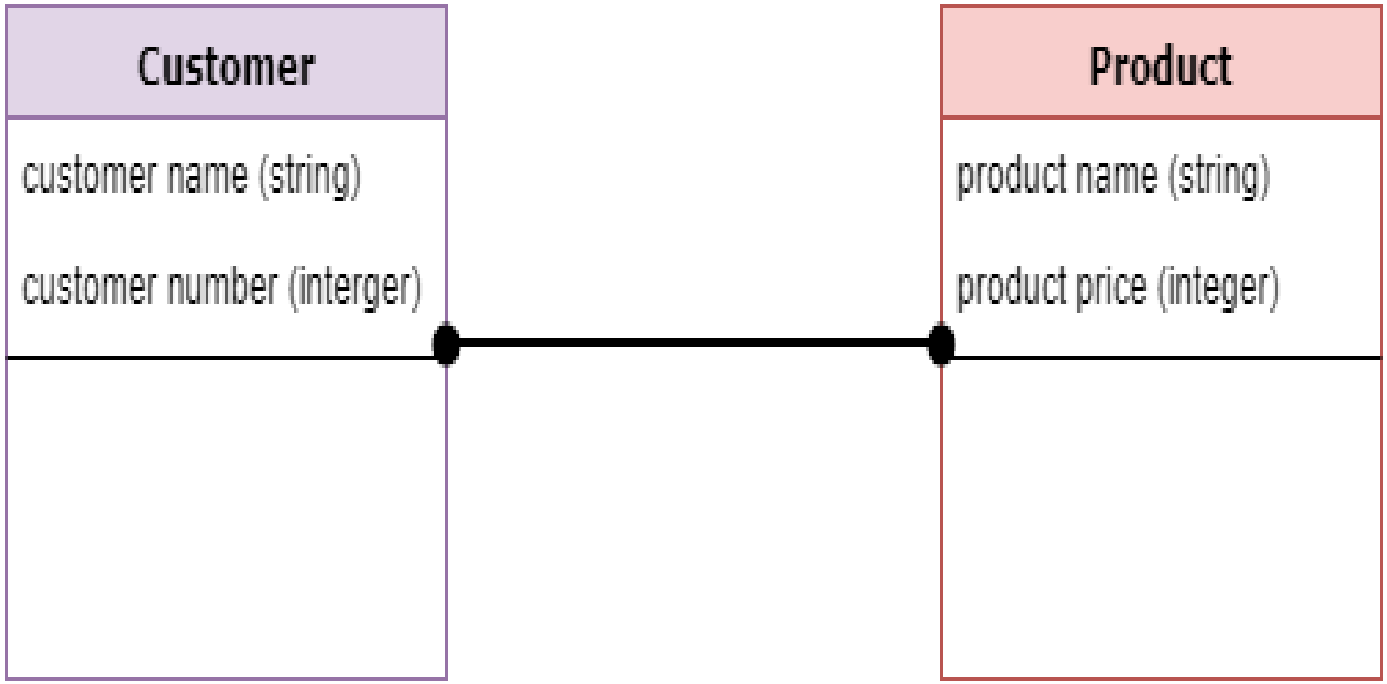
• The logical data model adds further information to the conceptual data model elements.

• The advantage of using a Logical data model is to provide a foundation to form the base for the Physical model. However, the modeling structure remains generic.

• At this Data Modeling level, no primary or secondary key is defined. At this Data modeling level, you need to verify and adjust the connector details that were set earlier for relationships.

Types of Data Model

2. Logical Data Model Example



Logical Data Model

Types of Data Model

3. Physical Data Model (Low level)

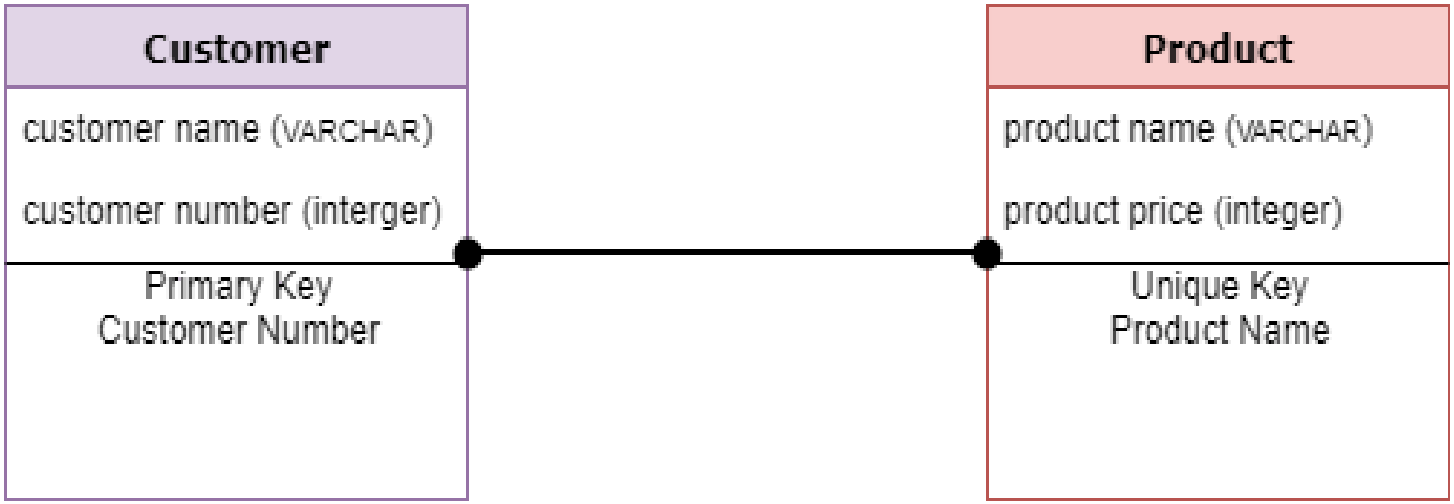
- A **Physical Data Model** describes a database-specific implementation of the data model.

- It offers database abstraction and helps generate the schema. This is because of the richness of meta-data offered by a Physical Data Model.

- The physical data model also helps in visualizing database structure by replicating database column keys, constraints, indexes, triggers, and other RDBMS features.

Types of Data Model

3. Physical Data Model Example



Physical Data Model

Types of Data Model

4. Entity-Relationship Data Model:

- An ER model is the logical representation of data as objects and relationships among them.

- These objects are known as entities, and relationship is an association among these entities.

- It was widely used in database designing. A set of attributes describe the entities.

- For example, student_name, student_id describes the 'student' entity. A set of the same type of entities is known as an 'Entity set', and the set of the same type of relationships is known as 'relationship set'.

Types of Data Model

5. Relational Data Model:

- **Relational Data Model:** This type of model designs the data in the form of rows and columns within a table.

- Thus, a relational model uses tables for representing data and in-between relationships.

- Tables are also called relations. This model was initially described by Edgar F. Codd, in 1969.

- The relational data model is the widely used model which is primarily used by commercial data processing applications.

Types of Data Model

6. Object-based Data Model:

- An extension of the ER model with notions of functions, encapsulation, and object identity, as well.

- This model supports a rich type system that includes structured and collection types. Thus, in 1980s, various database systems following the object-oriented approach were developed.

- Here, the objects are nothing but the data carrying its properties.

Data base Schema

- The overall design of a database is called schema.

- A database schema is the skeleton structure of the database. It represents the logical view of the entire database.

- A schema contains schema objects like table, foreign key, primary key, views, columns, data types, stored procedure, etc.

- A database schema can be represented by using the visual diagram. That diagram shows the database objects and relationship with each other.

- A database schema is designed by the database designers to help programmers whose software will interact with the database. The process of database creation is called data modeling.

Data base Schema

Example:

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

Database Instance

- It is important that we distinguish these two terms individually. Database schema is the skeleton of database.

- It is designed when the database doesn't exist at all. Once the database is operational, it is very difficult to make any changes to it.

- A database schema does not contain any data or information.

- **A database instance is a state of operational database with data at any given time. It contains a snapshot of the database.**

- Database instances tend to change with time. A DBMS ensures that its every instance (state) is in a valid state, by diligently following all the validations, constraints, and conditions that the database designers have imposed.

Database States

- 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.
- When we **define** a new database, we specify its database schema only to the DBMS.
- At this point, the corresponding database state is the empty state with no data.

