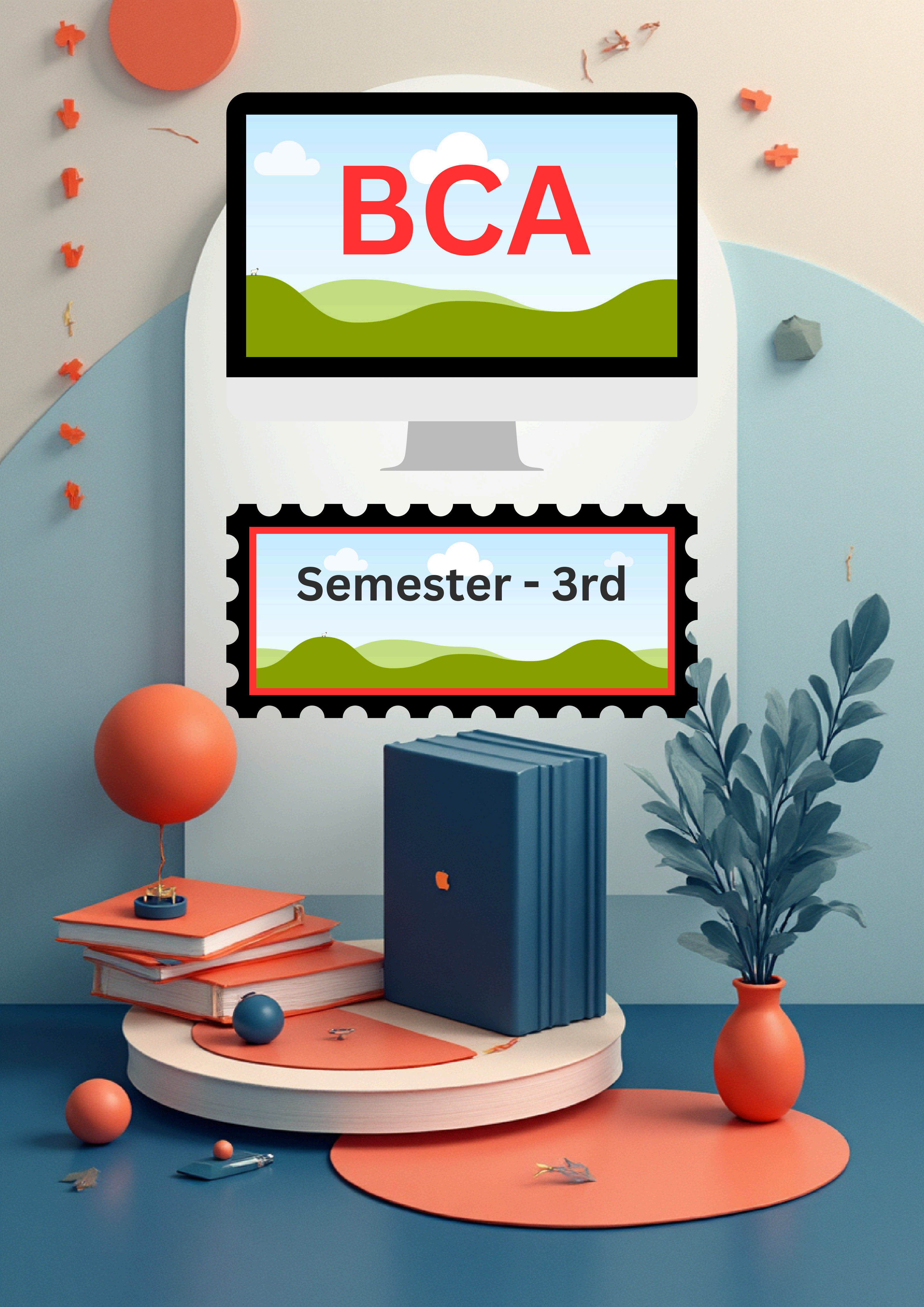
BCA
Semester - 3rd

# DATABASE MANAGEMENT SYSTEM

Notes - 1

## Contents

*Database and Database Users, Characteristics of the Database Approach. Structure, Function and Components of DBMS, Different people behind DBMS, Advantages of using DBMS. Database System Concepts and architecture: Data Models, Schemas, and Instances. DBMS 3-Level ANSI/SPARC Architecture and Data Independence. Types of DBMS.*

## Database and Database Users

**Database:**

A Database is a collection of related data that is organized in such a way that it can be easily accessed, managed, and updated.

