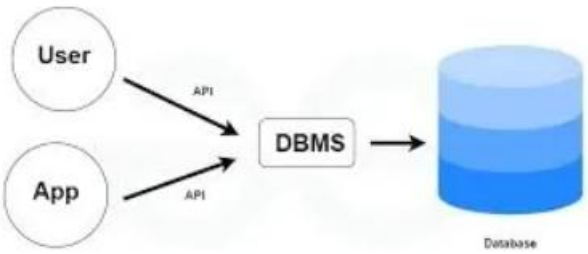
	Pithapur Rajahs Government College(A) Kakinada **Department of Computer Science**	**Program:** II B.Sc. (AI) **Semester :** III
Course 5	Course Name: DOCUMENT ORIENTED DATABASE	
	Hours Allocated: 3hrs/week	Credits: 3

UNIT – I SYLLABUS

Overview of Database Management Systems:
Introduction ,Data and Information , Characteristics of the Database Approach - Self-Describing Nature of the a Database System , Insulation between Programs and Data, Data Abstraction , Support of Multiple Views of the data, Sharing of Data and multiuser Transaction Processing , Actors on the Scene - Database Administrators , Database Designers , End Users , Sophisticated Users, Temporary Users, Specialized users, System Analysts and Application Programmers , Advantages of using a DBMS - Controlling Redundancy ,Restricting unauthorized Access , Permitting Inferencing and Actions Using Rules ,Providing Multiple User Interfaces , Representing Complex Relationships Among data , Enforcing Integrity Constraints , Providing Backup and Recovery, Database System Concepts and Architecture , DBMS Architecture and Data Independence - The Three-Schema Architecture , Data Independence , Database Languages and Interfaces.

Introduction of DBMS

A Database Management System (DBMS) is a software solution designed to efficiently manage organize and retrieve data in a structured manner.



- A DBMS is a system that allows users to create, modify and query databases while ensuring data integrity, security and efficient data access.
- Unlike traditional file systems, DBMS minimizes data redundancy, prevents inconsistencies and simplifies data management with features like concurrent access and backup mechanisms.
- DBMS plays a vital role in supporting data-driven decision-making and operational efficiency.

Key Features of DBMS

1. **Data Modeling**: Tools to create and modify data models, defining the structure and relationships within the database.
2. **Data Storage and Retrieval**: Efficient mechanisms for storing data and executing queries to retrieve it quickly.
3. **Concurrency Control**: Ensures multiple users can access the database simultaneously without conflicts.
4. **Data Integrity and Security**: Enforces rules to maintain accurate and secure data, including access controls and encryption.
5. **Backup and Recovery**: Protects data with regular backups and enables recovery in case of system failures.

What is Data?

Data is a raw and unorganized fact that is required to be processed to make it meaningful. It can be considered as facts and statistics collected together for reference or analysis.

Types of Data

There are two types of Data:

1. **Quantitative:** Quantitative data refers to numerical information like weight, height, etc.
2. **Qualitative:** Qualitative data refers to non-numeric information like opinions, perceptions, etc.

What is Information?

Information is defined as structured, organized, and processed data, presented within a context that makes it relevant and useful to the person who needs it. Data suggests that raw facts and figures regarding individuals, places, or the other issue, that is expressed within the type of numbers, letters or symbols.

Characteristics of the Database Approach

There are different characteristics of the database approach from the much older approach of programming with files. In a traditional file processing system, each user defines and implements its own modifications to the files needed for a selected software application as a part of programming the appliance. In the database approach, one repository maintains data which is defined once then accessed by various users in that database. In a file system, it will be independently like an application that is free to name elements. In comparison, during a database, the names or labels of knowledge are defined once and used repeatedly by queries, transactions, and applications.

Characteristics of Database Approach

Some of the most important characteristics of the database approach to the file processing approach are the following as follows.

Approach-1 : Self-Describing Nature of a Database System

- One of the most fundamental characteristics of the database approach is that the database system contains not only the database itself but also an entire definition or description of the database structure and constraints also known as metadata of the database.
- This definition is stored within the DBMS catalog, which contains information like the structure of every file, the sort and storage format of every data item, and various constraints/rules on the information.
- The knowledge stored within the catalog is named meta-data, and it describes the structure of the first database The catalog is employed by the DBMS software and also by database users such as database administrators who required to know the information about the database structure.
- A general-purpose DBMS software package is not written for a selected database application. Therefore, it must ask the catalog to understand the structure of the files during a specific database, like the sort and format of knowledge it will access.

