DBMS Lec 2:

A Database Management System (DBMS) helps organize, store, and manage large amounts of data efficiently. It uses abstraction to hide complex details from users and provides different views of data for different users, ensuring security, consistency, and easy access through structured architectures and specific languages.

Introduction to DBMS and Abstraction

DBMS is a system for managing data. The concept of abstraction is key; it hides complex internal workings and only shows users what they need, making systems easier to use and understand.

Real-World Analogy for Abstraction

Abstraction is compared to driving a car: users operate controls without needing to know the detailed mechanics. Similarly, DBMS users interact with data without seeing the underlying complexity.

Abstraction in Business Software

In business applications like accounting software, users perform tasks without knowing how data is stored,

encrypted, or processed. Only relevant information is shown, thanks to abstraction.

DBMS and Different User Views

DBMS allows multiple users or departments (like in Amazon) to access only the data they need. Each department—such as logistics or customer service—gets a tailored view, keeping sensitive or unnecessary information hidden.

Multi-User Access and Data Independence

Multiple users can access the same database, but each gets a personalized view. Data is stored once and mapped to different views, ensuring efficiency and data consistency.

Three Levels of DBMS Architecture

DBMS uses three levels of abstraction: -

- Physical (how data is stored),
- Logical (how data is organized and related),
- View/External (how data appears to each user). These layers separate storage details from user interaction, enabling flexibility and security.

Logical Level and Data Organization

The logical level describes how data is structured, such as tables for students and courses, and how relationships between data are defined. This is independent of how data is physically stored.

Data Independence

Data independence means changes in how data is stored (physical level) do not affect how data is used or viewed (logical and view levels). This makes updates and maintenance easier and safer.

Logical Schema: Attributes, Constraints, and Relationships

Each table (like students) has attributes (name, ID, phone), constraints (rules like unique IDs), and relationships (such as students enrolling in courses). These ensure data accuracy and meaningful connections.

Data Models and Schema

Data models define how data is logically organized and related. The schema is the blueprint for the database, guiding how data is structured and accessed.

Database Languages: DDL and DML

Two main types of database languages are: - DDL (Data Definition Language) for defining and changing the database structure,

DML (Data Manipulation Language) for inserting, updating, deleting, and retrieving data. SQL is a common language that combines both.

Interfaces and Connectivity (JDBC/ODBC)

Programming languages (like Java or Python) interact with databases using interfaces such as JDBC or ODBC. These interfaces translate program requests into database commands, enabling apps to use data efficiently.

Role of the Database Administrator (DBA)

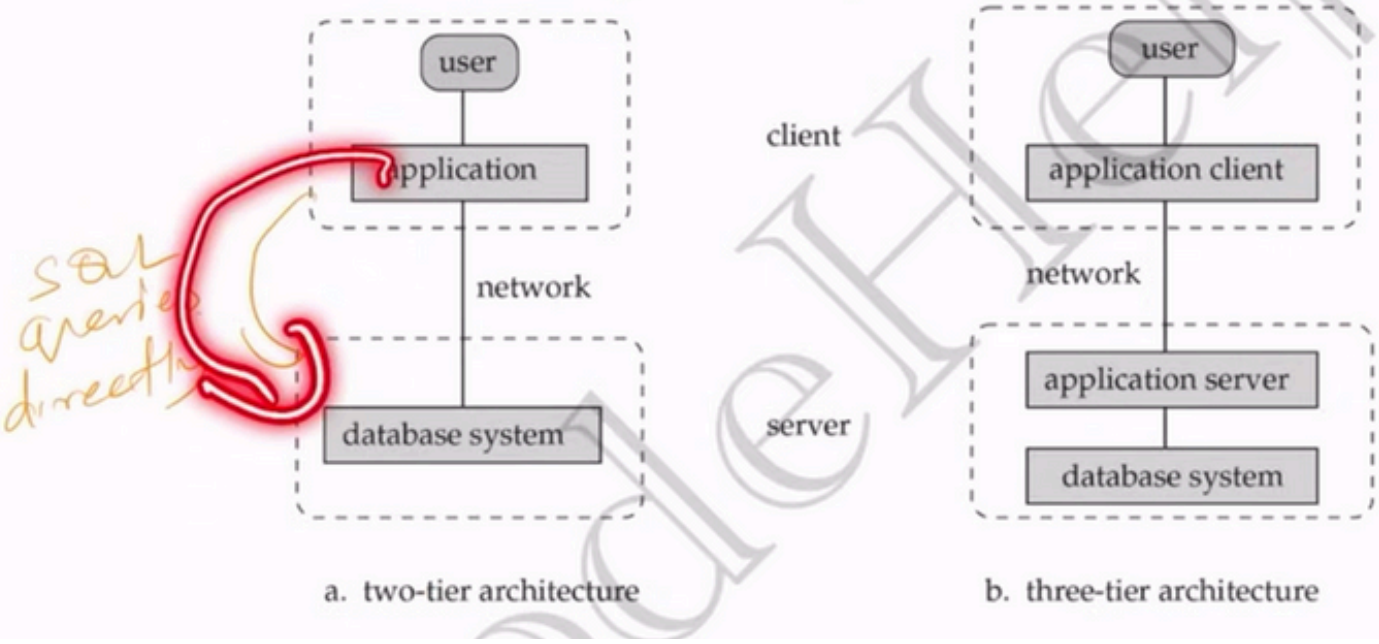
The DBA manages the database’s structure, security, access, and maintenance. They ensure data

consistency, control changes, manage user permissions, and handle backups.

DBMS Application Architectures: Single, Two, and Three Tier

- Single-tier: All parts (client, server, database) on one machine.
- Two-tier: Client and server are separate; the client communicates directly with the database server.
- Three-tier: Client, application server, and database server are all separate. This is common in large applications like Amazon or Facebook, improving scalability and security.

c. **T3** Architecture
   i. App is partitioned into 3 logical components.
   ii. Client machine is just a frontend and doesn't contain any direct DB calls.
   iii. Client machine communicates with App server, and App server communicated with DB system to access data.
   iv. **Business** logic, what action to take at that condition is in App server itself.
   v. T3 architecture are best for **WWW** Applications.
   vi. **Advantages**:
      1. **Scalability** due to distributed application servers.
      2. **Data integrity**, App server acts as a middle layer between client and DB, which minimize the chances of data corruption.
      3. **Security**, client can't directly access DB, hence it is more secure.



a. two-tier architecture b. three-tier architecture

Security, Integrity, and Maintenance in DBMS

DBMS ensures data security, integrity, and regular maintenance. Security controls limit access, integrity checks maintain correct data, and backups prevent data loss.