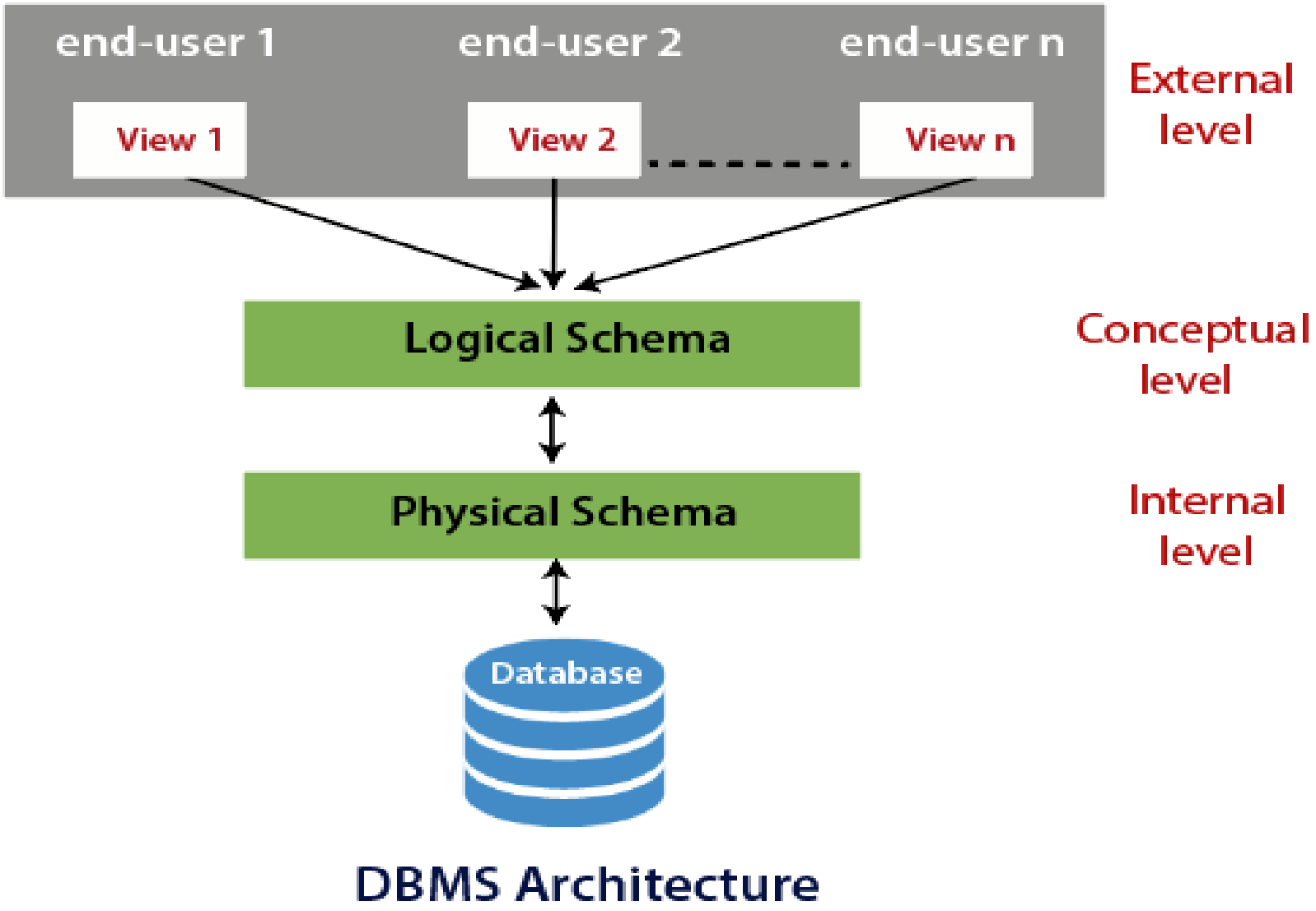
 Initial state - when the database is first **populated** or **loaded** with the initial data.

 From then on, every time an update operation is applied to the database, we get another database state. At any point in time, the database has a **current state.**

The DBMS is partly responsible for ensuring that every state of the database is a **valid state**—that is, a state that satisfies the structure and constraints specified in the schema.

Although, as mentioned earlier, the schema is not supposed to change frequently, it is not uncommon that changes occasionally need to be applied to the schema as the application requirements change. This is known as **schema evolution**

Three Tier Architecture



Three Tier Architecture

1. **External or View level:**

- This is the highest level of database abstraction.
- External or view level describes the actual view of data that is relevant to the particular user.
- This level also provides different views of the same database for a specific user or a group of users.
- An external view provides a powerful and flexible security mechanism by hiding the parts of the database from a particular user.

Three Tier Architecture

2. Conceptual or Logical level:

- The conceptual level describes the structure of the whole database.
- This level acts as a middle layer between the physical storage and user view.
- It explains what data to be stored in the database, what relationship exists among those data, and what the datatypes are.
- There is only one conceptual schema per database.

Three Tier Architecture

2. Conceptual or Logical level:

- Database administrator and the programmers work at this level.
- This level does not provide any access or storage details but concentrates on the relational model of the database.
- The conceptual schema also includes features that specify the checks to retain integrity and consistency.

Three Tier Architecture

3. Internal or Physical level:

- This is the lowest level of database abstraction.
- It describes how the data is actually stored in the database and provides methods to access data from the database.
- It allows viewing the physical representation of the database on the computer system.
- The interface between the conceptual schema and the internal schema identifies how an element in the conceptual schema is stored and how it may be accessed.

Three Tier Architecture

3. Internal or Physical level:

- If there is any change in the internal or physical schema, it needs to be addressed to the interface between the conceptual and internal schema.

- But there is no need to change in the interface of a conceptual and external schema.

- It means that the changes in physical storage devices such as hard disks, and the files organized on storage devices, are transparent to application programs and users.

Advantages of Three Tier Architecture

- This architecture makes the database abstract. It is used to hide the details of how data is physically stored in a computer system, which makes it easier to use for a user.

- This architecture allows each user to access the same database with a different customized view of data.

- This architecture enables a database admin to change the storage structure of the database without affecting the user currently on the system.

Data Independence

- Data independence can be explained using the three-schema architecture.

- Data independence refers characteristic of being able to modify the schema at one level of the database system without altering the schema at the next higher level.

Data Independence

1. **Logical Data Independence**

- Logical data independence refers characteristic of being able to change the conceptual schema without having to change the external schema.

- Logical data independence is used to separate the external level from the conceptual view.

- If we do any changes in the conceptual view of the data, then the user view of the data would not be affected.

- Logical data independence occurs at the user interface level.