In simple words, it is a systematic collection of information that represents real-world entities like students, employees, products, etc.

Example:

A college database may store information about students, teachers, courses, and results — all connected and managed together.

Purpose of Database:

To store data efficiently.

To reduce data duplication.

To maintain data consistency and accuracy.

To make data retrieval faster and easier.

**Database Users:**

There are different types of users who interact with the database depending on their role and technical knowledge.

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

They manage the overall database system.

Responsible for database design, security, backup, recovery, and performance.

Example: IT person who maintains a university database.

***2. Database Designers:***
They design the structure of the database — how data will be stored, related, and organized.

Example: Decides tables for “Student”, “Course”, “Fees”, etc.

***3. End Users:***
People who use the database system to perform their daily work.

Types of end users:
Casual Users: Access data occasionally using queries (like managers).
Naïve Users: Use predefined applications (like bank clerks).
Sophisticated Users: Use complex queries (like analysts).
Standalone Users: Maintain personal databases (like Excel users).

***4. Application Programmers:***
They write programs (using Java, Python, etc.) that interact with the database.

Example: Developer who builds a library management system using DBMS.

## Characteristics of the Database Approach

The database approach is different from the traditional file system because it provides a systematic and integrated way of storing and managing data.

Here are the main characteristics 👇

***1. Self-describing Nature of a Database System***

Database stores not only data but also metadata (data about data). Metadata includes details like table names, data types, and relationships.

Example: Information about columns, keys, and constraints is stored inside the database itself.

***2. Insulation Between Programs and Data (Data Independence)***

In DBMS, data and application programs are independent.
Changing the structure of data does not require changing the programs that access it.

Example: If you add a new column in a table, old programs still work fine.

***3. Data Abstraction***

Users see data at different levels of abstraction —
Physical Level: How data is stored internally.
Logical Level: What data is stored and relationships.
View Level: What part of data is shown to users.
This helps in hiding complex details from users.

***4. Multiple Views of Data***
Different users can have different views of the same data.

Example: A student can view only his marks, but the teacher can see marks of all students.

***5. Sharing of Data and Multi-user Transaction Processing***
DBMS allows multiple users to access and modify data at the same time.
It maintains data consistency using transaction control.

Example: Many bank employees can update customer accounts simultaneously.

***6. Control of Data Redundancy***
Same data is not stored repeatedly in different files.
Reduces storage waste and inconsistency.

Example: Student address stored only once instead of in multiple files.

***7. Data Integrity and Security***
DBMS ensures data is accurate, valid, and protected.
Integrity rules prevent invalid data (like negative age).
Security controls who can access or modify the data.

***8. Backup and Recovery***
DBMS provides automatic backup and recovery features in case of system failure.
Ensures data safety and business continuity.

***9. Enforced Standards***
DBMS follows standard data models (like relational model) and query languages (like SQL), ensuring uniformity and compatibility.

***10. Improved Data Access and Decision Making***
With the help of queries and reports, data can be analyzed easily.
Helps management in better decision-making.

**In Short:**
A Database is an organized collection of data, managed by a DBMS.
Different users (DBA, programmers, designers, and end users) interact with it.

The database approach provides features like data independence, sharing, reduced redundancy, and security — making data management more efficient and reliable.

## Structure of DBMS

The structure of a DBMS (Database Management System) means how different parts of the database system are organized and how they interact with each other.

