Introduction to Databases

Content

- Introduction to DBMS
- Database Management System
- Database-System Applications
- Purpose of Database System
- Views of Data
- Three Levels of Data Abstraction
- Components of Database System
- Database System Architecture

Introduction to DBMS

EBOX
AN AMPHISOFT PRODUCT

What is database?

- A database is a collection of information that is organized so that it can be easily accessed, managed and updated.
- Data is organized into rows, columns and tables, and it is indexed to make it easier to find relevant information.
- Data gets updated, expanded and deleted as new information is added.

DBMS: A database management system (DBMS) is system software for creating and managing databases.

Database Management System

- A database is a collection of data elements (facts) stored in a computer in a systematic way. The computer program used to manage and query a database is known as a database management system (DBMS). So a database is a collection of related data that we can use for
- Defining - specifying *types* of data
- Constructing - storing & populating
- Manipulating - querying, updating, reporting

Database Management System

- A Database Management System (DBMS) is a software package to facilitate the creation and maintenance of a computerized database. A Database System (DBS) is a DBMS together with the data itself.

Features of a database:

- It is a persistent (stored) collection of related data.
- The data is input (stored) only once.
- The data is organized (in some fashion).
- The data is accessible and can be queried

Database-System Applications

- Here are some representative applications.

➢ **Enterprise Information**

  - Sales
  - Accounting
  - Human Resources
  - Manufacturing
  - Online Retailers

Database-System Applications

- **Banking & Finance**
  - Banking
  - Credit Card Transactions
  - Finance
- **Universities**
- **Airlines**
- **Telecommunications**

Purpose of DBMS:

Database management systems were developed to handle the following difficulties of typical file-processing systems supported by conventional operating systems:

1. Data redundancy and inconsistency

- Storing the same data multiple times is called data redundancy.
- Data inconsistency is the various copies of the same data may no larger Agree

Purpose of DBMS:

2. Difficulty in accessing data

- File processing environments do not allow needed data to be retrieved in a convenient and efficient manner.

3. Data isolation –multiple files and formats

- Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult

Purpose of DBMS:

4. Integrity problems

- Data integrity means that the data values in the data base should be accurate in the sense that the value must satisfy some rules.

5. Atomicity of updates

- Atomic means the transaction must happen in its entirety or not at all. It is difficult to ensure atomicity in a conventional file processing system.

Purpose of DBMS:

6. Concurrent access by multiple users

- Data may be accessed by many applications that have not been coordinated previously so it is not easy to provide a strategy to support multiple users to update data simultaneously.

7. Security problems

- Data security means prevention of data accession by unauthorized users. Enforcing security constraints to the file processing system is difficult.

Views of Data

- A major purpose of a database system is to provide users with an abstract view of the data i.e the system hides certain details of how the data are stored and maintained.
- Views provide a level of security and a mechanism to customize the appearance of the database.
- A view can present a consistent, unchanging picture of the structure of the database, even if the underlying database is changed.

Data Abstraction

- Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called data abstraction.

Three levels of data abstraction:

- **Physical Level**
- **Logical Level**
- **View Level**

Three Levels of Data Abstraction

Physical Level

- The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low-level data structures in detail.