Data Independence

2. Physical Data Independence

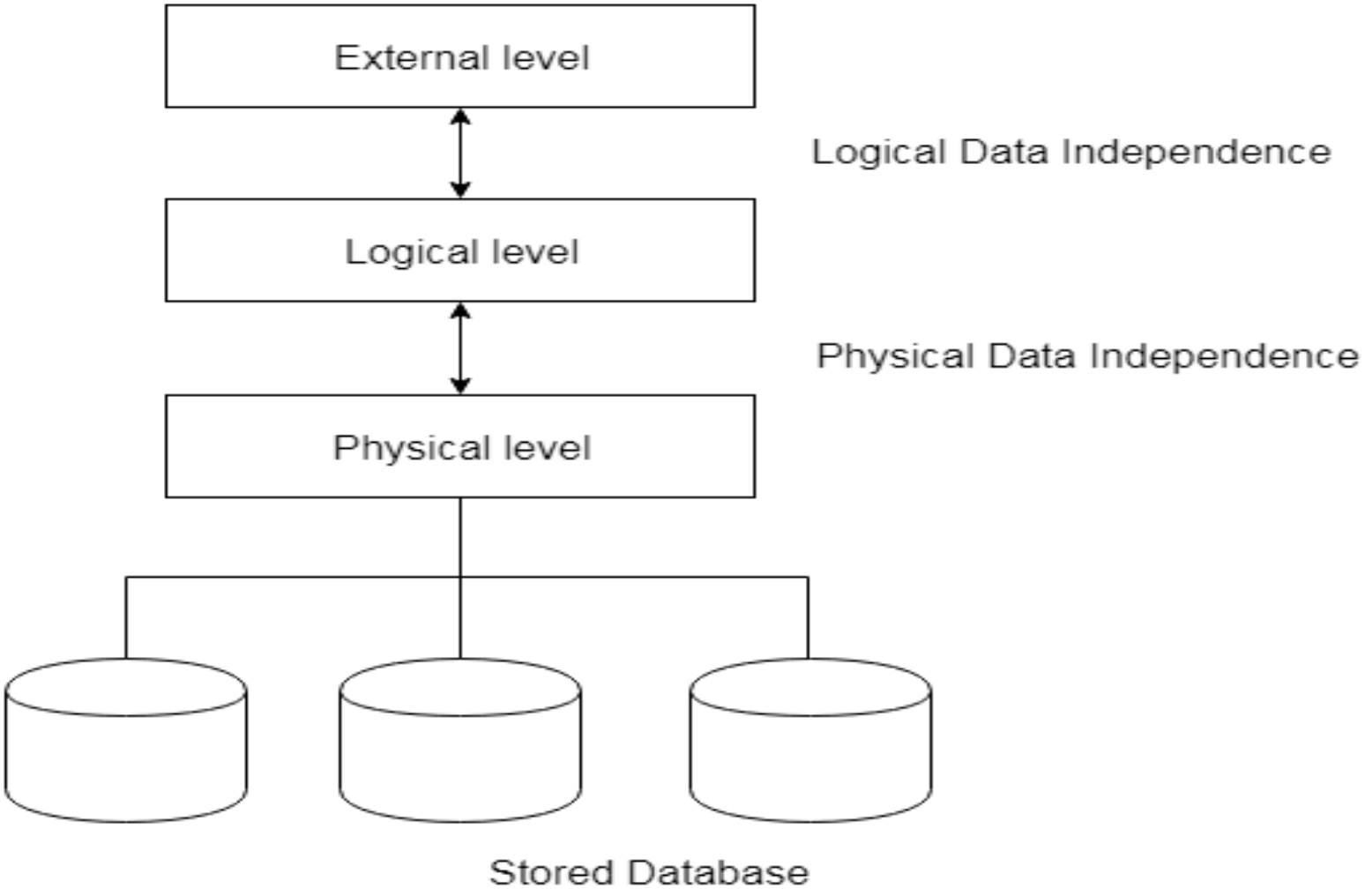
- Physical data independence can be defined as the capacity to change the internal schema without having to change the conceptual schema.

- If we do any changes in the storage size of the database system server, then the Conceptual structure of the database will not be affected.

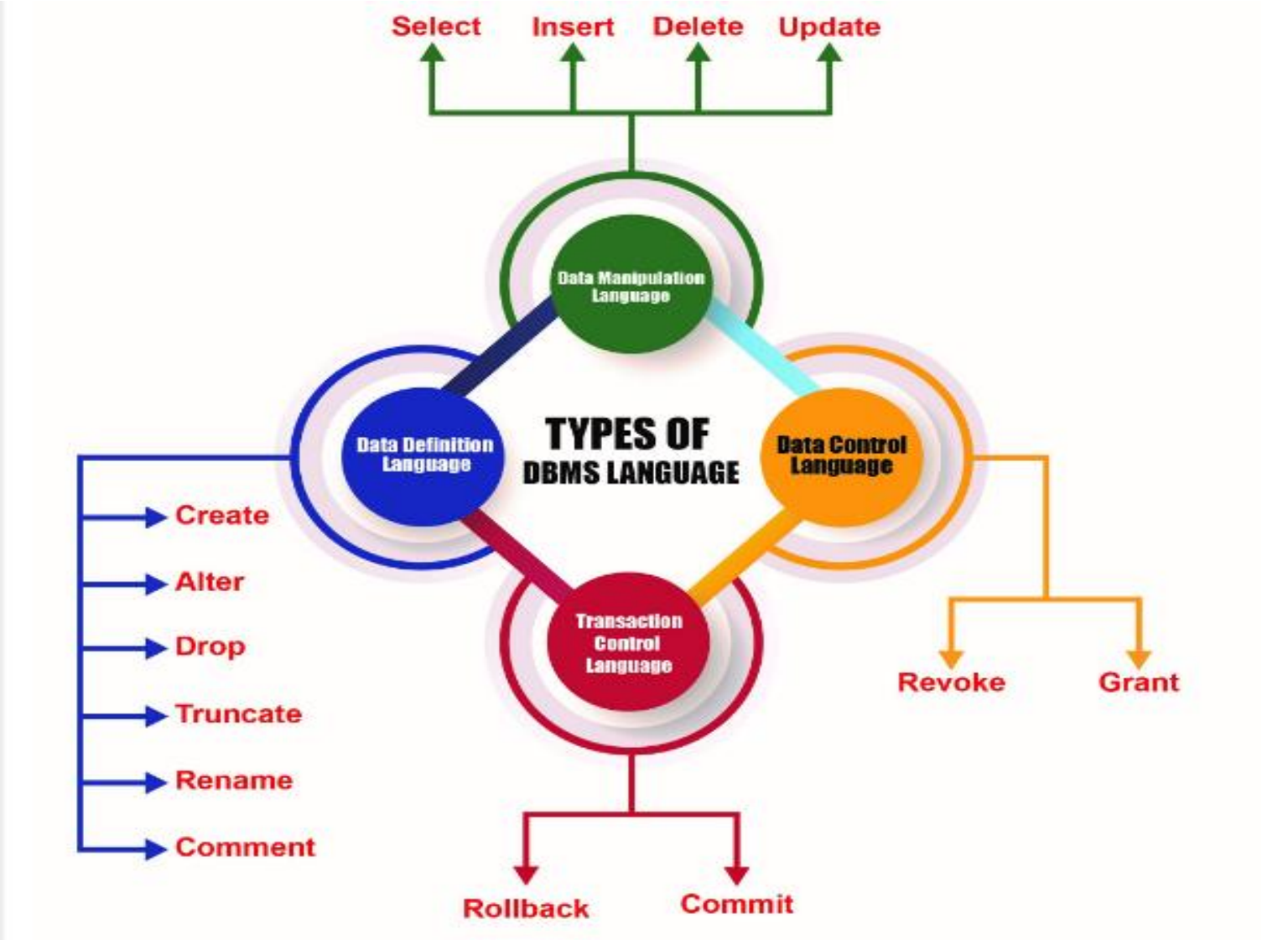
- Physical data independence is used to separate conceptual levels from the internal levels.

- Physical data independence occurs at the logical interface level.

Data Independence



Database Languages



Database Languages

1. **Data Definition Language**

- **DDL** stands for **D**ata **D**efinition **L**anguage. 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, you can create the skeleton of the database.

- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Database Languages

1. **Data Definition Language**

Here are some tasks that come 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 remove all records from a table.
- **Rename:** It is used to rename an object.
- **Comment:** It is used to comment on the data

Database Languages

2. Data Manipulation Language

DML stands for **D**ata **M**anipulation **L**anguage. It is used for accessing and manipulating data in a database. It handles user requests.

Here are some tasks that come under DML:

- **Select:** It is used to retrieve data from a database.
- **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 all records from a table.
- **Merge:** It performs UPSERT operation, i.e., insert or update operations.
- **Call:** It is used to call a structured query language or a Java subprogram.
- **Explain Plan:** It has the parameter of explaining data.
- **Lock Table:** It controls concurrency.

Database Languages

3. Data Control Language

DCL stands for **D**ata **C**ontrol **L**anguage. 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 data control language does not have the feature of rolling back.)

Here are some tasks that come under DCL:

Grant: It is used to give user access privileges to a database.

Revoke: It is used to take back permissions from the user.

These are the following operations which have the authorization of Revoke: CONNECT, INSERT, USAGE, EXECUTE, DELETE, UPDATE and SELECT.