A DBMS acts as a bridge between the user and the database, managing all data storage, access, and processing.

The DBMS structure generally consists of three levels (as per the ANSI-SPARC model):

***a. Internal Level (Physical Level)***
Describes how data is stored in the computer.
Deals with storage structure, file organization, and indexes.

Example: How student records are stored on the hard disk.

***b. Conceptual Level (Logical Level)***
Describes what data is stored in the database and the relationships among them.
Focuses on the overall logical view of the database.

Example: Student table linked to Course table through Course_ID.

***c. External Level (View Level)***
Describes how users see the data.
Different users can have different views of the same database.

Example: A teacher can see all student marks, but a student can only see their own.

## Functions of DBMS

A DBMS performs several important functions to ensure efficient data management, accuracy, and security.
Here are the key functions 👇

1. Data Storage, Retrieval, and Update
DBMS stores large amounts of data efficiently.
Allows users to retrieve and update data quickly using query languages (like SQL).

2. User Access Control and Security
Provides login IDs, passwords, and roles to restrict unauthorized access.
Ensures data security and privacy.

3. Data Integrity Management
Ensures that data entered into the database is correct and consistent.
Example: Prevents entering a student age as "-5".

4. Backup and Recovery
Automatically creates data backups and provides recovery options in case of failures.

5. Data Independence
Allows changing the data structure without affecting the programs that use the data.

6. Data Sharing and Multi-user Access
Multiple users can access the database simultaneously without conflicts.

7. Concurrency Control
Manages multiple transactions happening at the same time, ensuring data accuracy.

8. Transaction Management
Ensures all database operations follow the ACID properties (Atomicity, Consistency, Isolation, Durability).

9. Communication with the Database
Provides query languages (like SQL) and APIs to communicate between user and database.

10. Report Generation
Helps users create reports for decision-making using data analysis and queries.

## Components of DBMS

A DBMS is made up of several components (modules) that work together to manage the database system efficiently.

Let's understand each one clearly

1. Hardware
The physical devices used for storing and running the database.
Example: Servers, hard disks, processors, memory (RAM).

2. Software
The actual DBMS software (like MySQL, Oracle, PostgreSQL) that manages data.
Includes the operating system and application programs that access the database.

3. Data
The core part of the system — includes actual data, metadata, and data dictionary.
Stored in tables, records, and fields.

4. Procedures
The set of instructions and rules for using and managing the DBMS.
Example: Steps to take backup, restore data, or add a new user.

5. Database Access Language
Language used to interact with the database — mainly SQL (Structured Query Language).
Used to insert, delete, update, and retrieve data.

6. Database Manager (DBMS Engine)
The central part that handles all interactions between user requests and the physical database.
Manages queries, storage, transactions, and access control.

7. Query Processor
Translates user queries (in SQL) into low-level instructions that the DBMS can execute.
It ensures that queries are optimized for speed and performance.

8. Database Schema and Metadata
Schema: The logical structure or blueprint of the database (tables, fields, relationships).
Metadata: Information about the data (like data type, constraints, etc.).

9. Concurrency Control Manager
Manages simultaneous access to data by multiple users without causing errors or duplication.

10. Database Administrator (DBA)
A person or module responsible for maintaining the DBMS — handles security, performance, and recovery.

# Different People behind DBMS

In a Database Management System (DBMS), different people or roles are involved in the designing, using, and managing of the database.

Each plays a specific role in ensuring the database works efficiently and safely.

Here are the main types of people behind a DBMS 👇

**1. Database Administrators (DBA)**

The most important person responsible for the overall management of the database system.
They make sure the database runs smoothly, is secure, and performs well.

Main Duties:
Installing and configuring the DBMS software.
Taking regular backups and performing recovery when needed.
Managing user permissions and security.
Monitoring performance and tuning the database.
Ensuring data consistency and integrity.