Logical level

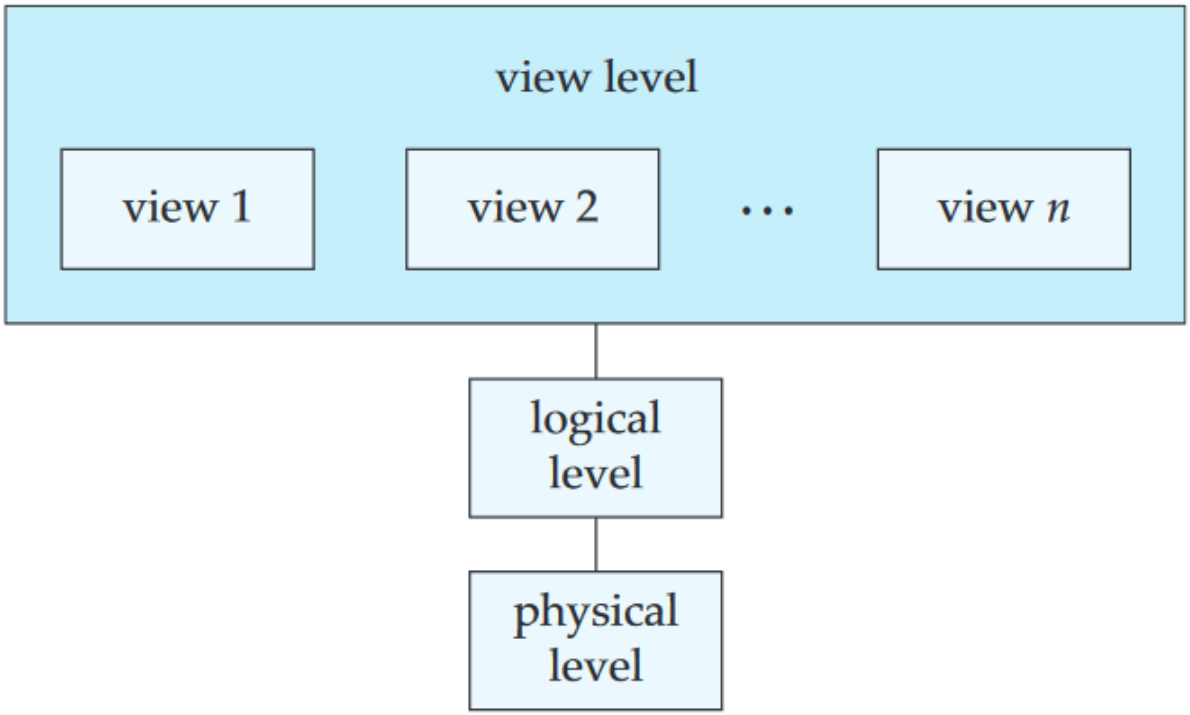
- The next-higher level of abstraction describes what data are stored in the database, and what relationships exist among those data.
- This level is referred to as physical data independence.

Three Levels of Data Abstraction

View Level

- The highest level of abstraction describes only part of the entire database.
- The view level of abstraction exists to simplify their interaction with the system.

Three Levels of Data Abstraction

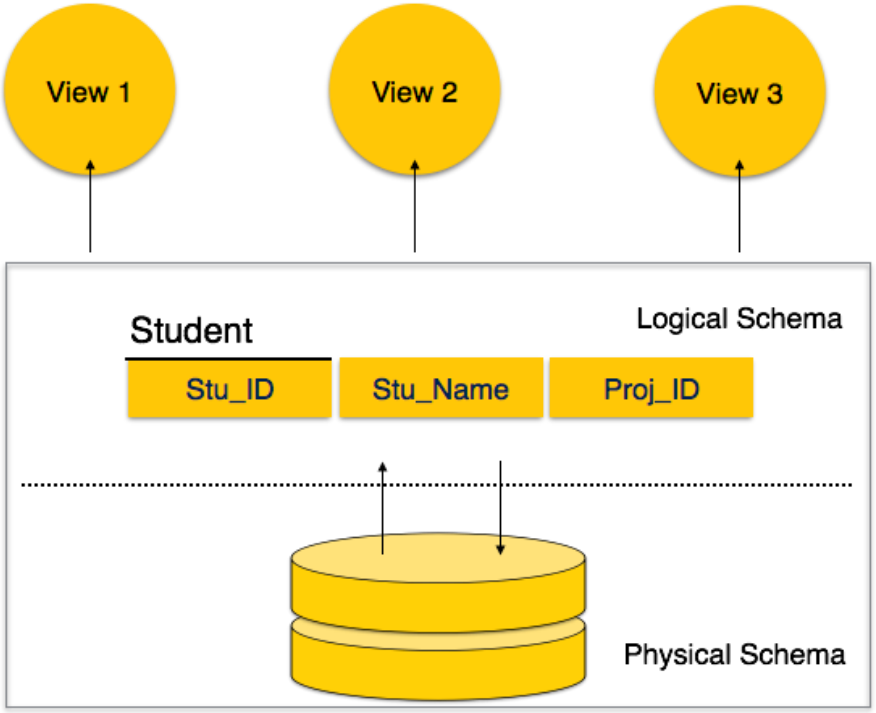


Instance and Schema

- **Definition of schema**: Design of a database is called the schema. Schema is of three types: Physical schema, logical schema and view schema.
- The design of a database at physical level is called **physical schema**, how the data stored in blocks of storage is described at this level.
- **Logical Database Schema** – This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