Approach-2 : Insulation between Programs and Data, and Data Abstraction

- In a traditional file processing system, the structure of database knowledge files is embedded within the application programs, so any changes to the structure of a file may require changing all programs that access that file.
- Against this, DBMS access programs don't require such changes in most cases, so independence is achieved between them.
- The structure of knowledge files is stored within the DBMS catalog separately from the programs that access them. We call this property program-data independence.
- The characteristic that allows program-data independence and program-operation independence is known as data abstraction.

Approach-3 : Support for Multiple Views of the Data

- A database sometimes has many users, each of whom may require a special perspective or view of the database.
- A view could also be a subset of the database, or it's going to contain virtual data that is derived from the database files but isn't explicitly stored.
- Some users might not get to remember whether the information they ask for is stored or derived.
- A multi-user DBMS whose users have a spread of distinct applications must provide facilities for outlining multiple views. This provides many benefits for large databases such as the Aadhaar database.

Approach-4 : Sharing of knowledge and Multi-user Transaction Processing

- A multi-user DBMS, as its name implies, must allow multiple users to access the database at an equivalent time or concurrently.
- This is often essential if data for multiple applications is to be integrated and maintained during a single database such as the latest feature of WhatsApp integration with Facebook.
- The DBMS must implement concurrency control in the software to make sure that several users trying to update equivalent data do so in a controlled manner in order that the results of the updates are correct.
- For instance, when several reservation agents attempt to assign a seat on an airline flight, the DBMS should make sure that each seat is often accessed by just one user agent at a single time for an assignment to a passenger.

Different Types of Database Users

A Database User is defined as a person who interacts with data daily, updating, reading, and modifying the given data. Database users can access and retrieve data from the database through the Database Management System (DBMS) applications and interfaces.

Types of Database Users

Database users are categorized based on their interaction with the database. There are seven types of database users in DBMS. Below mentioned are the types of database users:

1. Database Administrator (DBA)

A Database Administrator (DBA) is a person/team who defines the schema and also controls the 3 levels of the database. The DBA will then create a new account ID and password for the user if he/she needs to access the database. DBA is also responsible for providing security to the database and he allows only authorized users to access/modify the database. DBA is responsible for problems such as security breaches and poor system response time.

DBA also monitors the recovery and backup and provides technical support.

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

DBA repairs damage caused due to hardware and/or software failures.

DBA is the one having privileges to perform DCL (Data Control Language) operations such as GRANT and REVOK, to allow/restrict a particular user from accessing the database.

2. Naive / Parametric End Users

Parametric End Users are the unsophisticated who don't have any DBMS knowledge but they frequently use the database applications in their daily life to get the desired results. For example, 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. A System Analyst

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

4. Sophisticated Users

Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database. They can develop their own database applications according to their requirement. They don't write the program code but they interact the database by writing SQL queries directly through the query processor.

5. Database Designers

Data Base Designers are the users who design the structure of database which includes tables, indexes, views, triggers, stored procedures and constraints which are usually enforced before the database is created or populated with data. He/she controls what data must be stored and how the data items to be related. It is the responsibility of Database Designers to understand the requirements of different user groups and then create a design which satisfies the need of all the user groups.

6. Application Programmers

Application Programmers also referred as System Analysts or simply Software Engineers, 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. Application programmers design, debug, test, and maintain set of programs called "canned transactions" for the Naive (parametric) users in order to interact with database.

7. Casual Users / Temporary Users

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

8. Specialized users

Specialized users are sophisticated users who write specialized database application that does not fit into the traditional data-processing framework. Among these applications are computer aided-design systems, knowledge-base and expert systems etc.

Advantages of Database Management System

Database Management System (DBMS) is a collection of interrelated data and a set of software tools/programs that access, process, and manipulate data. It allows access, retrieval, and use of that data by considering appropriate security measures. The Database Management system (DBMS) is really useful for better data integration and security.