Example:
In a college database, the DBA manages the system that stores all student and teacher records safely.

**2. Database Designers**

They decide the structure and organization of the database before it is created.

Their job is to design tables, fields, keys, and relationships between data.

Main Duties:
Identify what data should be stored.
Design database schema (tables, relationships, constraints).
Ensure data is stored efficiently and without redundancy.

Example:
In a library system, designers decide to create tables like “Books”, “Members”, and “IssuedBooks” with proper relationships.

**3. Application Programmers**
They write programs or applications that interact with the database using programming languages like Java, Python, or PHP.
These programs allow users to enter, update, or view data easily.

Main Duties:
Create forms, reports, and user interfaces.
Write queries and code to insert, update, and delete data.
Connect the frontend (app) to the backend (database).

Example:
A developer creates a web app for students to view their marks from the database.

**4. End Users**
These are the people who actually use the database in their daily work.
They don’t manage or design the database — they just use it to get information.

***Types of End Users:***
Naïve Users:
Use predefined forms or applications.
Example: Bank teller entering customer details.

Casual Users:
Occasionally use queries to access data.
Example: Manager checking employee reports.

Sophisticated Users:
Use complex queries or analysis tools.
Example: Data analyst running SQL reports.

Standalone Users:
Maintain personal databases on their computer.
Example: A teacher maintaining marks in MS Access or Excel.

**5. System Analysts**
They study the requirements of users and design a system that meets those needs.
They work as a link between users and developers.

Main Duties:
Analyze user needs.
Design overall system flow.
Help in database system design and testing.

**6. Database System Developers**
They are responsible for creating and maintaining the DBMS software itself.
They work on low-level parts like query processing, indexing, and transaction control.

Example:
Developers at Oracle or Microsoft who design the actual DBMS software (like Oracle DB, SQL Server).

## Advantages of Using DBMS

A Database Management System (DBMS) offers many advantages compared to the traditional file-based system.

Here are the most important benefits

**1. Reduced Data Redundancy**

In file systems, the same data may be stored in multiple files, wasting space.
DBMS stores data only once and allows it to be shared by multiple users.

Example: Student address stored once and used by both admission and exam departments.

**2. Data Consistency**

Since data is stored only once, changes made in one place are reflected everywhere.
Ensures that all users see the same updated information.

**3. Data Sharing**

DBMS allows multiple users to access and share the same data at the same time.

Example: Many employees can check customer data simultaneously.

**4. Improved Data Security**

Only authorized users can access or modify the data.
DBAs can set passwords, roles, and permissions for different users.

**5. Data Integrity**
DBMS ensures that only valid and accurate data is entered into the system using rules and constraints.
Example: Prevents entering negative salary or invalid roll numbers.

**6. Data Independence**
Application programs and data are separate.
You can change the database structure without affecting the applications using it.

**7. Backup and Recovery**
DBMS provides automatic backup and recovery features to protect data from system crashes or failures.

**8. Concurrency Control**
DBMS manages multiple transactions at the same time without data conflicts.
Example: Two users withdrawing money from the same bank account safely.

**9. Better Decision Making**
Data can be easily analyzed using queries and reports.
Helps management make informed and quick decisions.

**10. Easy Maintenance and Scalability**
DBMS makes it easy to modify, add, or delete data structures.
It can handle large databases and support many users as the system grows.

## Database System Concepts and Architecture

The database system architecture describes how a database is structured, organized, and accessed by users.

It explains how different components of a DBMS interact to store, process, and manage data efficiently.

To understand the architecture, we use the three-level architecture model proposed by ANSI/SPARC (American National Standards Institute / Standards Planning and Requirements Committee).

👉 Three Levels of Database Architecture:

**1. Internal Level (Physical Level)**

Deals with how data is physically stored in memory (like on disks or drives).
Includes file structures, indexes, and access paths.
It hides storage details from higher levels.
Example: Data stored in binary format or records saved in data blocks.

