

Introduction to Database Systems

Database Management Systems

Introduction: Database System, Characteristics (Database Vs File System), Advantages of Database Systems, Database Applications, Database Users, Brief Introduction of Different Data Models; Concepts of Schema, Instance and Data Independence; Three Tier Schema Architecture for Data Independence; Database System Structure, Centralized and Client Server Architecture for Database Systems.

Entity Relationship Model: Introduction, Concept of Entities, Attributes, Entity Sets, Relationships, Relationship Sets, Key and Participation Constraints, Class Hierarchies, and Aggregation, Developing E-R diagrams for Databases.

Definition of Data

★ **Data:** Raw, unprocessed facts

Ex: 25, Suresh, Bangalore

★ **Information:** Processed data

Ex: The age of Suresh is 25.

★ **Database:** Collection of **related** data

Ex: Online banking system, library management system

★ **Meta-data:** The database definition

❖ FILE SYSTEM APPROACH



❖ DBMS APPROACH



Applications Of DBMS



Database Applications Examples

Enterprise Information

Sales: customers, products, purchases

Accounting: payments, receipts, assets

Human Resources: Information about employees, salaries, payroll taxes.

Manufacturing: management of production, inventory, orders, supply chain.

Banking and finance

customer information, accounts, loans, and banking transactions.

Credit card transactions

Finance: sales and purchases of financial instruments (e.g., stocks and bonds;
storing real-time market data

Universities: registration, grades

Database Applications Examples

Airlines: reservations, schedules

Telecommunication: records of calls, texts, and data usage, generating monthly bills, maintaining balances on prepaid calling cards

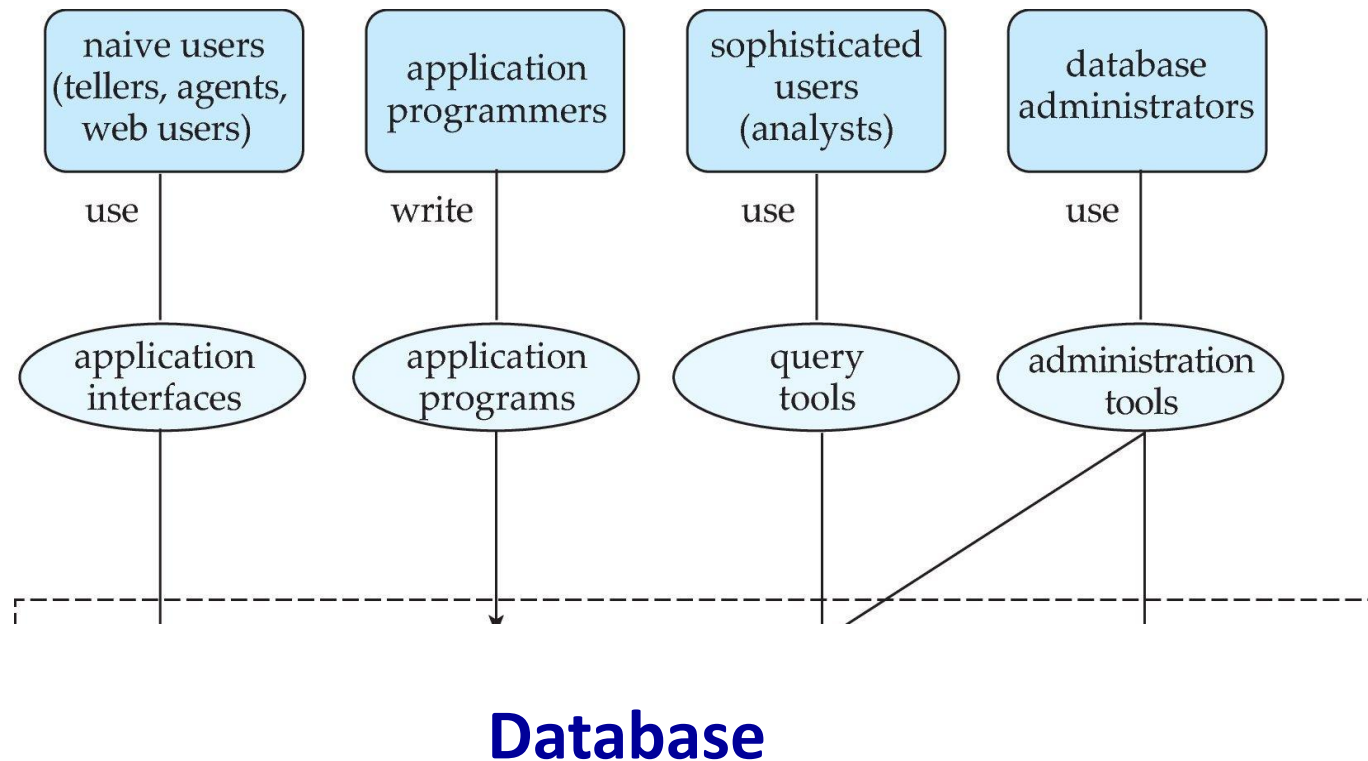
Web-based services

Online retailers: order tracking, customized recommendations

Online advertisements

Navigation systems: For maintaining the locations of various places of interest along with the exact routes of roads, train systems, buses, etc.

Database Users and Administrators



History of Database Systems

First generation

Hierarchical model

Information Management System (IMS)

Network model

Conference on Data System Languages (CODASYL)

Data Base Task Group (DBTG)

Second generation

Relational model

E. F. Codd

DB2, Oracle

Third generation

Object-relational DBMS

Object-oriented DBMS

Historical Development and Database Technology

Early Database Applications: Hierarchical and Network Models (in mid 1960's).

Relational Model based Systems: Researched and experimented with in IBM and the universities (in 1970).

Object-oriented applications: OODBMSs (in late 1980's and early 1990's)

Types of Databases

There are various types of databases used for storing different varieties of data:



Data Models

60's

Hierarchical

Network

70's

80's

Relational

*Choice for most
applications today*

90's



Object Models

Knowledge Bases, Rules

Present

History of

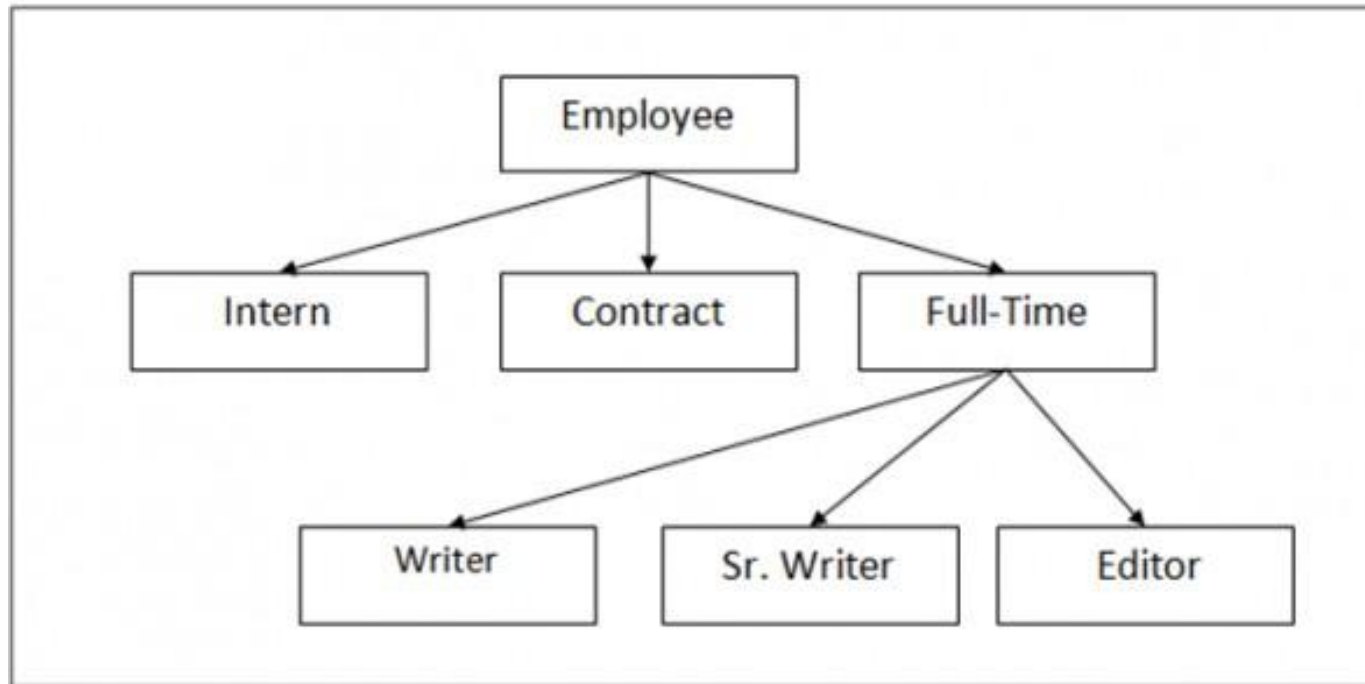
Direct Access Data Model: implemented in a joint effort by IBM and North American Rockwell around 1965. Resulted in the IMS family of systems. The most popular model. Other system based on this model: System 2k (SAS inc.)