The advantages of database management systems are:

1. **Data Security:** The more accessible and usable the database, the more it is prone to security issues. As the number of users increases, the data transferring or data sharing rate also increases thus increasing the risk of data security. It is widely used in the corporate world where companies invest large amounts of money, time, and effort to ensure data is secure and used properly. A Database Management System (DBMS) provides a better platform for data privacy and security policies thus, helping companies to improve Data Security.
2. **Data integration:** Due to the Database Management System we have access to well-managed and synchronized forms of data thus it makes data handling very easy and gives an integrated view of how a particular organization is working and also helps to keep track of how one segment of the company affects another segment.
3. **Data abstraction:** The major purpose of a database system is to provide users with an abstract view of the data. Since many complex algorithms are used by the developers to increase the efficiency of databases that are being hidden by the users through various data abstraction levels to allow users to easily interact with the system.
4. **Reduction in data Redundancy:** When working with a structured database, DBMS provides the feature to prevent the input of duplicate items in the database. for e.g. – If there are two same students in different rows, then one of the duplicate data will be deleted.
5. **Data sharing:** A DBMS provides a platform for sharing data across multiple applications and users, which can increase productivity and collaboration.
6. **Data consistency and accuracy:** DBMS ensures that data is consistent and accurate by enforcing data integrity constraints and preventing data duplication. This helps to eliminate data discrepancies and errors that can occur when data is stored and managed manually.
7. **Data organization:** A DBMS provides a systematic approach to organizing data in a structured way, which makes it easier to retrieve and manage data efficiently.
8. **Efficient data access and retrieval:** DBMS allows for efficient data access and retrieval by providing indexing and query optimization techniques that speed up data retrieval. This reduces the time required to process large volumes of data and increases the overall performance of the system.
9. **Concurrency and maintained Atomicity**: That means, if some operation is performed on one particular table of the database, then the change must be reflected for the entire database. The DBMS allows concurrent access to multiple users by using the synchronization technique.
10. **Scalability and flexibility:** DBMS is highly scalable and can easily accommodate changes in data volumes and user requirements. DBMS can easily handle large volumes of data, and can scale up or down depending on the needs of the organization. It provides flexibility in data storage, retrieval, and manipulation, allowing users to easily modify the structure and content of the database as needed.

Controlling Redundancy

Redundancy means having multiple copies of the same data in the database. This problem arises when a database is not normalized. Suppose a table of student details attributes is: student ID, student name, college name, college rank, and course opted.

Student_ID	Name	Contact	College	Course	Rank
100	Mahesh	7300934851	GEU	CS	1
101	Sriram	7900734858	GEU	B.Tech	1
102	Rajkumar	7300936759	GEU	CS	1
103	Sekhar	7300901556	GEU	M.Tech	1

It can be observed that values of attribute college name, college rank, and course are being repeated which can lead to problems. Problems caused due to redundancy are:

- Insertion anomaly
- Deletion anomaly
- Updation anomaly

Insertion Anomaly

If a student detail has to be inserted whose course is not being decided yet then insertion will not be possible till the time course is decided for the student.

Student_ID	Name	Contact	College	Course	Rank
100	Mahesh	7300934851	GEU		1

This problem happens when the insertion of a data record is not possible without adding some additional unrelated data to the record.

Deletion Anomaly

If the details of students in this table are deleted then the details of the college will also get deleted which should not occur by common sense. This anomaly happens when the deletion of a data record results in losing some unrelated information that was stored as part of the record that was deleted from a table.

It is not possible to delete some information without losing some other information in the table as well.

Updation Anomaly

Suppose the rank of the college changes then changes will have to be all over the database which will be time-consuming and computationally costly.

Student_ID	Name	Contact	College	Course	Rank
100	Mahesh	7300934851	GEU	CS	1
101	Sriram	7900734858	GEU	B.Tech	1
102	Rajkumar	7300936759	GEU	CS	1
103	Sekhar	7300901556	GEU	M.Tech	1

All places should be updated, If updation does not occur at all places then the database will be in an inconsistent state.

Redundancy in a database occurs when the same data is stored in multiple places. Redundancy can cause various problems such as data inconsistencies, higher storage requirements, and slower data retrieval.

Restricting unauthorized Access

Data Protection and Privacy: Databases often store sensitive information, including personal, financial, and proprietary data. Unauthorized access can lead to data breaches, which may expose individuals' personal information and compromise privacy. Protecting against unauthorized access helps ensure compliance with data protection regulations, like GDPR, HIPAA, and others, which mandate safeguarding personal and sensitive data.

Maintaining Data Integrity: Unauthorized access increases the risk of data manipulation, which can corrupt the database's integrity. Data integrity is critical for accurate analysis and decision-making, and unauthorized users may alter, delete, or falsify records, undermining the reliability of the information stored.

Preventing Financial Loss and Fraud: Unauthorized access to databases can result in theft of valuable data or intellectual property, leading to direct financial loss. Hackers or malicious insiders might exploit data for financial gain, commit fraud, or sell proprietary information. Restricting access reduces these risks and protects the organization from potentially significant financial harm.