Instance and Schema

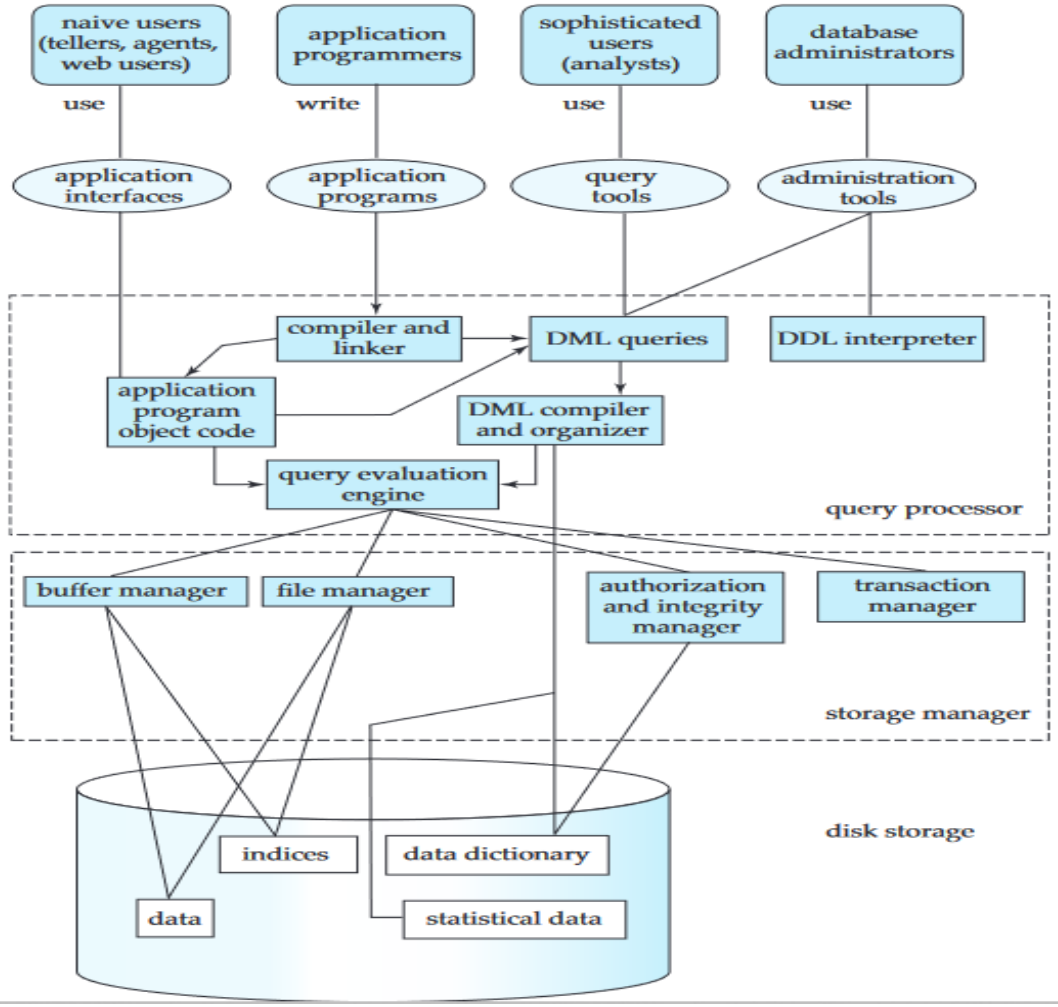
- Design of database at view level is called **view schema**. This generally describes end user interaction with database systems.



Instance and Schema

- **Definition of instance**: The data stored in database at a particular moment of time is called instance of database. Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.
- For example, lets say we have a single table student in the database, today the table has 100 records, so today the instance of the database has 100 records.

Components of Database System



Components of Database System

Storage Manager:

- The storage manager is the component of a database system that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible for the interaction with the file manager.
- The storage manager is responsible for storing, retrieving, and updating data in the database

Components of Database System

Components of Storage Manager:

- Authorization and Integrity Manager
- Transaction Manager
- File Manager
- Buffer Manager

Components of Database System

Data Structures implemented by Storage Manager:

- Data Files
- Data Dictionary
- Indices

Components of Database System

Query Processor:

The query processor components include:

- **DDL interpreter**, which interprets DDL statements and records the definitions in the data dictionary.
- **DML compiler**, which translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.
- **Query evaluation engine**, which executes low-level instructions generated by the DML compiler.