Network Model: the first one to be implemented by Honeywell in 1964-65 (IDS System). Adopted heavily due to the support by CODASYL (CODASYL - DBTG report of 1971). Later implemented in a large variety of systems - IDMS (Cullinet - now CA), DMS 1100 (Unisys), IMAGE (H.P.), VAX -DBMS (Digital Equipment Corp.).

Relational Model: proposed in 1970 by E.F. Codd (IBM), first commercial system in 1981-82. Now in several commercial products (DB2, ORACLE, SQL Server, SYBASE, INFORMIX)

Hierarchical Data Model

- In Hierarchical Model, a hierarchical relation is formed by collection of relations and forms a tree-like structure.
- The relationship can be defined in the form of parent child type. One of the first and most popular Hierarchical Model is Information Management System (IMS), developed by IBM

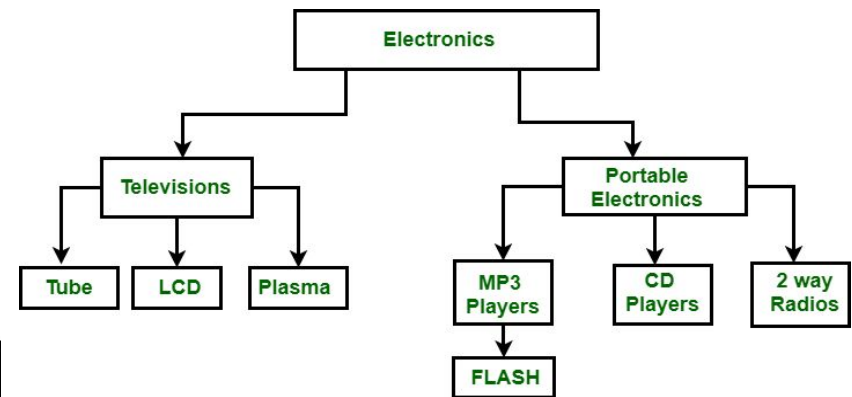
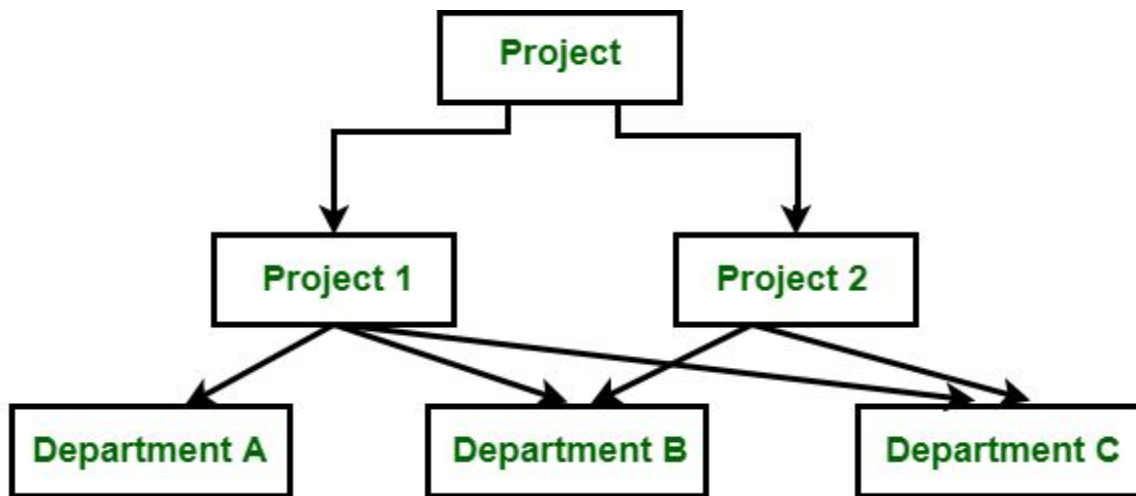


Hierarchical Model:

In Hierarchical Model, data are represented by records and relationships among data are represented by links. But unlike in Network model, data are organized in an ordered tree structure, which is called Hierarchical structure.

What is the Hierarchical Model in DBMS?

The Hierarchical Model was the first database management system model. This concept uses a hierarchical tree structure to organize the data. The hierarchy begins at the root, which contains root data, and then grows into a tree as child nodes are added to the parent node



Features of a Hierarchical Model

1. Parent-Child Relationship

A parent node exists for each child node. However, a parent node might have several child nodes. It is not permitted to have more than one parent.

2. One-to-many Relationship

The data is organized in a tree-like form, with the data types having a one-to-many relationship. There can only be one path from any node to its parent.

3. Deletion Problem

When a parent node is removed, the child node is removed as well.

4. Pointers

Pointers are used to connect the parent and child nodes and to traverse and navigate between the stored data

Merits and De-Merits of Hierarchical Data Model

■ Merits

- The design of the hierarchical model is simple.
- Provides Data Integrity since it is based on parent/child relationship
- Data sharing is feasible since the data is stored in a single database.
- Even for large volumes of data, this model works perfectly.

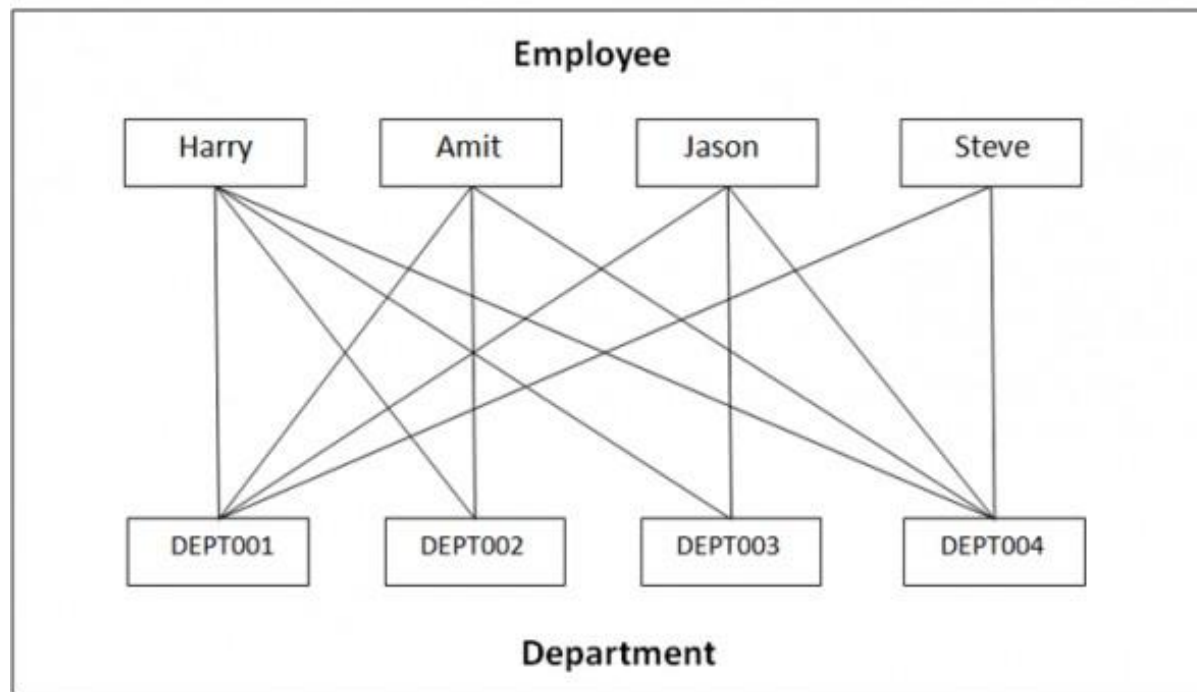
■ De-Merits

- Implementation is complex.
- This model has to deal with anomalies like Insert, Update and Delete.
- Maintenance is difficult since changes done in the database may want you to do changes in the entire database structure.