Ensuring Operational Continuity: Many organizations rely on databases for daily operations, and unauthorized access can disrupt these operations. Malicious users might delete or corrupt critical records, affecting business functions and leading to costly downtime. Access restrictions help maintain the continuity of operations by ensuring that only authorized personnel can make changes to vital data.

Representing Complex Relationships Among data

A relationship in a DBMS exists when a variable has a connection with the properties stored in different tables. Such relationships help the organization of entities intertwined with each other, ultimately enabling efficient data processing. They're exhibited usually via keys in a table, which is either columns or fields that specify a distinctive arrangement for each record.

Types of Relationships in a Database

1. One-to-One (1:1) Relationship

In one to one relationships, a record is present in one table along with its corresponding existing relation, and the vacant relation among the records is present in another table. The type of relationship we are talking about is not as usual, and it is normally used when two entities that belong to a specific set need to be stored independently for normalization or security purposes. In another case, a person's employees' data consists of a record in the "personal details" table in a human resources database.

2. One-to-Many (1:N) Relationship

A relationship where the items from one table can be linked to only one or many items from another table is called a one-to-many relationship; in some cases, one item from the first table correlates with only one item from the second table. This connection becomes very strong in that it is particularly used to describe situations where one object can be linked to many similar or identical objects. For example, in an online store backend database, every customer may place multiple orders, yet the master customer record stays the same. If a record has more than one order, these are obtained from the backend database.

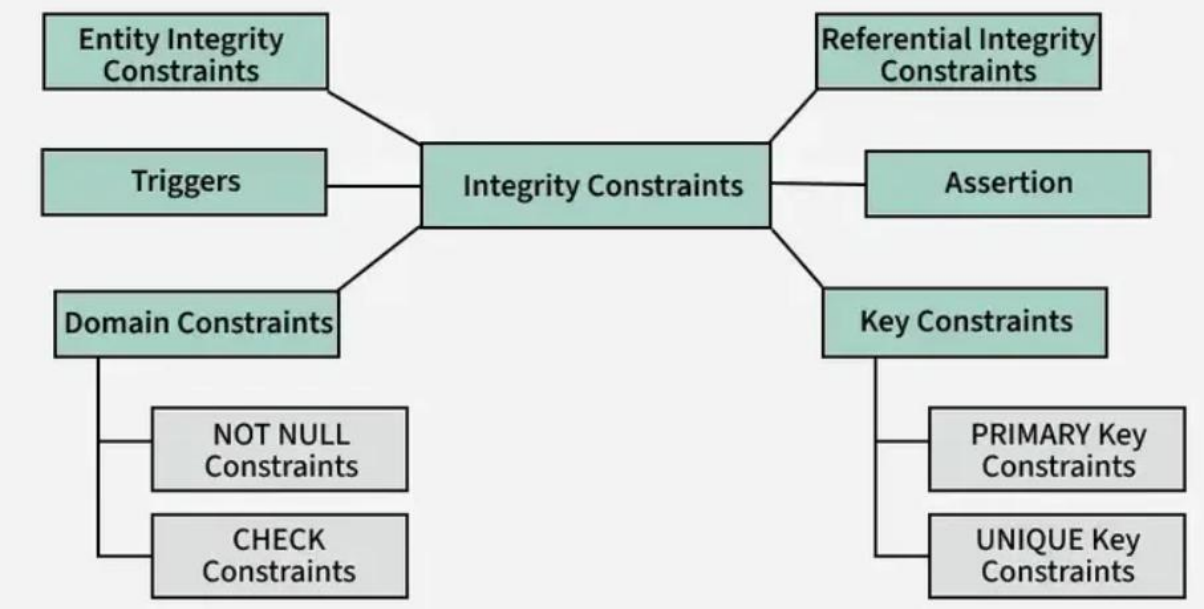
3. Many-to-Many (N:M) Relationship

The duality of a many-to-many relationship is characterized by the presence of multiple records belonging to a table in association with multiple records from another table. The interconnection of these relationships follows a junction table format, which is the component that holds both tables together. In the many-to-many relationship model, a wide variety of complex relationships can be established where each entity has many related entities. Such a database for a music streaming service could have a table representing each track that belongs to multiple playlists, and each of them could contain multiple tracks.

Enforcing Integrity Constraints

Integrity constraints are a set of rules used in DBMS to ensure that the data in a database is accurate, consistent and reliable. These rules helps in maintaining the quality of data by ensuring that the processes like adding, updating or deleting information do not harm the integrity of the database.