Components of Database System

Transaction Management:

- Transaction Management is a crucial component in database systems, responsible for ensuring the consistency, isolation, and durability of transactions.
- A transaction is a collection of operations that performs a single logical function in a database application.
- Each transaction is a unit of both atomicity and consistency.

Database System Architecture

The architecture of DBMS depends on the computer system on which it runs. For example, in a client-server DBMS architecture, the database systems at server machine can run several requests made by client machine.

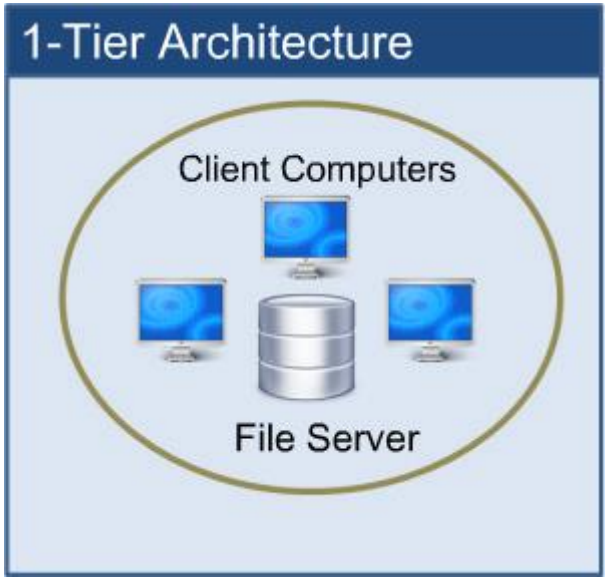
Types of DBMS Architecture

- There are three types of DBMS architecture:

    1. Single tier architecture
    2. Two tier architecture
    3. Three tier architecture

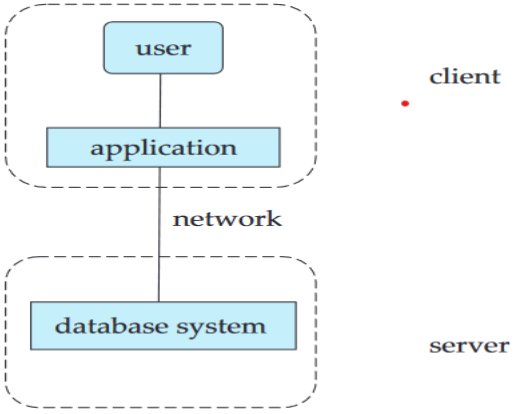
Single tier architecture

- Single-tier architecture involves putting all of the required components for a software application or technology on a single server or platform.



Two tier architecture

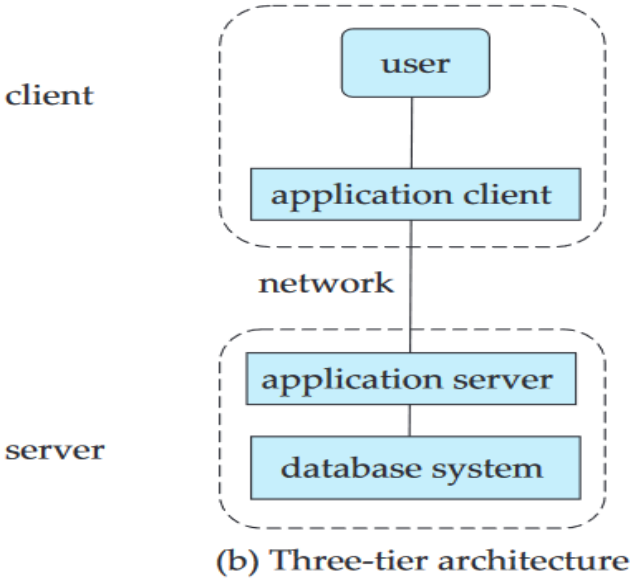
- In a two-tier architecture, the application resides at the client machine, where it invokes database system functionality at the server machine through query language statements. Application program interface standards like ODBC and JDBC are used for interaction between the client and the server.



(a) Two-tier architecture

Three tier architecture

- In a three-tier architecture, the client machine acts as merely a front end and does not contain any direct database calls.
- Instead, the client end communicates with an application server, usually through a forms interface.



(b) Three-tier architecture

Three tier architecture

- The application server in turn communicates with a database system to access data.
- The business logic of the application, which says what actions to carry out under what conditions, is embedded in the application server, instead of being distributed across multiple clients.
- Three-tier applications are more appropriate for large applications, and for applications that run on the World Wide Web.

THANK YOU