Network Data Model

- The Hierarchical Model creates hierarchical tree with parent/child relationship, whereas the Network Model has graph and links.
- links.

The relationship can be defined in the form of links and it handles many-to-many relations. This itself states that a record can have more than one parent.

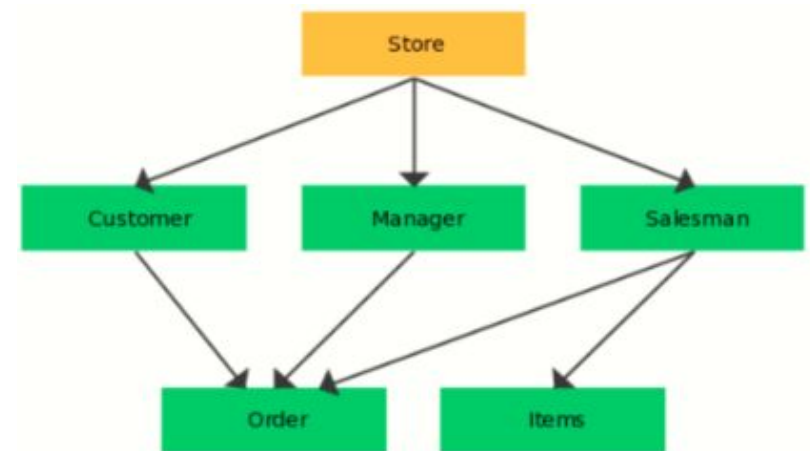
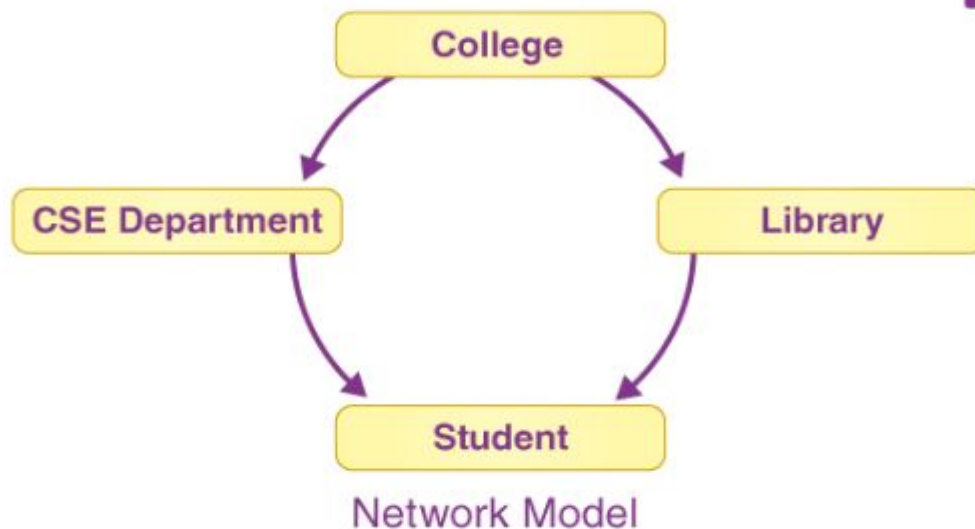


Network Model:

The network model uses two different structures. The data are represented by a collection of records and the relationships among data are represented by links.

What is the Network Model in DBMS?

The hierarchical model is extended in the network model. Prior to the relational model, it was the most popular model. To increase database performance and standards, the network model was devised to express complicated data relationships more effectively than hierarchical models.



Merits and De-Merits of Network Model

■ Merits :

- Easy to design the Network Model
- The model can handle one-one, one-to-many, many-to-many relationships.
- It isolates the program from other details. Based on standards and conventions.

■ De-Merits :

- Pointers bring complexity since the records are based on pointers and graphs.
- Changes in the database isn't easy that makes it hard to achieve structural independence.

Relational Model:

A data model in which both data and their relationships are represented by means of tables is called Relational Model.

The relation is the only data structure used in this model to represent both entities and their interrelationships. A relation is a two dimensional table with a unique name.

Each row of a table is called a tuple and each column of a table is called an attribute. The set of all possible values in an attribute is called the domain of the attribute.

Relational Data Model

- A relational model groups data into one or more tables. These tables are related to each other using common records.

The data is represented in the form of rows and columns i.e. tables:


	Column1	Column2	Column3
Row1			
Row2			
Row3			
Row4			
Row5			

Relational Data Model

- **Example** : Let us see an example of two relations
<**Employee**> and <**Department**> linked to each other, with
DepartmentID, which is **Foreign Key** of <**Employee**> table
and Primary key of
<**Department**> table.

<Employee>		
EmployeeID	EmpName	DepartmentID
E09	Emily	D001
E04	Tom	D002
E11	Emma	D003

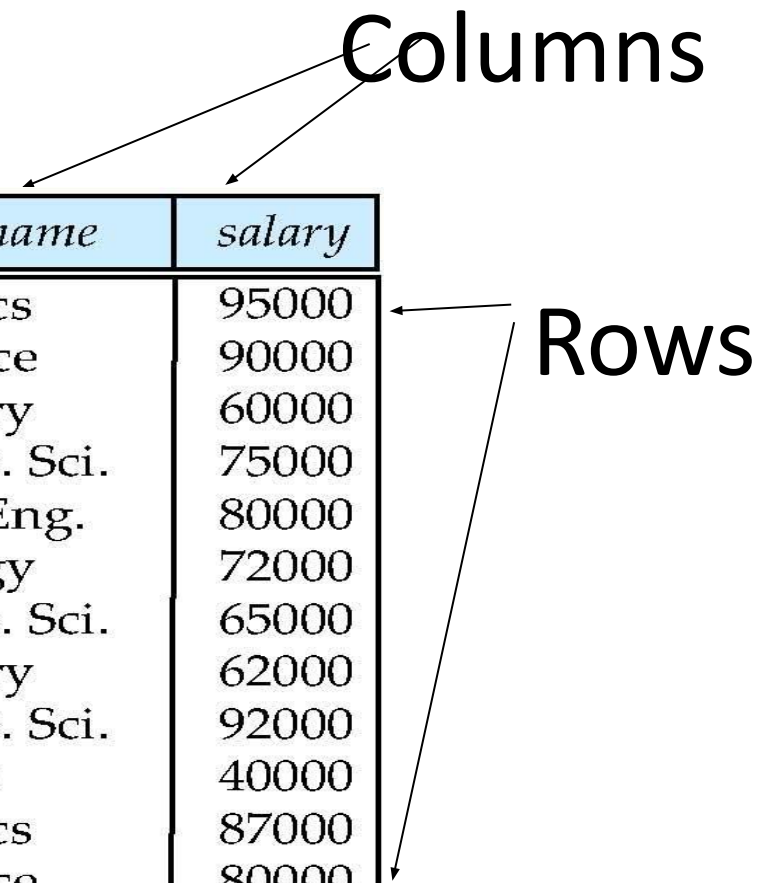
<Department>		
DepartmentID	DeptName	DeptZone
D001	Finance	North
D002	Operations	East
D003	Marketing	West



Relational Model

All the data is stored in various tables.

Example of tabular data in the relational model



<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

Instances and Schemas

Schema – the logical structure of the database (e.g., set of customers and accounts and the relationship between them)

Instance – the actual content of the database at a particular point in time

Instances and Schemas

Logical Schema – the overall logical structure of the database

Example: The database consists of information about a set of customers and accounts in a bank and the relationship between them

Physical schema – the overall physical structure of the database

Instance – the actual content of the database at a particular point in time

Schema
(in relational data model)



Student (studno, name, address)
Course (courseno, lecturer)

Instance



Student (123, Egger, Bozen)
Course (CS321, Nutt)

Data Independence

Ability to modify a schema definition in one level without affecting a schema definition in the other levels.

The interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Two levels of data independence

- Physical data independence

- Logical data independence

Data Independence

Data independence can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. We can define two types of data independence:

- 1) **Logical data independence** is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database, to change constraints, or to reduce the database. Only the view definition and the mappings need to be changed in a DBMS that supports logical data independence.
- 2) **Physical data independence** is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized for example, by creating additional access structures to improve the performance of retrieval or update.

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.

DBMS Architecture

Architecture is the frame work of the Database Management System.

Standard database consisting of Conceptual, external and internal levels.

Conceptual Level-Logical schema of database

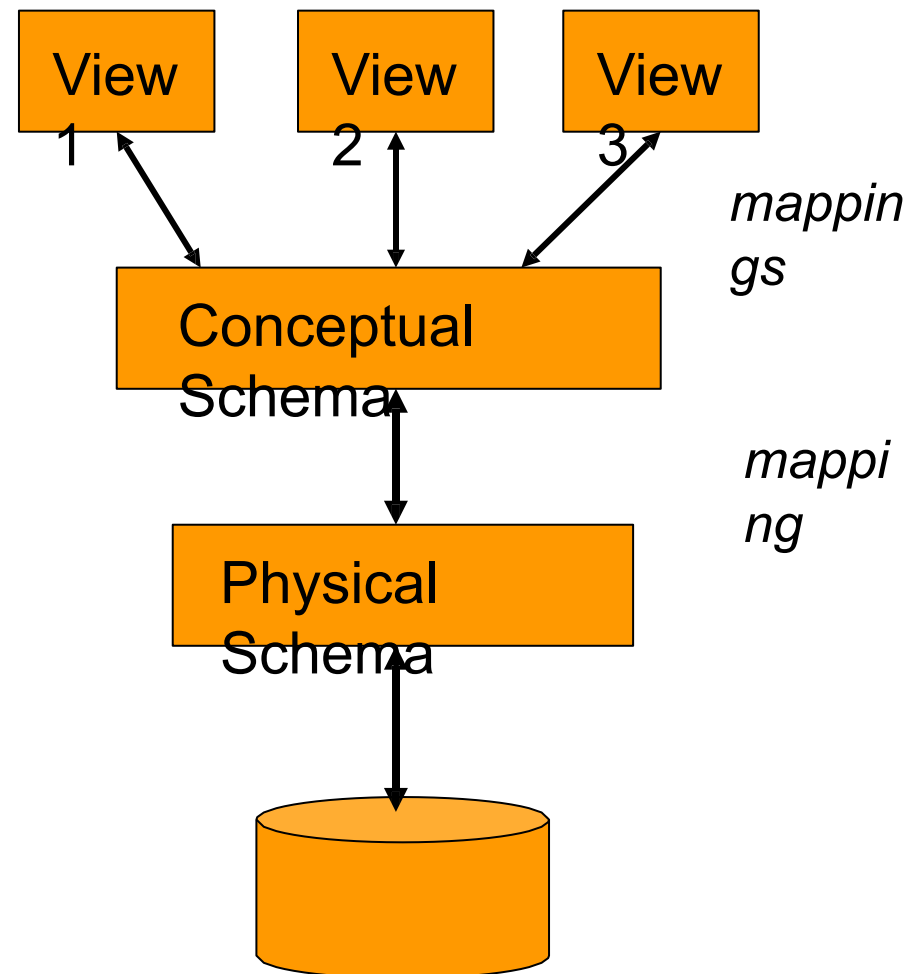
External Level-User views of database

Internal Level : Physical views of database

Three Levels of Abstraction

Architecture for DBMSs :

- *Many external views*
- *One conceptual*
(= logical) *schema*
- *One physical*
(= internal) *schema*
 - Views describe how users see the data
 - Conceptual schema defines logical structure
 - Physical schema describes the files and indexes used



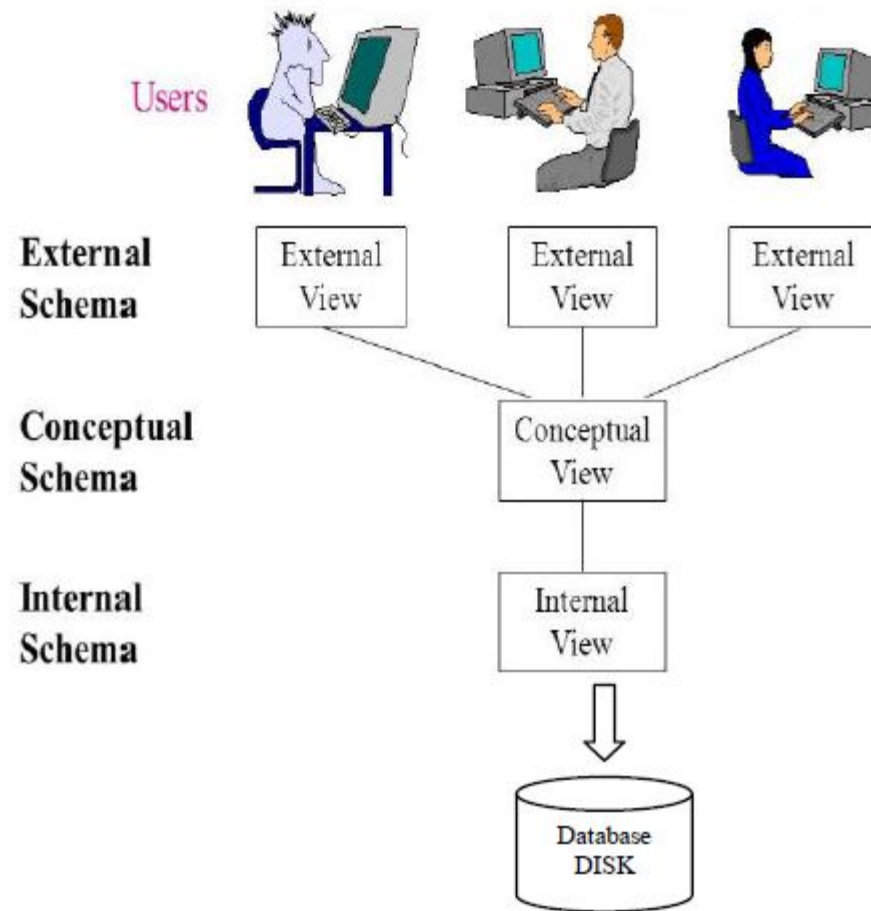
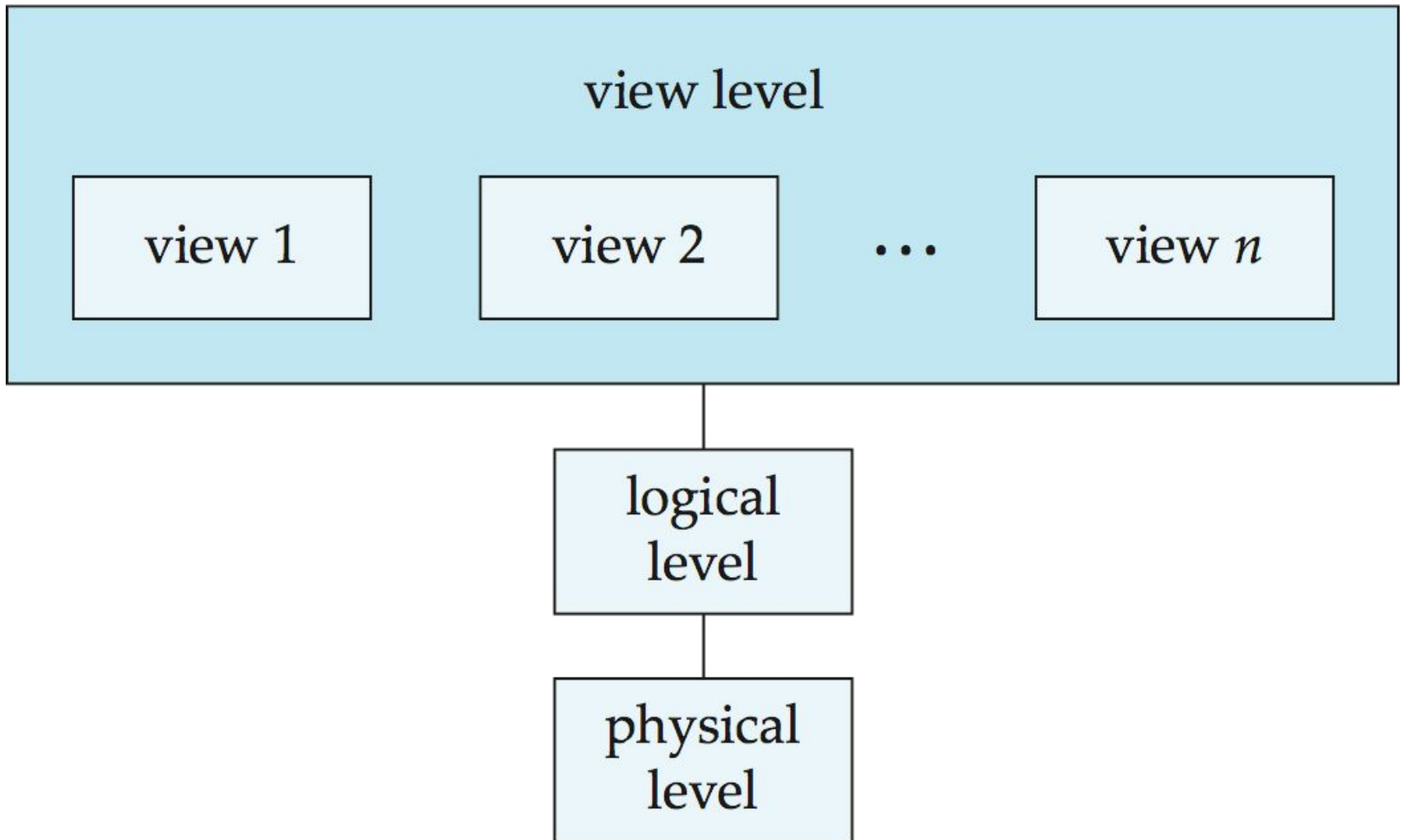