Database Languages

4. Transaction Control Language

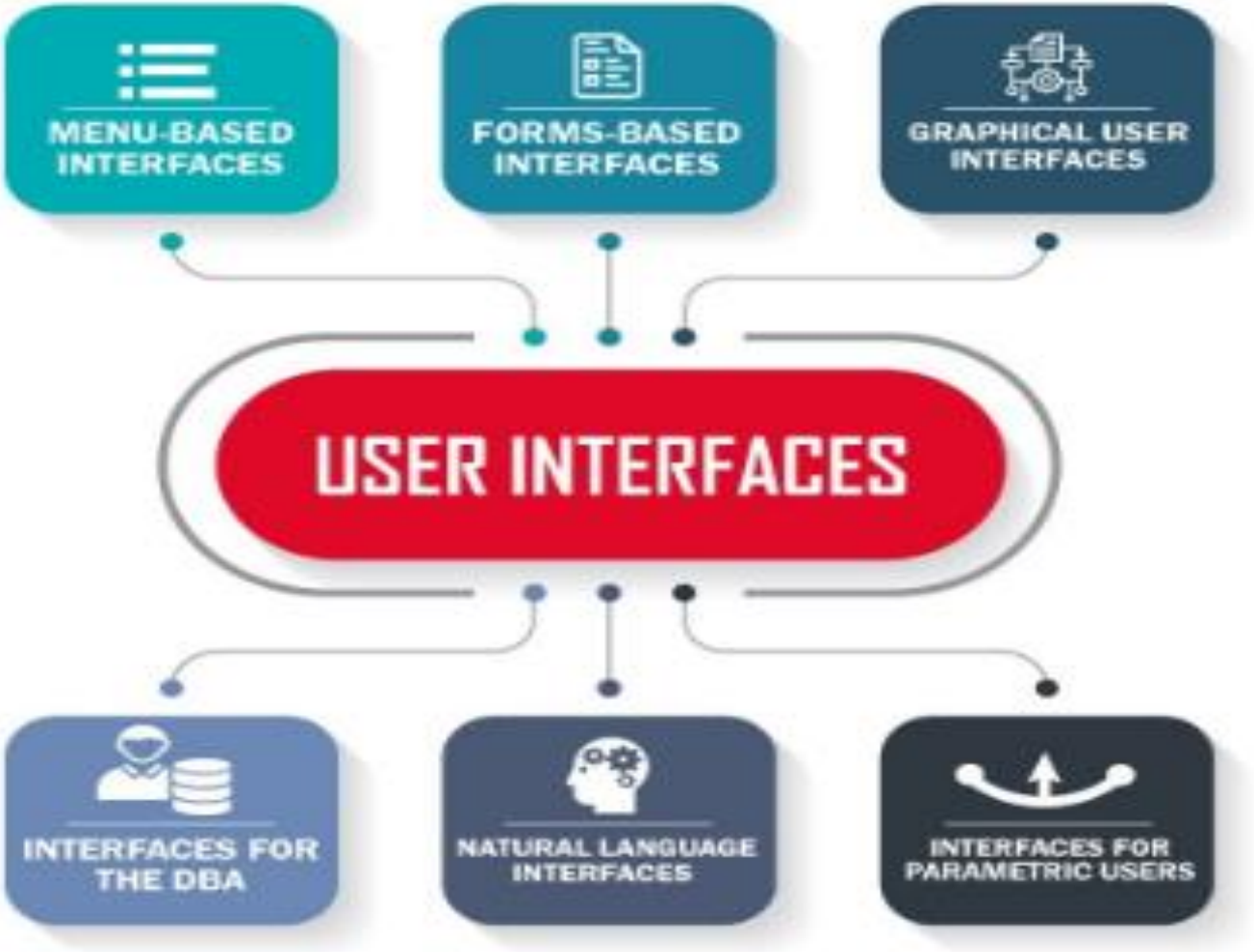
TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

Here are some tasks that come under TCL:

Commit: It is used to save the transaction on the database.

Rollback: It is used to restore the database to original since the last Commit.

Database Interfaces



Database Interfaces

1. Menu-Based Interfaces for Web Clients or Browsing – These interfaces present the user with lists of options (called menus) that lead the user through the formation of a request.

- Basic advantage of using menus is that they removes the tension of remembering specific commands and syntax of any query language, rather than query is basically composed step by step by collecting or picking options from a menu that is basically shown by the system.

- Pull-down menus are a very popular technique in Web based interfaces. They are also often used in browsing interface which allow a user to look through the contents of a database in an exploratory and unstructured manner.

Database Interfaces

2. Forms-Based Interfaces –

• A forms-based interface displays a form to each user.

• Users can fill out all of the form entries to insert a new data, or they can fill out only certain entries, in which case the DBMS will redeem same type of data for other remaining entries.

• This type of forms are usually designed or created and programmed for the users that have no expertise in operating system.

• Many DBMSs have forms specification languages which are special languages that help specify such forms.

Database Interfaces

3. Graphical User Interface –

• A GUI typically displays a schema to the user in diagrammatic form.

• The user then can specify a query by manipulating the diagram. In many cases, GUI's utilize both menus and forms.

• Most GUIs use a pointing device such as mouse, to pick certain part of the displayed schema diagram.

Database Interfaces

4. Natural language Interfaces –

These interfaces accept request written in English or some other language and attempt to understand them.

- A Natural language interface has its own schema, which is similar to the database conceptual schema as well as a dictionary of important words.
- The natural language interface refers to the words in its schema as well as to the set of standard words in a dictionary to interpret the request.
- If the interpretation is successful, the interface generates a high-level query corresponding to the natural language and submits it to the DBMS for processing, otherwise a dialogue is started with the user to clarify any provided condition or request.
- The main disadvantage with this is that the capabilities of this type of interfaces are not that much advance.

Database Interfaces

5. Interfaces for DBA –

- Most database system contains privileged commands that can be used only by the DBA's staff.

- These include commands for creating accounts, setting system parameters, granting account authorization, changing a schema, reorganizing the storage structures of a databases.

Database Interfaces

6. Interfaces for Parametric Users.

- Parametric users, such as bank tellers, often have a small set of operations that they must perform repeatedly.

- For example, a teller is able to use single function keys to invoke routine and repetitive transactions such as account deposits or withdrawals, or balance inquiries.

- Systems analysts and programmers design and implement a special interface for each known class of naive users. Usually a small set of abbreviated commands is included, with the goal of minimizing the number of keystrokes required for each request.

- For example, function keys in a terminal can be programmed to initiate various commands. This allows the parametric user to proceed with a minimal number of keystrokes.

Database Dictionary

Data Dictionary consists of database metadata. It has records about objects in the database.

What Data Dictionary consists of

Data Dictionary consists of the following information −

- Name of the tables in the database
- Constraints of a table i.e. keys, relationships, etc.
- Columns of the tables that related to each other
- Owner of the table
- Last accessed information of the object
- Last updated information of the object

Database Dictionary

An example of Data Dictionary can be personal details of a student −

Example

<StudentPersonalDetails>

Student_ID	Student_Name	Student_Address	Student_City

The following is the data dictionary for the above fields −

Field Name	Datatype	Field Length	Constraint	Description
Student_ID	Number	5	Primary Key	Student id
Student_Name	Varchar	20	Not Null	Name of the student
Student_Address	Varchar	30	Not Null	Address of the student
Student_City	Varchar	20	Not Null	City of the student

Types Database Dictionary

Types of Data Dictionary

Here are the two types of data dictionary −

1. Active Data Dictionary

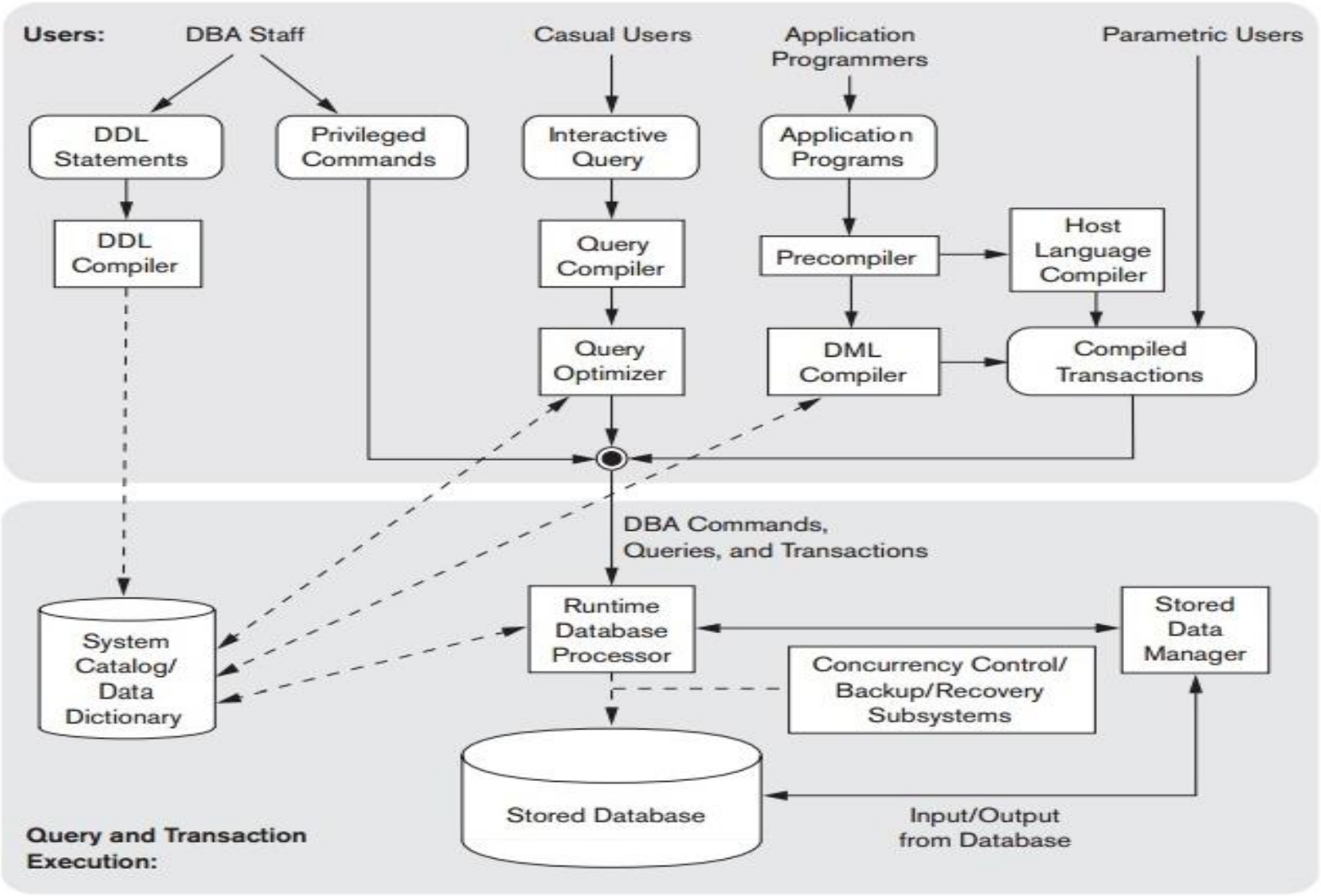
- The DBMS software manages the active data dictionary automatically.