**2. Conceptual Level (Logical Level)**

Describes what data is stored in the database and the relationships among data.
It represents the entire database logically for the organization.
Example: Student table linked with Course table through Course_ID.

**3. External Level (View Level)**

Describes how users see the data.
Different users can have different customized views of the same database.
Example:
A student can view his marks.
A teacher can view all students’ marks.

## Data Models

A Data Model is a way to describe the structure of a database — it defines how data is stored, connected, and manipulated.

It gives a framework for organizing data logically.

👉 Simply: Data model = Structure + Operations + Constraints

**Types of Data Models**

***1. High-Level (Conceptual) Data Model***

Provides a user-friendly view of data.
Focuses on entities (things), their attributes, and relationships.

Example: Entity-Relationship (ER) Model
Entities: Student, Teacher, Course
Relationships: Enrolls, Teaches

***2. Record-Based (Logical) Data Model***

Represents data in records (rows) and fields (columns).
Commonly used in relational databases.

Types:
Relational Model: Data in tables (rows & columns).
Network Model: Data connected with links forming a graph.
Hierarchical Model: Data arranged like a tree (parent-child).

***3. Physical Data Model***

Describes how data is stored internally on computer hardware.
Focuses on file organization, indexes, and storage methods.

Example: How “Student” records are stored on the disk with access paths.

## Schemas

A Schema is the blueprint or structure of a database — it defines how the data is organized and how the relationships are set up.

👉 Think of schema as a database design or plan.
It defines:
Tables
Fields (columns)
Data types
Relationships between tables
Constraints (like primary key, foreign key)

Example:
If your database has tables like Students, Courses, Fees, their names, columns, and relationships together form the database schema.

***Types of Schemas (based on architecture):***
1. Physical Schema
Describes how data is physically stored.
Includes details like file paths, storage format, and indexes.
Example: Student table stored in file "student_data.db".

2. Logical Schema
Describes the logical structure of the database — tables, relationships, constraints.
Example: "Student" table linked to "Course" table by Course_ID.

3. View Schema (External Schema)
Describes how individual users see the data.
Example: A student's view showing only his own marks and attendance.

## Instances

A Database Instance is the data actually stored in the database at a specific moment of time.

👉 In simple words:
Schema = Structure (Design)
Instance = Data (Current state)

📘 Example:
Schema: Student(Name, RollNo, Course)

| Name | RollNo | Course |
|---|---|---|
| Riya | 101 | BCA |
| Aman | 102 | MCA |

Here, the table structure (columns) is the Schema,
and the rows of actual data are the Instance.

**Final Summary:**
The Database Architecture defines how data is stored and viewed at different levels.

The Data Model defines how data is structured and connected.

The Schema defines the overall design of the database, while the Instance represents the actual data at a specific moment.

# DBMS 3-Level ANSI/SPARC Architecture

The ANSI/SPARC (American National Standards Institute / Standards Planning and Requirements Committee) proposed a three-level architecture for DBMS to separate the user's view of data from how it is physically stored.

This architecture helps in data abstraction and data independence, meaning users don't need to know how data is actually stored inside the computer.

### 👉 The Three Levels of DBMS Architecture:

**1. External Level (View Level)**
This is the topmost level.
It shows how individual users see the data.
Different users can have different views of the same database depending on their needs.

Features:
Each view shows only relevant data to the user.
Provides security by hiding unnecessary data.
Users interact with data through applications, forms, or reports.

📘 Example:
In a college database —
A student can see only his marks.
A teacher can see marks of all students.
A clerk can see fee details only.

**2. Conceptual Level (Logical Level)**
This is the middle level.
It describes what data is stored in the database and how the data is related.
It provides a logical view of the entire database for the organization.

Features:
Hides physical storage details.
Defines entities, attributes, relationships, and constraints.
Represents the overall structure of the database.

📘 Example:
The conceptual design of a university database contains tables like:
Student, Course, Faculty, Department, and their relationships.