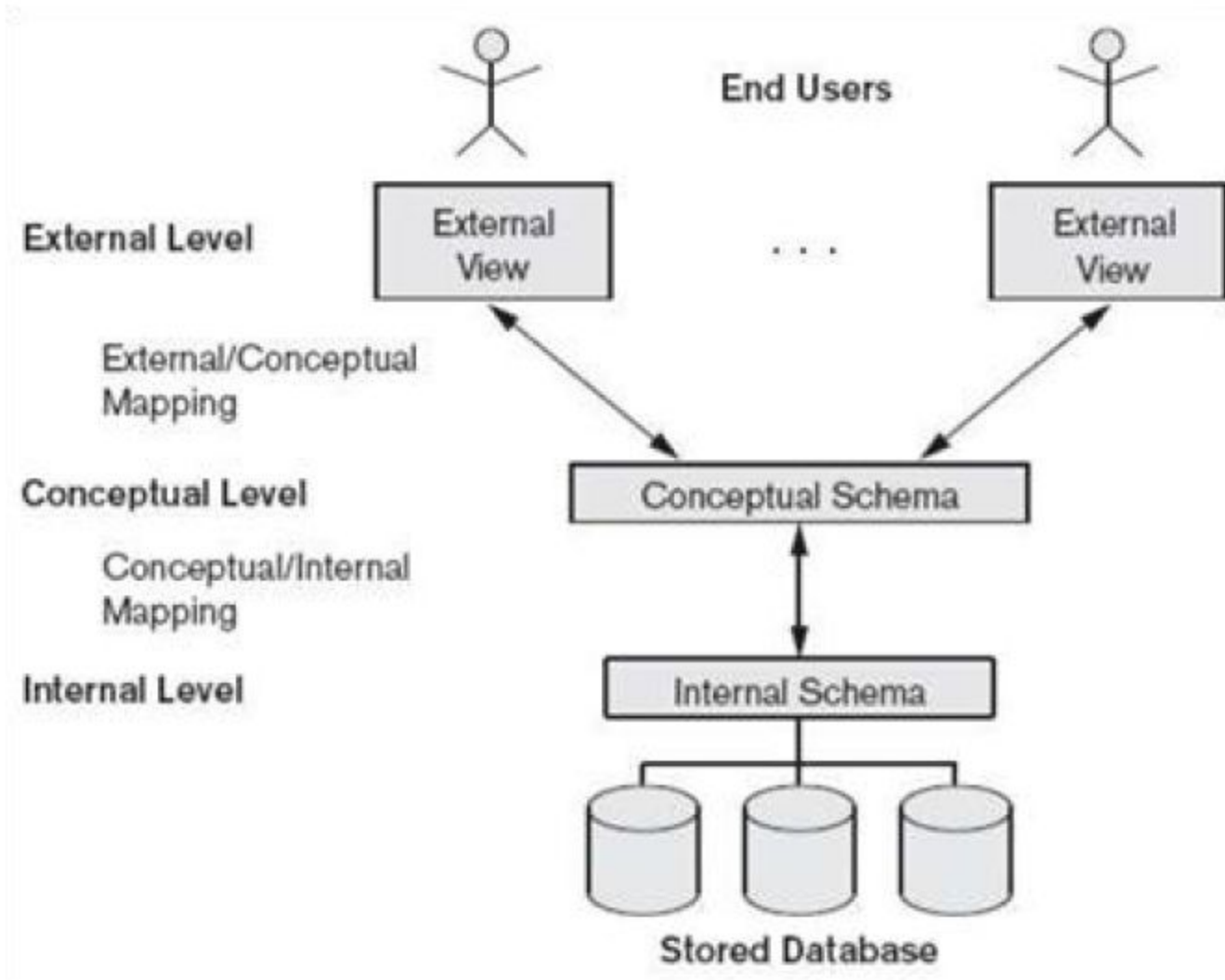
Figure 1.2 : Levels of Abstraction in a DBMS