- The modification is an automatic task and most RDBMS has active data dictionary. It is also known as integrated data dictionary.

2. Passive Data Dictionary

Managed by the users and is modified manually when the database structure change. Also known as non-integrated data dictionary.

Database System Environment



Database System Environment

- The figure is divided into two halve. The top half of the diagram refers to the various users of the database environment and their interfaces.

- The lower half demonstrates the internals of the DBMS responsible for storage of data and processing of transaction.

- The database and the DBMS catalogue are usually stored on disk. Access to the disk is principally controlled by operating system (OS).

- This includes disk input/Output. A higher point stored data manager module of DBMS controls access to DBMS information that is stored on the disk.

Database System Environment

- If we consider the top half of the figure it shows interface to casual users, DBA staff, application programmers and parametric users.

- The DDL compiler specified in the DDL, processes schema definitions as well as stores the description of the schema in the DBMS Catalogue.

- The catalogue includes information such as names and sizes of the sizes of the files and data types of data of data items.

- Storage particulars of every file mapping information among schemas as well as constraints.

Database System Environment

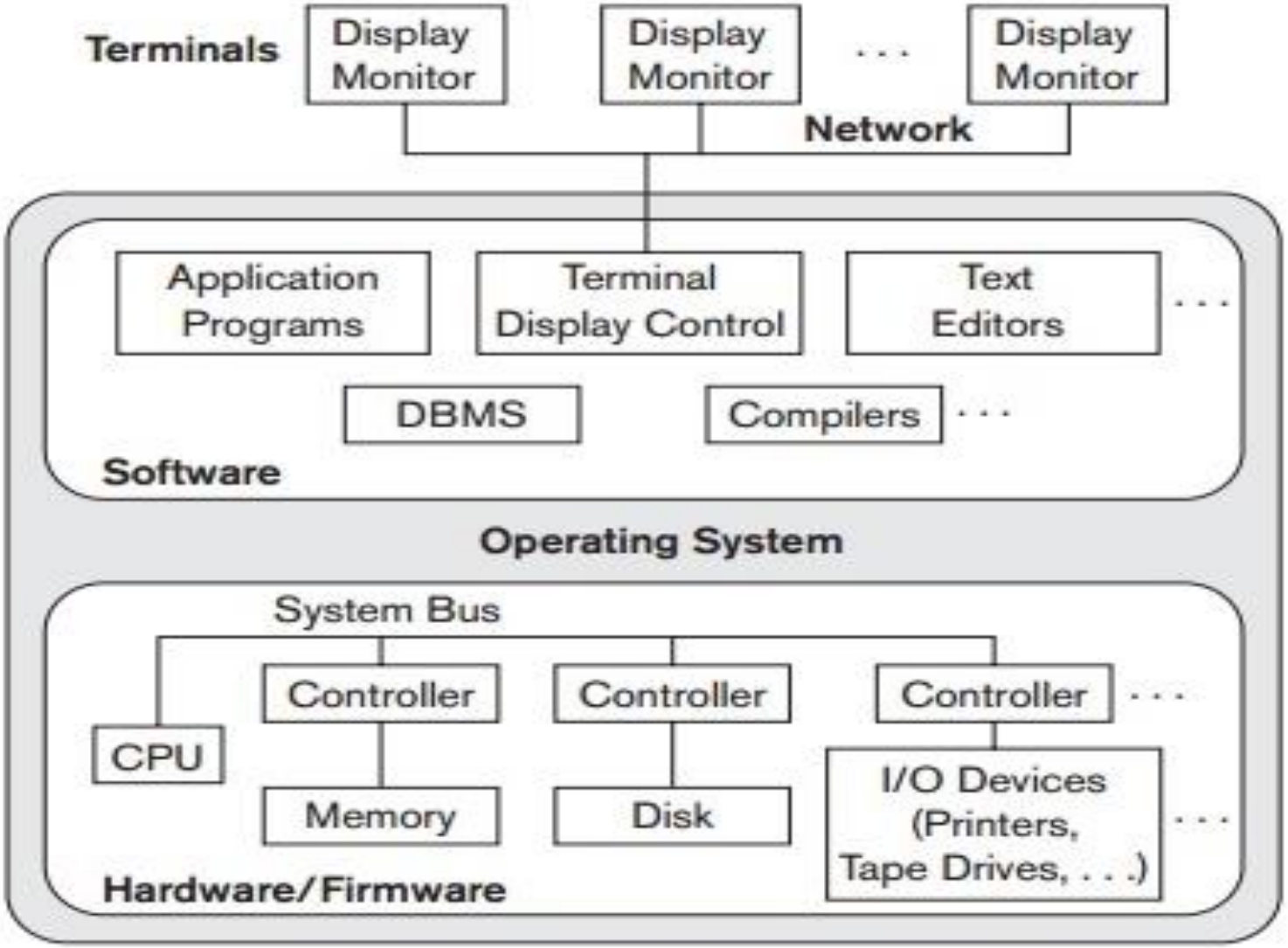
- Casual users as well as persons with occasional need of information from database interact using some of interface which is interactive query interface.

- The queries are parsed analyze for correctness of the operations for the model. The names of the data elements as well as therefore on by a query compiler that compiles them into internal form.

- The interior query is subjected to query optimization. The query optimizer is worried with rearrangement and possible recording of operations and eliminations of redundancies.

- Application programmer inscribes programs in host languages. The precompiled take out DML commands from an application program.

Centralized DBMS Architecture



Centralized DBMS Architecture

- Architectures for DBMSs have followed trends similar to those for general computer system architectures.

- Earlier architectures used mainframe computers to provide the main processing for all system functions, including user application programs and user interface programs, as well as all the DBMS functionality.

- The reason was that most users accessed such systems via computer terminals that did not have processing power and only provided display capabilities.

- Therefore, all processing was performed remotely on the computer system, and only display information and controls were sent from the computer to the display terminals, which were connected to the central computer via various types of communications networks.

Centralized DBMS Architecture

- As prices of hardware declined, most users replaced their terminals with PCs and workstations.

- At first, database systems used these computers similarly to how they had used display terminals, so that the DBMS itself was still a **centralized** DBMS in which all the DBMS functionality, application program execution, and user inter-face processing were carried out on one machine.

- Diagram illustrates the physical components in a centralized architecture. Gradually, DBMS systems started to exploit the available processing power at the user side, which led to client/server DBMS architectures.