Integrity constraints also define how different parts of the database are connected and ensure that these relationships remain valid. They play an essential role in making sure the data is meaningful and follows the logical structure of the database.



Providing Backup and Recovery

What is Backup?

Backup refers to storing a copy of original data which can be used in case of data loss. Backup is considered one of the approaches to data protection. Important data of the organization needs to be efficiently backed up to protect valuable data. Backup can be achieved by storing a copy of the original data separately or in a database on storage devices. There are various types of backups available like full backup, incremental backup, Local backup, mirror backup, etc.

Advantages of Backup

- Protection against data loss.
- Ensuring smooth workflow.
- Enables recovery of previous data.
- User can back up the data and delete it from their system to free the space or memory.

Disadvantages of Backup

- Hardware and software expenses are higher.
- Maintenance cost of hardware and software increases.
- Failure to properly backup critical data can result in irreversible loss.

What is Recovery?

Recovery refers to restoring lost data by following some processes. Even if the data was backed up still lost so it can be recovered by using/implementing some recovery techniques. When a database fails due to any reason then there is a chance of data loss, so in that case recovery process helps in improve the reliability of the database.

Example: Recuva is a data recovery tool. Using Recuva you can restore lost and deleted files.

Advantages of Recovery

- It prevents permanent loss of data.
- Use of recovery tools is cost-effective.
- Helps in disaster recovery.

Disadvantages of Recovery

- Data Recovery is not always guaranteed.
- The Data Recovery tools may be expensive.
- Using untrustworthy or poorly developed data recovery software increases security risk.

DBMS Architecture

A DBMS architecture defines how users interact with the database to read, write, or update information. A well-designed architecture and schema (a blueprint detailing tables, fields and relationships) ensure data consistency, improve performance and keep data secure.

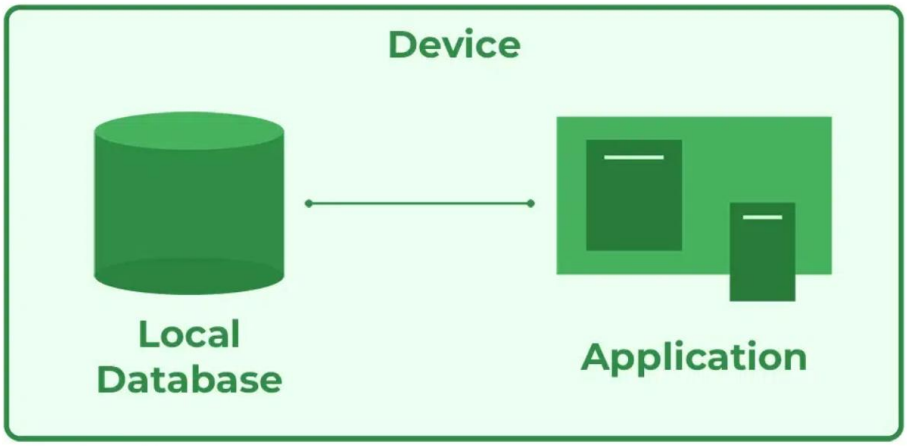
Types of DBMS Architecture

There are several types of DBMS Architecture that we use according to the usage requirements.

- 1-Tier Architecture
- 2-Tier Architecture
- 3-Tier Architecture

1-Tier Architecture

In 1-Tier Architecture, the user works directly with the database on the same system. This means the client, server and database are all in one application. The user can open the application, interact with the data and perform tasks without needing a separate server or network connection.



- A common example is Microsoft Excel. Everything from the user interface to the logic and data storage happens on the same device. The user enters data, performs calculations and saves files directly on their computer.
- This setup is simple and easy to use, making it ideal for personal or standalone applications. It does not require a network or complex setup, which is why it's often used in small-scale or individual use cases.
- This architecture is simple and works well for personal, standalone applications where no external server or network connection is needed.

Advantages of 1-Tier Architecture

Below mentioned are the advantages of 1-Tier Architecture.

- **Simple Architecture:** 1-Tier Architecture is the most simple architecture to set up, as only a single machine is required to maintain it.
- **Cost-Effective:** No additional hardware is required for implementing 1-Tier Architecture, which makes it cost-effective.
- **Easy to Implement:** 1-Tier Architecture can be easily deployed and hence it is mostly used in small projects.