View of Data

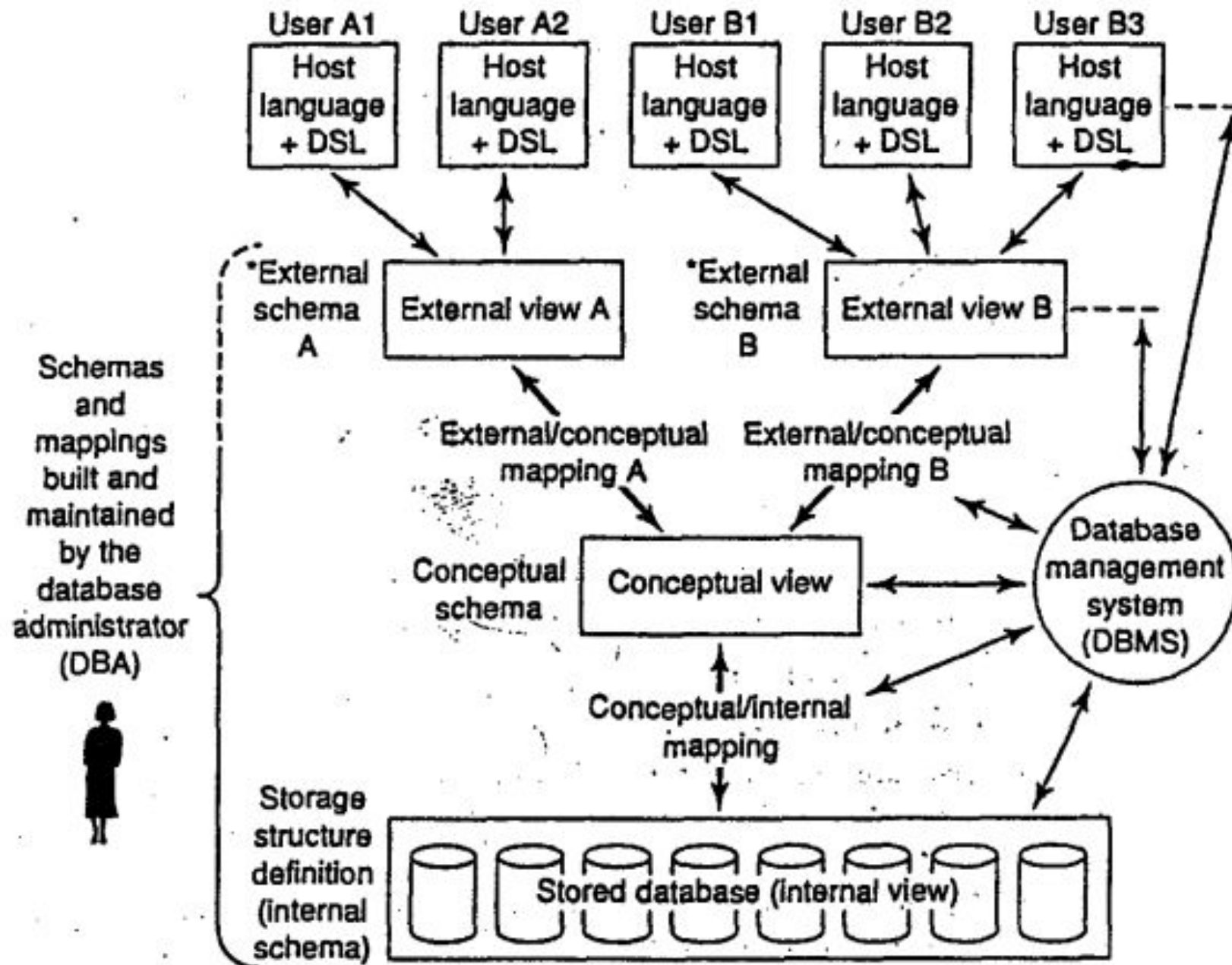
An architecture for a database system



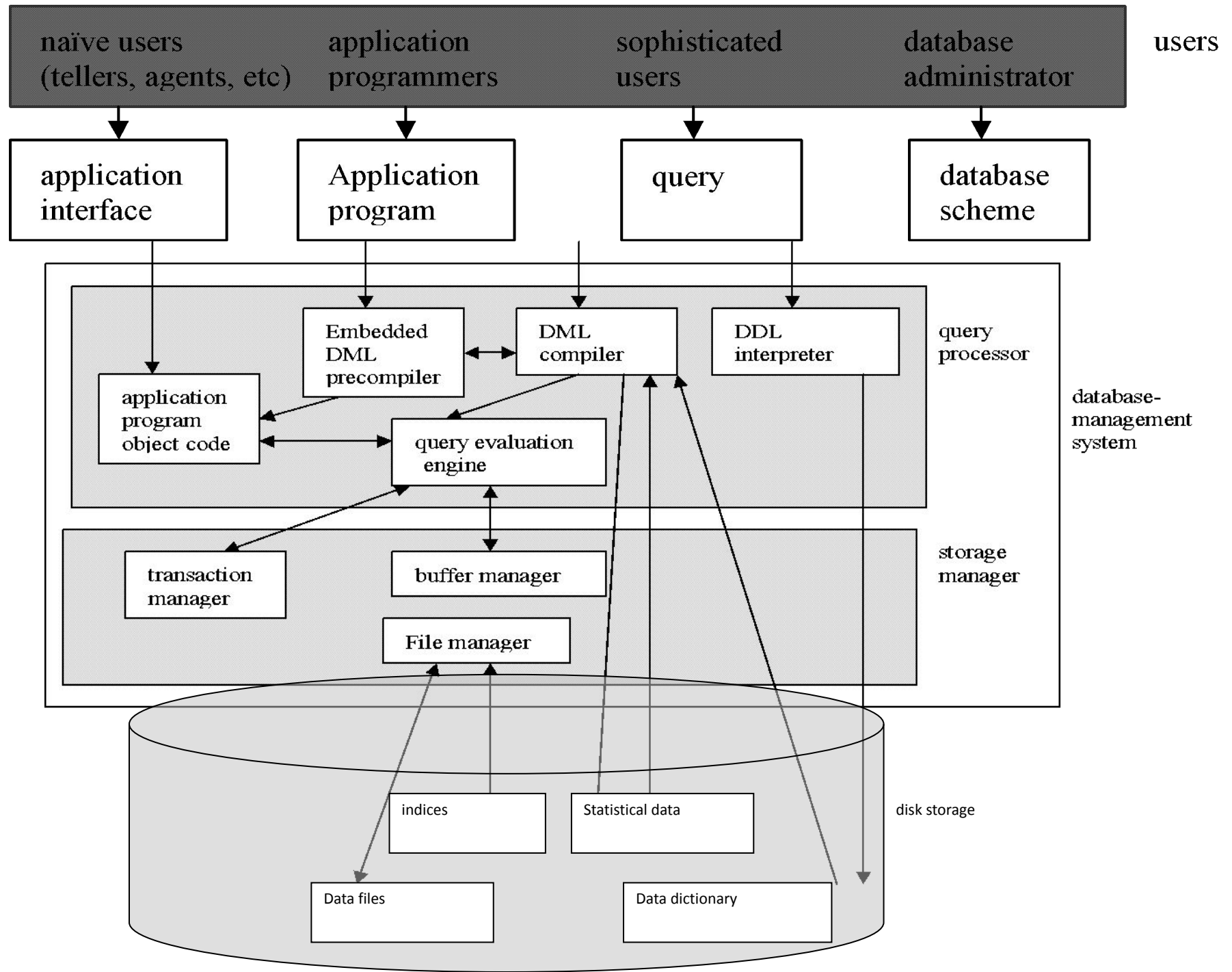
Three Schema Architecture



Architecture of DBMS



Overall System Structure

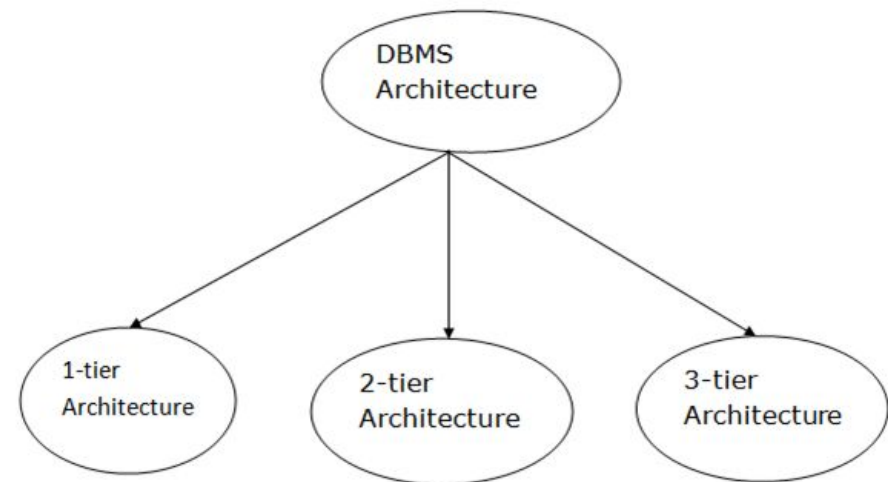


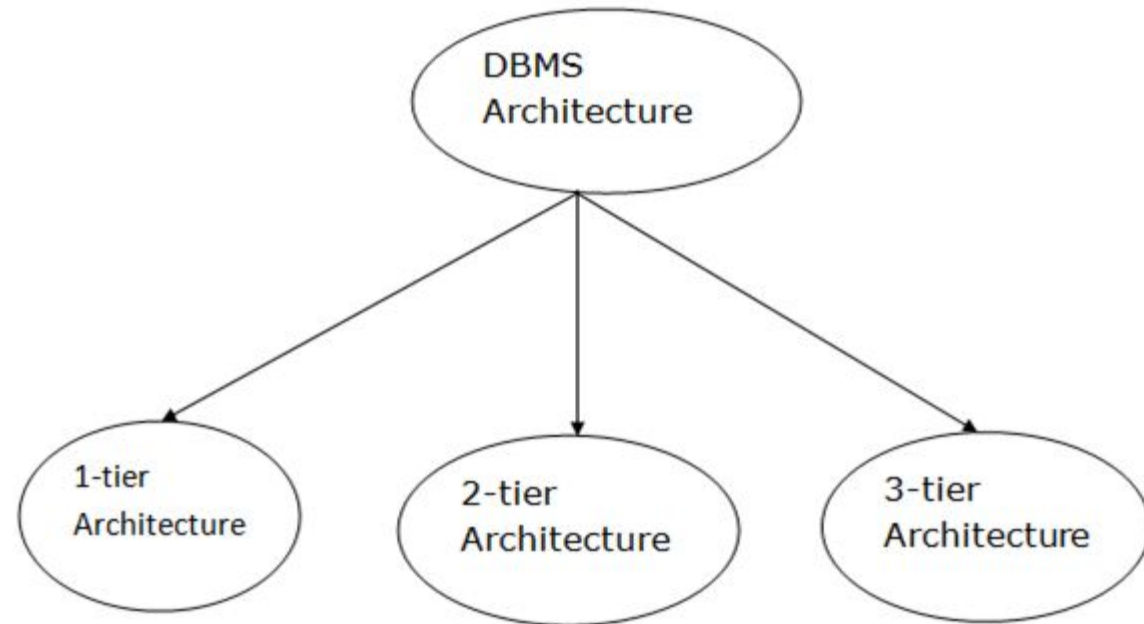
Types of Architectures

Single Tier (1-Tier) Architecture

Two Tier Architecture

Three Tier Architecture





1-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.

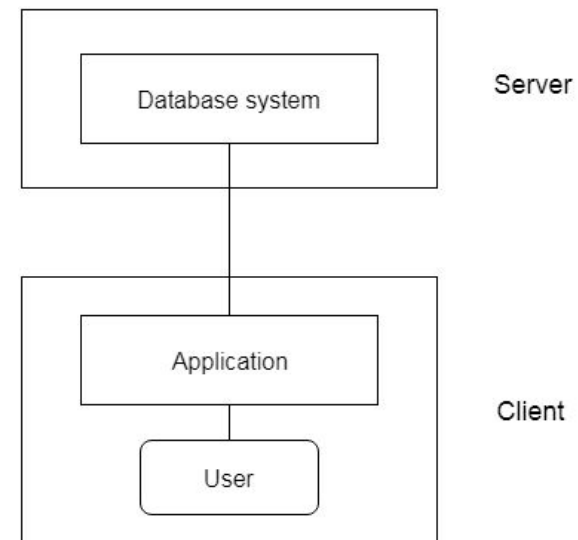
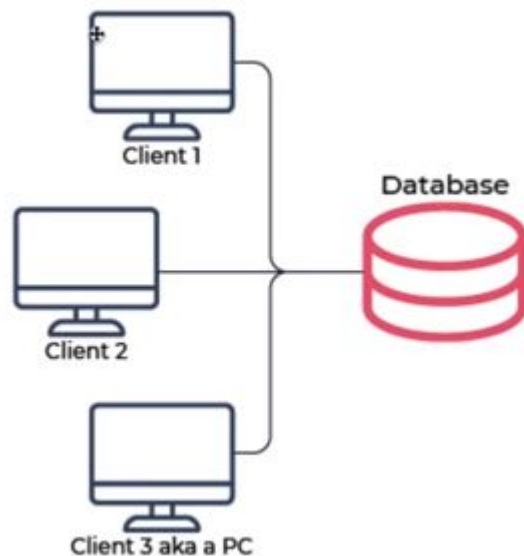


1-Tier Architecture

- In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the

2-Tier Architecture

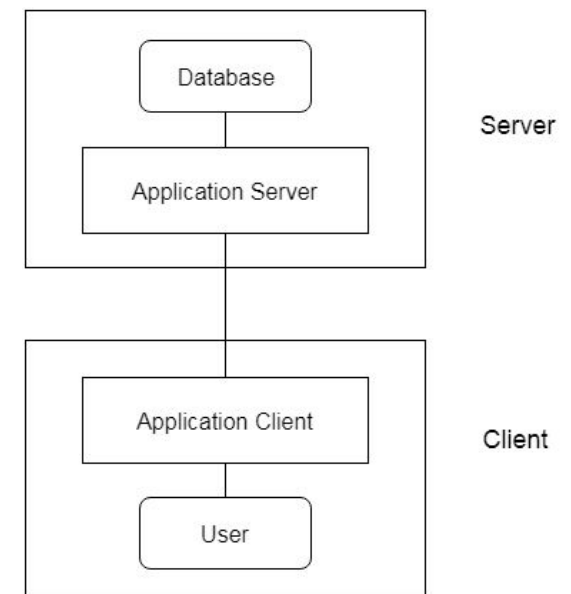
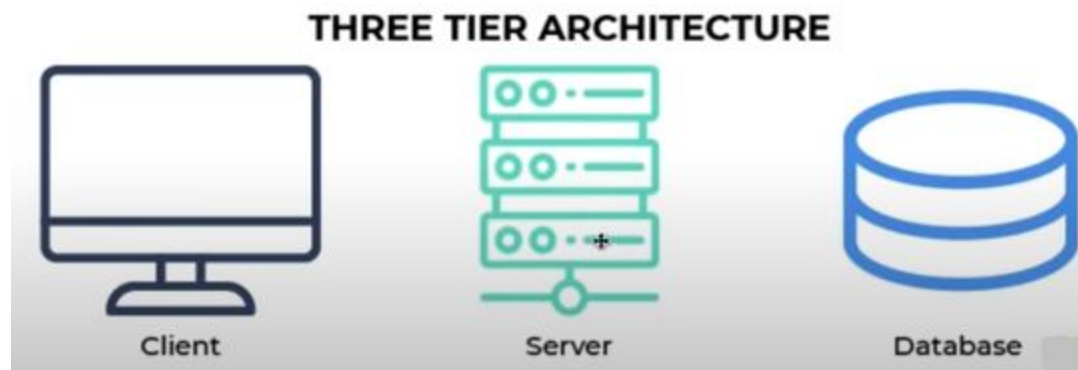
2-Tier Architecture : A 2 Tier Architecture in DBMS is a Database architecture where the presentation layer runs on a client (PC, Mobile, Tablet, etc.), and data is stored on a server called the second tier. Two tier architecture provides added security to the DBMS as it is not exposed to the end-user directly. It also provides direct and faster communication.



- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.
- The user interfaces and application programs are run on the client-side.
- The server side is responsible to provide the functionalities like: query processing and transaction management.
- To communicate with the DBMS, client-side application establishes a connection with the server side.

3-Tier 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.



- The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server.
- The application on the client-end interacts with an application server which further communicates with the database system.
- End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
- The 3-Tier architecture is used in case of large web application.

Database Architecture

Centralized databases

One to a few cores, shared memory

Client-server,

One server machine executes work on behalf of multiple client machines.