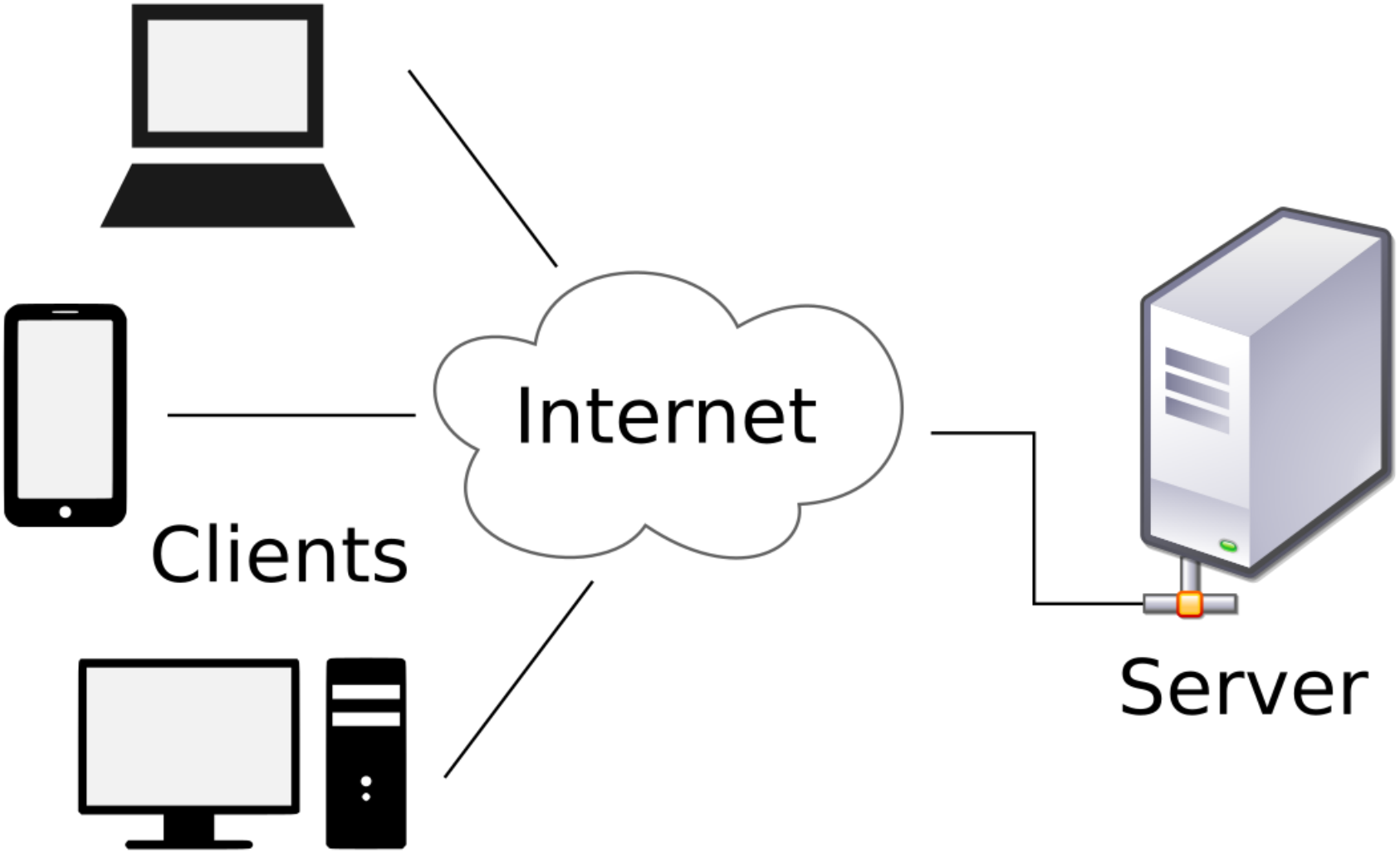
Client ServerArchitecture

Client-server architecture, architecture of a computer network in which many clients (remote processors) request and receive service from a centralized server (host computer).

Client computers provide an interface to allow a computer user to request services of the server and to display the results the server returns. Servers wait for requests to arrive from clients and then respond to them.

Ideally, a server provides a standardized transparent interface to clients so that clients need not be aware of the specifics of the system (i.e., the hardware and software) that is providing the service. Clients are often situated at workstations or on personal computers, while servers are located elsewhere on the network, usually on more powerful machines.

Client ServerArchitecture



Types of Client Server Architecture

- 1 tier architecture
- 2 tier architecture
- 3 tier architecture

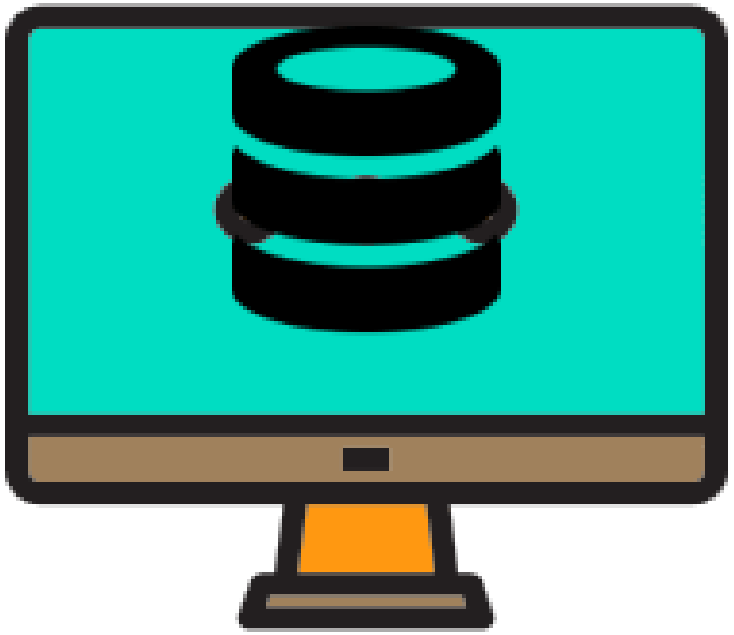
1 Tier Architecture

- **1 Tier Architecture** in DBMS is the simplest architecture of Database in which the client, server, and Database all reside on the same machine.

- A simple one tier architecture example would be anytime you install a Database in your system and access it to practice SQL queries.

- But such architecture is rarely used in production.