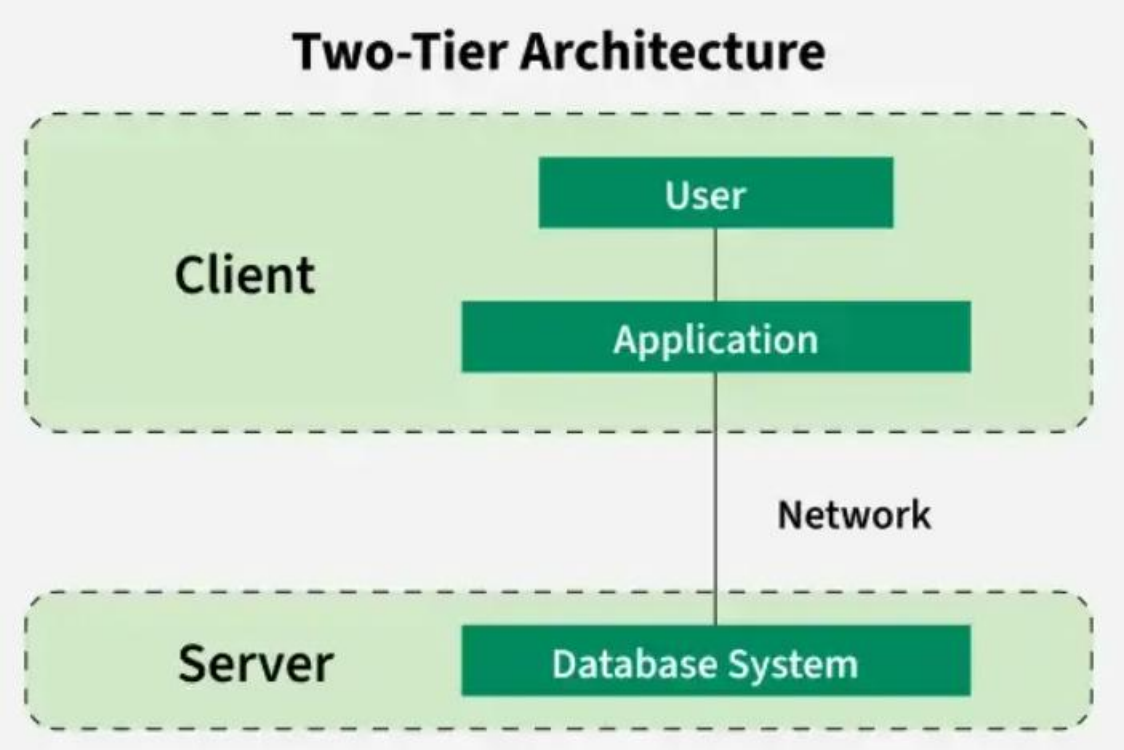
Disadvantages of 1-Tier Architecture

- **Limited to Single User:** Only one person can use the application at a time. It’s not designed for multiple users or teamwork.
- **Poor Security:** Since everything is on the same machine, if someone gets access to the system, they can access both the data and the application easily.
- **No Centralized Control:** Data is stored locally, so there's no central database. This makes it hard to manage or back up data across multiple devices.
- **Hard to Share Data:** Sharing data between users is difficult because everything is stored on one computer.

2-Tier Architecture

The 2-tier architecture is similar to a basic client-server model. The application at the client end directly communicates with the database on the server side. APIs like ODBC and JDBC are used for this interaction. The server side is responsible for providing query processing and transaction management functionalities.

- On the client side, the user interfaces and application programs are run. The application on the client side establishes a connection with the server side to communicate with the DBMS. For Example: A Library Management System used in schools or small organizations is a classic example of two-tier architecture.
- **Client Layer (Tier 1):** This is the user interface that library staff or users interact with. For example they might use a desktop application to search for books, issue them, or check due dates.



- **Database Layer (Tier 2):** The database server stores all the library records such as book details, user information and transaction logs.
- The client layer sends a request (like searching for a book) to the database layer which processes it and sends back the result. This separation allows the client to focus on the user interface, while the server handles data storage and retrieval.

Advantages of 2-Tier Architecture

- **Easy to Access:** 2-Tier Architecture makes easy access to the database, which makes fast retrieval.
- **Scalable:** We can scale the database easily, by adding clients or upgrading hardware.
- **Low Cost:** 2-Tier Architecture is cheaper than 3-Tier Architecture and Multi-Tier Architecture.
- **Easy Deployment:** 2-Tier Architecture is easier to deploy than 3-Tier Architecture.
- **Simple:** 2-Tier Architecture is easily understandable as well as simple because of only two components.