Parallel databases

Many core shared memory

Shared disk

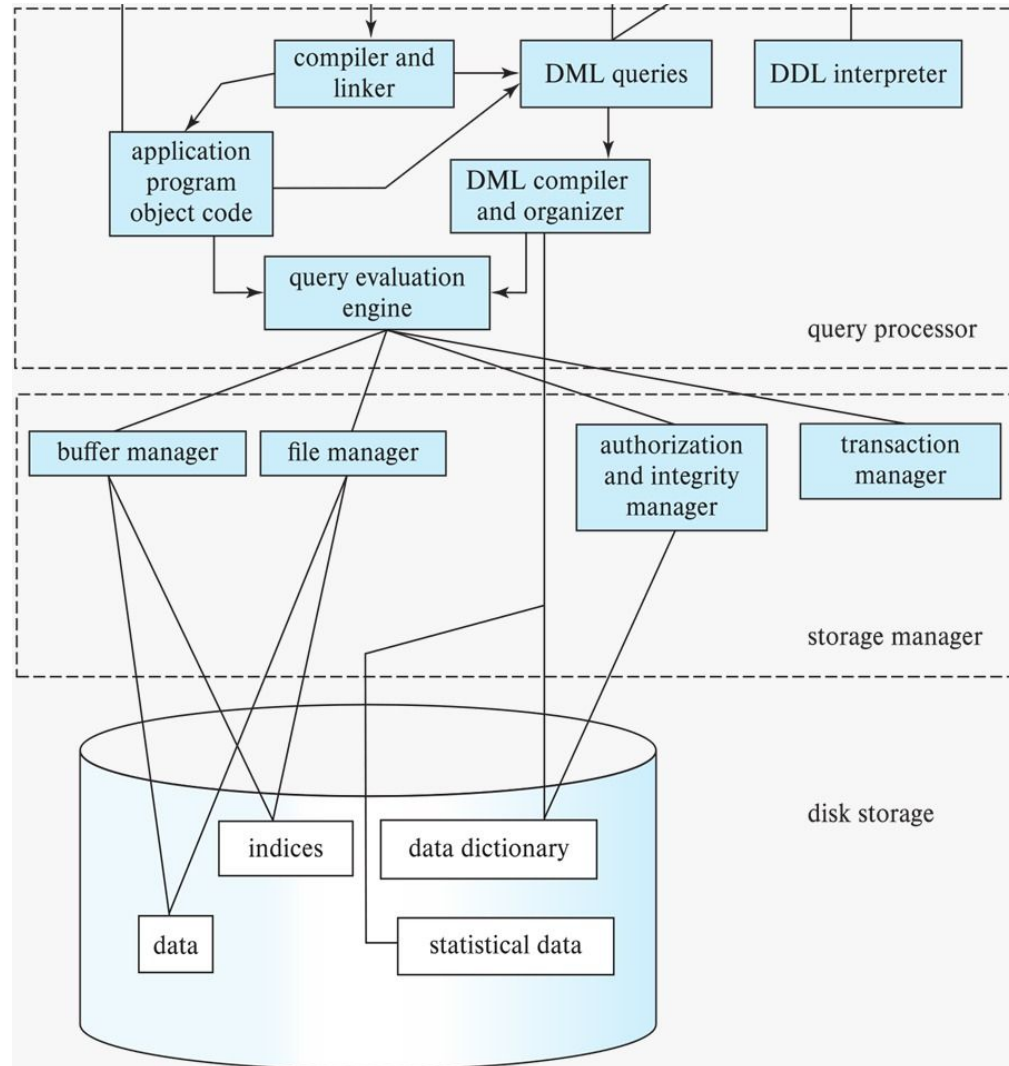
Shared nothing

Distributed databases

Geographical distribution

Schema/data heterogeneity

Database Architecture (Centralized/Shared-Memory)



Database Applications

Database applications are usually partitioned into two or three parts

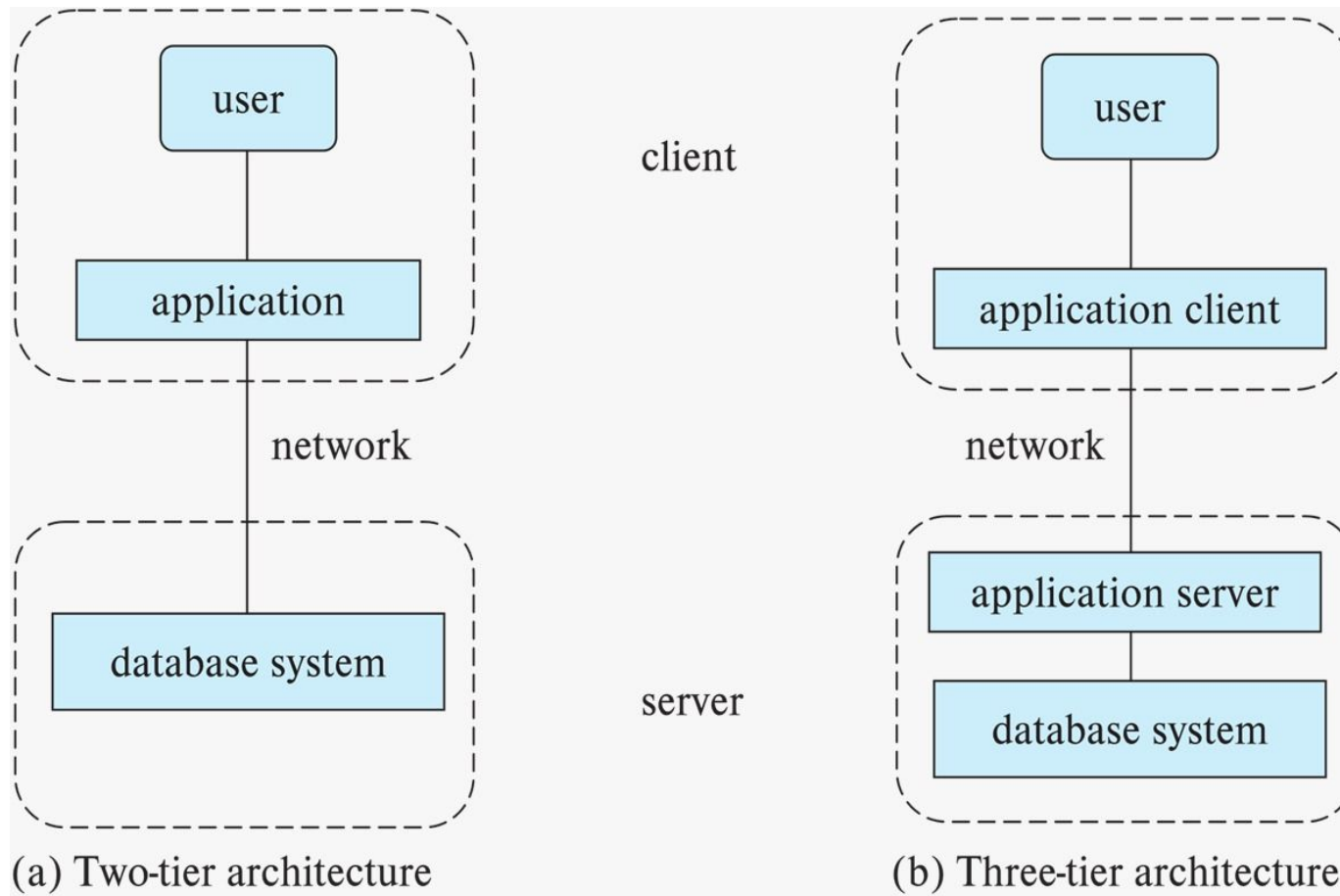
Two-tier architecture -- the application resides at the client machine, where it invokes database system functionality at the server machine

Three-tier architecture -- the client machine acts as a front end and does not contain any direct database calls.

The client end communicates with an application server, usually through a forms interface.

The application server in turn communicates with a database system to access data.

Two-tier and three-tier architectures



Possible Questions

1. Compare Database Management Systems with File Processing Systems
2. Discuss the applications of Database Management Systems
3. Explain the Data Base Architecture with a diagram / Explain Three Tier architecture of DBMS
4. Explain the roles of different database users
5. Explain brief introduction about Data Models
6. What is Data independence ? Explain about Physical Data independence and Logical Data independence
7. Define instance and Schema with examples
8. Discuss about DBMS Environment
9. Describe levels of abstraction in DBMS

Possible Questions

10. What are the Characteristics of Database Management Systems
11. What are the Features of Database Management Systems
12. Explain the duties of DBA
13. Explain Three-Tier architecture for data independence
14. Describe the structure of a Database Management System.

MCQs

Which is the advantage of database

- a) Prevents Data redundancy
- b) Restricts unauthorized access
- c) Backup and recovery
- d) All

Which is the database language

- a) C
- b) C++
- c) SQL
- d) None of these

Which level of database is viewed by user

- a) Internal level
- b) External Level
- c) Conceptual Level
- d) All of these

Which is the data model

- a) Relational
- b) Object-Relational
- c) Network
- d) All of these

Which is not the feature of database

- a) Data redundancy
- b) Independence
- c) Flexibility
- d) Data Integrity

Which is the type of data independence

- a) Physical data independence
- b) Logical data independence
- c) Both
- d) None

Which is the feature of database

- a) Query Language
- b) Multi user access
- c) Data Dictionary
- d) All of these

Which is the component of database management system

- a) Query Language
- b) Database Manager
- c) File manager
- d) All of these

The view of total database content is

- a) Conceptual view
- b) Internal view
- c) External view
- d) Physical view

Internal level has

- a) Individual Users View of the database
- b) Community view of the database
- c) Physical Representation of the database
- d) All of these