1 Tier Architecture



Single Tier Architecture

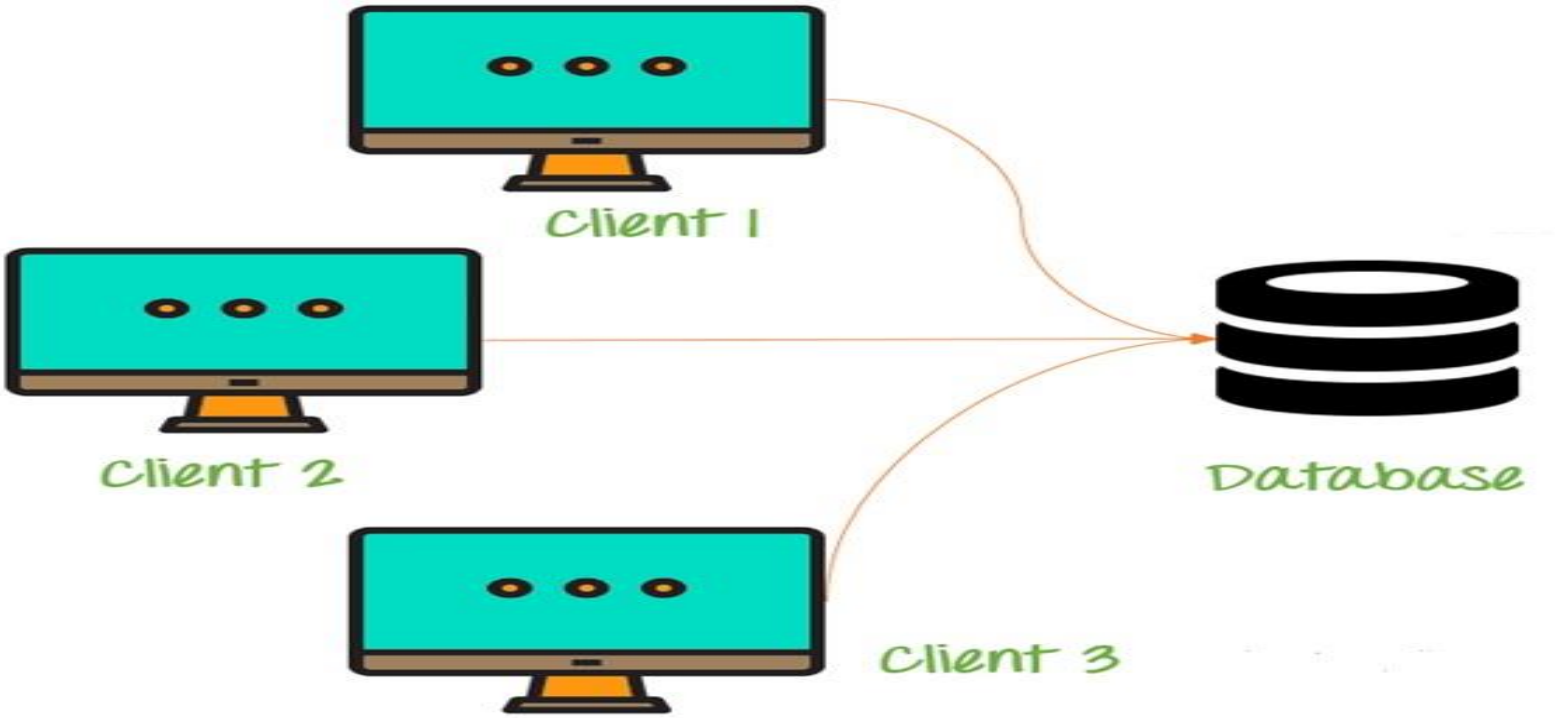
2 Tier Architecture

- A **2 Tier Architecture** in DBMS is a Database architecture where the presentation layer runs on a client (PC, Mobile, Tablet, etc.), and data is stored on a server called the second tier.

- Two tier architecture provides added security to the DBMS as it is not exposed to the end-user directly.

- It also provides direct and faster communication.

2 Tier Architecture



In the above 2 Tier client-server architecture of database management system, we can see that one server is connected with clients 1, 2, and 3.

Two Tier Architecture Example:

A Contact Management System created using MS- Access.

Two-Tier Client/Server Architectures for DBMSs

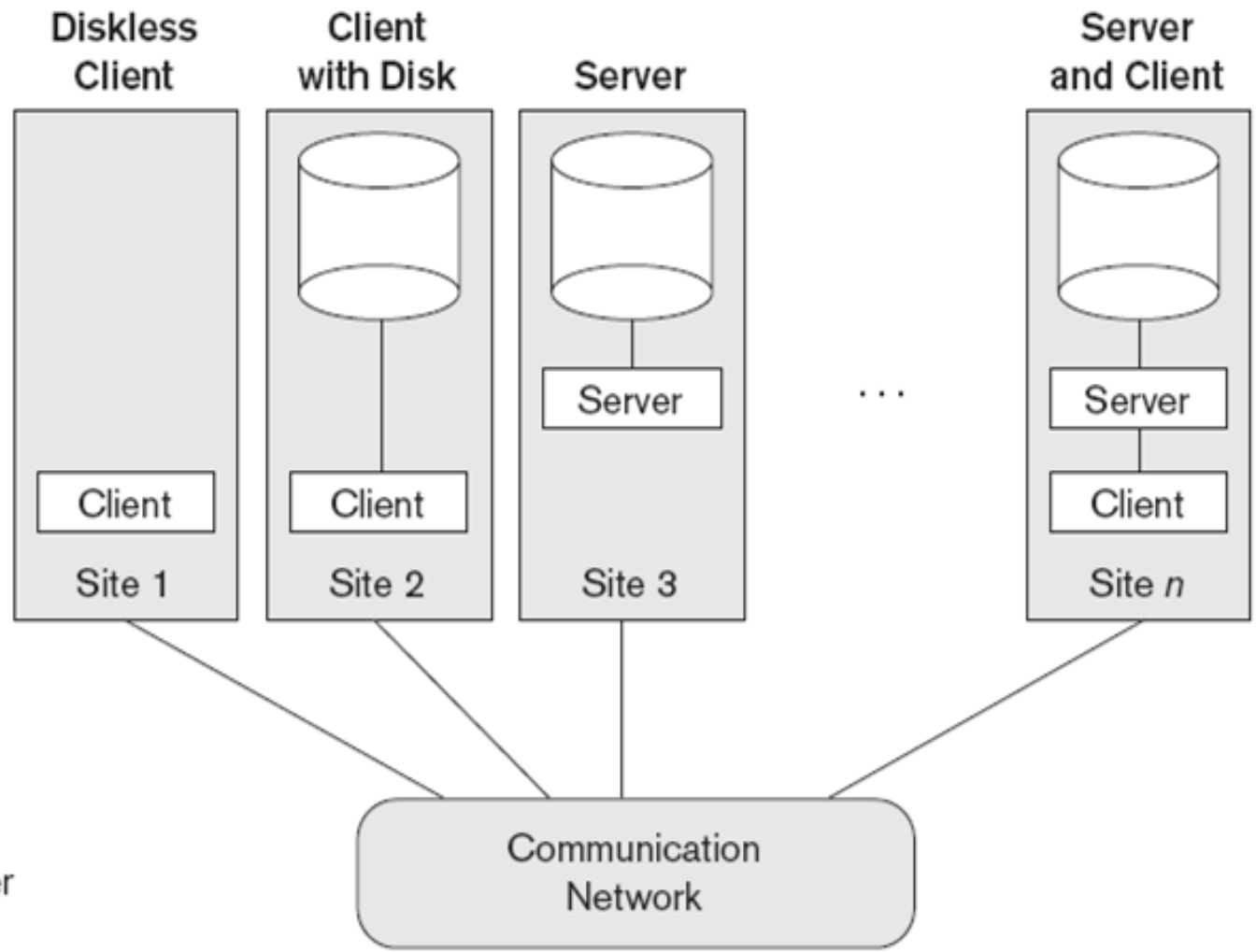


Figure 2.6
Physical two-tier client/server architecture.

3 Tier Architecture

- A **3 Tier Architecture** in DBMS is the most popular client server architecture in DBMS in which the development and maintenance of functional processes, logic, data access, data storage, and user interface is done independently as separate modules.

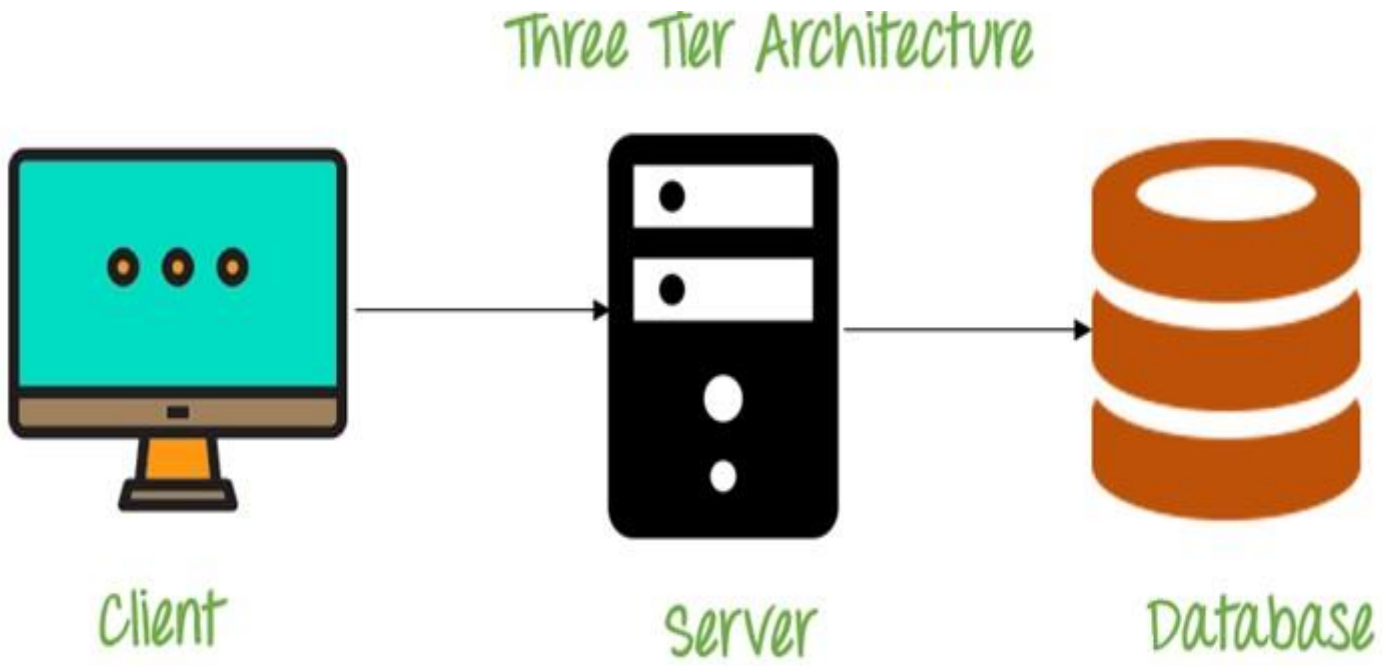
- Three Tier architecture contains a presentation layer, an application layer, and a database server.