**3. Internal Level (Physical Level)**
This is the lowest level.
It describes how data is actually stored inside the storage devices (like hard disks).
Includes details of file organization, indexing, data blocks, and access paths.

Features:
Focuses on storage efficiency and performance.
Deals with compression, encryption, and data format.

📘 Example:
How "Student" records are stored in files, indexed by "Roll No", and organized in data blocks.

## Data Independence

Data Independence means that changes in one level of the database do not affect the other levels.

It is one of the most important advantages of using DBMS.

Types of Data Independence

**1. Logical Data Independence**

The ability to change the logical (conceptual) schema without affecting the external schema (user view) or application programs.
Example: Adding a new column (like “Email”) in the Student table should not affect user applications that only use “Name” and “RollNo”.

Benefits:
Easy to modify database structure.
Reduces cost and effort for system maintenance.

**2. Physical Data Independence**

The ability to change the physical storage details without affecting the conceptual schema.
Example: Moving data from SSD to cloud or changing indexing method — the logical structure remains the same.

Benefits:
DBAs can improve performance without disturbing the database design.
Makes the system flexible and efficient.

## Types of DBMS

A Database Management System (DBMS) can be classified based on how data is organized, stored, and related to each other.

There are mainly four types of DBMS 👇

**1. Hierarchical DBMS**

Data is stored in a tree-like structure (parent-child relationship).
Each child record has only one parent, but a parent can have many children.
Data access is done through navigating the tree path.

Example:
Company → Departments → Employees

Advantages:
Simple and fast for one-to-many relationships.
Data integrity is high because of parent-child linkage.

Disadvantages:
Difficult to reorganize or modify structure.
Not suitable for many-to-many relationships.

📘 Example System: IBM's Information Management System (IMS).

**2. Network DBMS**

Data is represented as records connected by links (pointers) forming a graph structure.
A record can have multiple parent and child records (many-to-many relationship).

Example:
Student → Enrolls in → Course
Course → Has → Faculty

Advantages:
Flexible relationships.
Efficient for complex data connections.

Disadvantages:
Complicated to design and maintain.
Requires understanding of pointers and navigation.

Example System: Integrated Data Store (IDS).

**3. Relational DBMS (RDBMS)**
Data is stored in the form of tables (relations) consisting of rows (tuples) and columns (attributes).
Uses keys and relationships to connect tables.
Provides Structured Query Language (SQL) for accessing data.

Example:
Student Table
RollNoNameCourse
101
Riya
BCA

Advantages:
Easy to use and understand.
Data independence and integrity.
Supports SQL for data operations.

Disadvantages:
Slower for very large, complex databases.

Example Systems: MySQL, Oracle, PostgreSQL, SQL Server.

**4. Object-Oriented DBMS (OODBMS)**
Data is stored in the form of objects, similar to object-oriented programming (OOP).
Each object contains data (attributes) and methods (functions).

Advantages:
Supports complex data types (images, videos, etc.).
Works well with OOP languages like Java, C++, Python.

Disadvantages:
More complex to design.
Not as widely used as RDBMS.

Example Systems: ObjectDB, db4o, Versant.

## Practice Questions

1. What is a Database? Explain different types of Database Users.
2. What are the main Characteristics of the Database Approach?
3. Explain the Structure, Function, and Components of a DBMS.
4. Who are the different people behind a DBMS and what are their roles?
5. What are the main Advantages of using a DBMS over a File System?
6. Explain the concept of Data Models, Schemas, and Instances in a Database System.
7. Describe the 3-Level ANSI/SPARC DBMS Architecture with a neat diagram.
8. What is Data Independence? Explain its types with examples.
9. Write and explain the different Types of DBMS.
10. Explain the difference between Database and DBMS with suitable examples.

Check the answer in the Practice Questions section on our website.


prepfolio.co.in
Visit Website
Thank You