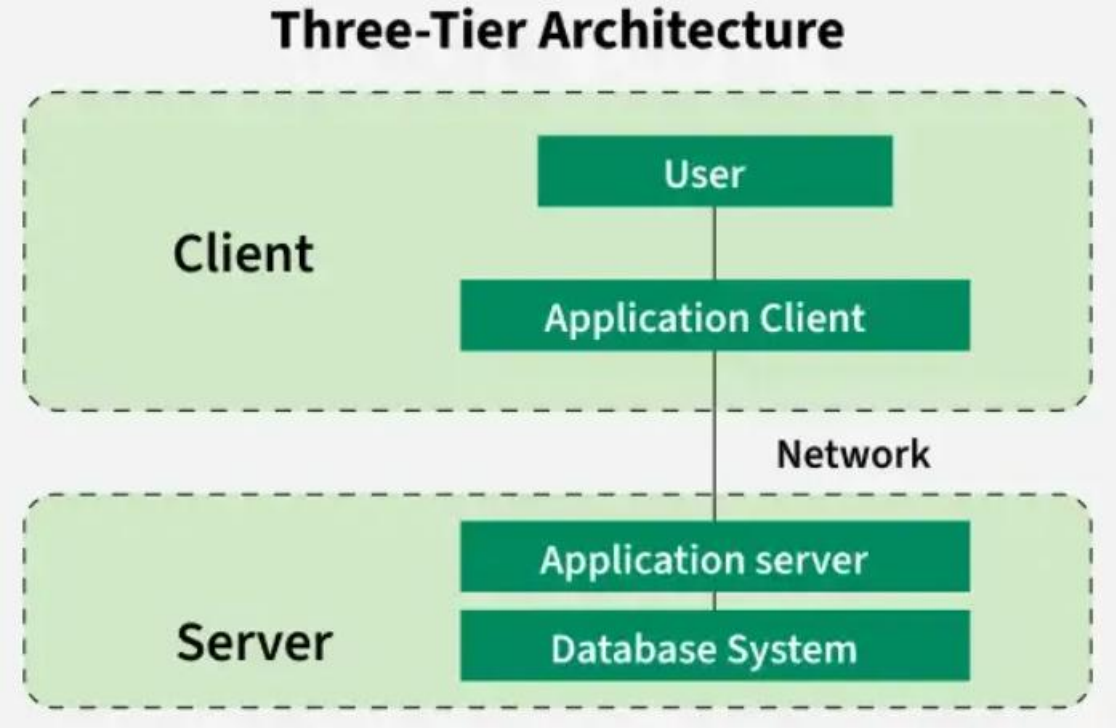
Disadvantages of 2-Tier Architecture

- **Limited Scalability:** As the number of users increases, the system performance can slow down because the server gets overloaded with too many requests.
- **Security Issues:** Clients connect directly to the database, which can make the system more vulnerable to attacks or data leaks.

- **Tight Coupling:** The client and the server are closely linked. If the database changes, the client application often needs to be updated too.
- **Difficult Maintenance:** Managing updates, fixing bugs, or adding features becomes harder when the number of users or systems increases.

3-Tier Architecture

In 3-Tier Architecture, there is another layer between the client and the server. The client does not directly communicate with the server. Instead, it interacts with an application server which further communicates with the database system and then the query processing and transaction management takes place. This intermediate layer acts as a medium for the exchange of partially processed data between the server and the client. This type of architecture is used in the case of large web applications.



Example: E-commerce Store

- **User**: You visit an online store, search for a product and add it to your cart.
- **Processing**: The system checks if the product is in stock, calculates the total price and applies any discounts.
- **Database**: The product details, your cart and order history are stored in the database for future reference.

Advantages of 3-Tier Architecture

- **Enhanced scalability:** Scalability is enhanced due to the distributed deployment of application servers. Now, individual connections need not be made between the client and server.

- **Data Integrity:** 3-Tier Architecture maintains Data Integrity. Since there is a middle layer between the client and the server, data corruption can be avoided/removed.
- **Security:** 3-Tier Architecture Improves Security. This type of model prevents direct interaction of the client with the server thereby reducing access to unauthorized data.