- 3-Tier database Architecture design is an extension of the 2-tier client-server architecture.

A 3-tier architecture has the following layers:

- Presentation layer (your PC, Tablet, Mobile, etc.)
- Application layer (server)
- Database Server

3 Tier Architecture



3 Tier Architecture

- The Application layer resides between the user and the DBMS, which is responsible for communicating the user's request to the DBMS system and send the response from the DBMS to the user.

- The application layer(business logic layer) also processes functional logic, constraint, and rules before passing data to the user or down to the DBMS.

The goal of Three Tier client-server architecture is:

- To separate the user applications and physical database
- To support DBMS characteristics
- Program-data independence
- Supporting multiple views of the data

Three Tier Architecture Example:

Any large website on the internet, Flip kart, IRCTC

Three-Tier and n-Tier Architectures for Web Applications

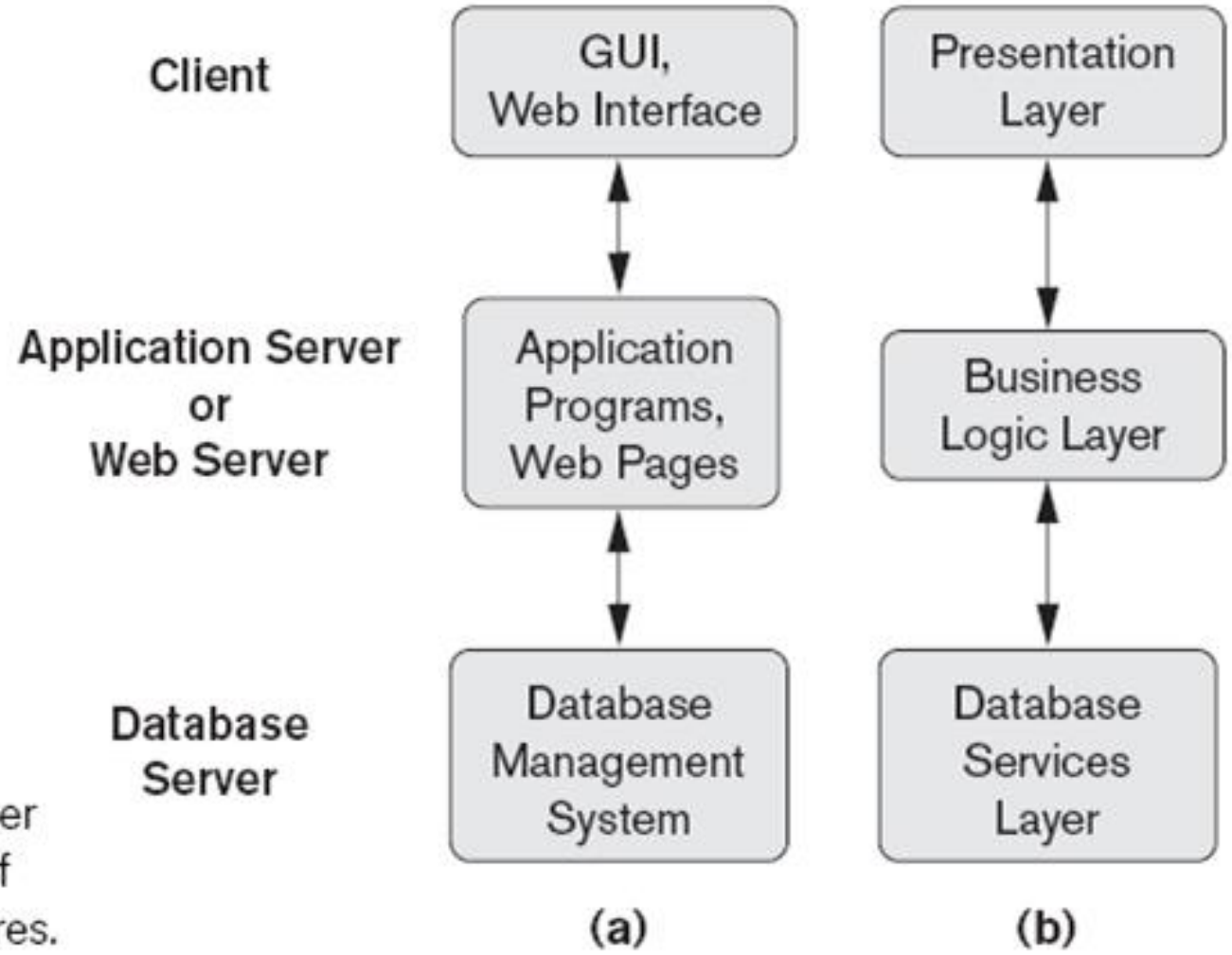


Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

Classification of Database Management Systems

- The first is the **data model** on which the DBMS is based.

- The main data model used in many current commercial DBMSs is the **relational data model**.

- The **object data model** has been implemented in some commercial systems but has not had widespread use.

- Many legacy applications still run on database systems based on the **hierarchical** and **network data models**. Examples of hierarchical DBMSs include IMS (IBM) and some other sys-tems like System 2K (SAS Inc.) and TDMS. IMS is still used at governmental and industrial installations, including hospitals and banks, although many of its users have converted to relational systems.

Classification of Database Management Systems

- The relational DBMSs are evolving continuously, and, in particular, have been incorporating many of the concepts that were developed in object databases. This has led to a new class of DBMSs called **object-relational DBMS**s. We can categorize DBMSs based on the data model: relational, object, object-relational, hierarchical, network, and other.

- More recently, some experimental DBMSs are based on the XML (eXtended Markup Language) model, which is a tree-structured (hierarchical) data model. These have been called **native XML DBMSs.** Several commercial relational DBMSs have added XML interfaces and storage to their products.

Classification of Database Management Systems

- The second criterion used to classify DBMSs is the number of users supported by the system.

- Single-user systems support only one user at a time and are mostly used with PCs.

- Multiuser systems, which include the majority of DBMSs, support con-current multiple users.

Classification of Database Management Systems

- The third criterion is the **number of sites** over which the database is distributed.

- A 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 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.

Classification of Database Management Systems

- **Homogeneous** DDBMSs use the same DBMS software at all the sites, whereas **heterogeneous** DDBMSs can use different DBMS software at each site.

- It is also possible to develop **middleware software** to access several autonomous preexisting databases stored under heterogeneousDBMSs.

- This leads to a **federated** DBMS (or **multidatabase system**), in which the participating DBMSs are loosely coupled and have a degree of local autonomy. Many DDBMSs use client-server architecture **.**

Classification of Database Management Systems

- **Homogeneous** DDBMSs use the same DBMS software at all the sites, whereas **heterogeneous** DDBMSs can use different DBMS software at each site.

- It is also possible to develop **middleware software** to access several autonomous preexisting databases stored under heterogeneous DBMSs.

- This leads to a **federated** DBMS (or **multi database system**), in which the participating DBMSs are loosely coupled and have a degree of local autonomy. Many DDBMSs use client-server architecture **.**

Classification of Database Management Systems

- The fourth criterion is cost. It is difficult to propose a classification of DBMSs based on cost.

- Today we have open source (free) DBMS products like MySQL and PostgreSQL that are supported by third-party vendors with additional services.

- The main RDBMS products are available as free examination 30-day copy versions as well as personal versions, which may cost under $100 and allow a fair amount of functionality.

- The giant systems are being sold in modular form with components to handle distribution, replication, parallel processing, mobile capability, and so on, and with a large number of parameters that must be defined for the configuration.

Classification of Database Management Systems

- Furthermore, they are sold in the form of licenses—site licenses allow unlimited use of the database system with any number of copies running at the customer site.

- Another type of license limits the number of concurrent users or the number of user seats at a location. Standalone single user versions of some systems like Microsoft Access are sold per copy or included in the overall configuration of a desktop or laptop.

- In addition, data warehousing and mining features, as well as support for additional data types, are made available at extra cost. It is possible to pay millions of dollars for the installation and maintenance of large database systems annually.

Classification of Database Management Systems

- We can also classify a DBMS on the basis of the **types of access path** options for storing files. One well-known family of DBMSs is based on inverted file structures. Finally, a 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.