Disadvantages of 3-Tier Architecture

- **More Complex:** 3-Tier Architecture is more complex in comparison to 2-Tier Architecture. Communication Points are also doubled in 3-Tier Architecture.
- **Difficult to Interact:** It becomes difficult for this sort of interaction to take place due to the presence of middle layers.
- **Slower Response Time:** Since the request passes through an extra layer (application server), it may take more time to get a response compared to 2-Tier systems.
- **Higher Cost:** Setting up and maintaining three separate layers (client, server and database) requires more hardware, software and skilled people. This makes it more expensive.

DBMS Architecture and Data Independence

DBMS Architecture:

A DBMS is typically structured in layers, with the most common being the three-schema architecture:

1. **1. Internal Level/Schema:**

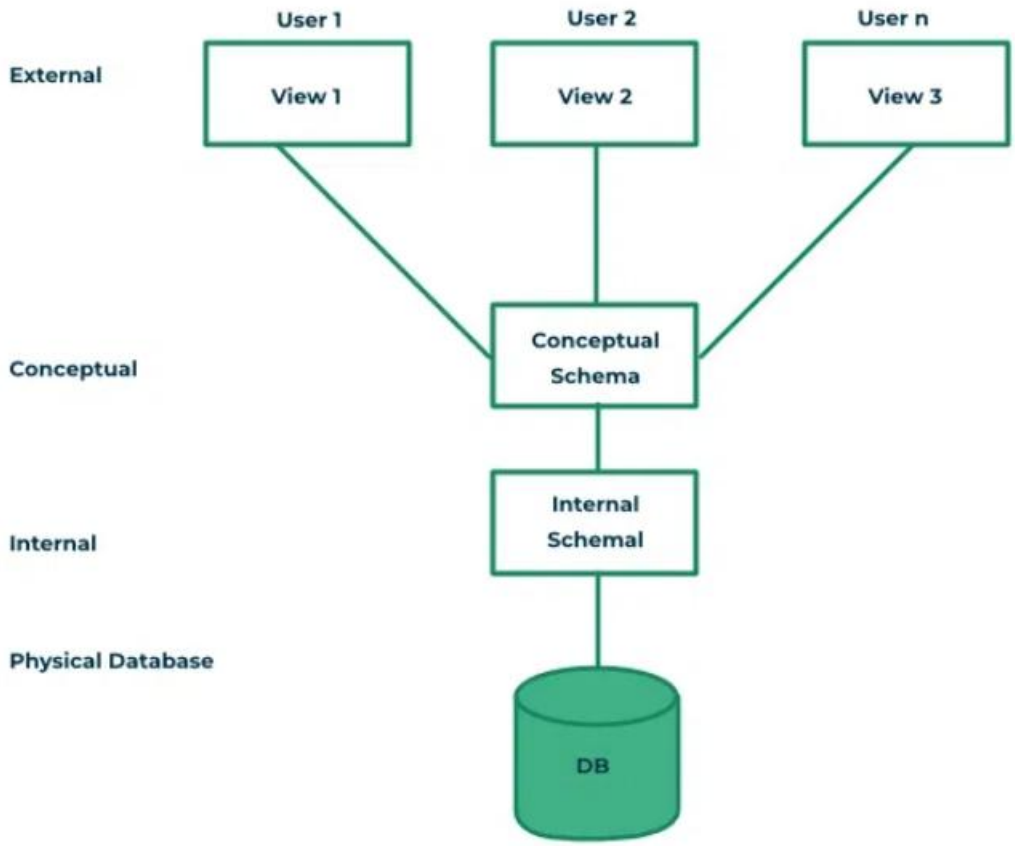
This level describes the physical storage of data, including how data is stored on disk, file structures, and access paths.

2. **2. Conceptual Level/Schema:**

This level represents the overall logical structure of the database, including entities, attributes, and relationships. It provides a unified view of the data for all users.

3. **3. External Level/Schema:**

This level represents the different views of the database that users have. Each user or application may only need to access a subset of the data and may have their own specific view of it.



Data Independence:

Data independence is the ability to modify the database schema at one level without affecting the schema at other levels. This means that changes at the physical level (e.g., storage structure) or the logical level (e.g., adding a new table) won't require changes in the application programs that use the database.

There are two main types of data independence:

1. **1. Logical Data Independence:**

This refers to the ability to modify the conceptual schema (logical structure) without affecting the external schemas (user views). For example, adding a new attribute to a table without requiring changes to the user interfaces that don't need that attribute.

2. **2. Physical Data Independence:**

This refers to the ability to modify the internal schema (physical storage) without affecting the conceptual schema. For example, changing the storage structure or using a different storage device without requiring changes to the logical structure of the database or the applications that use it.

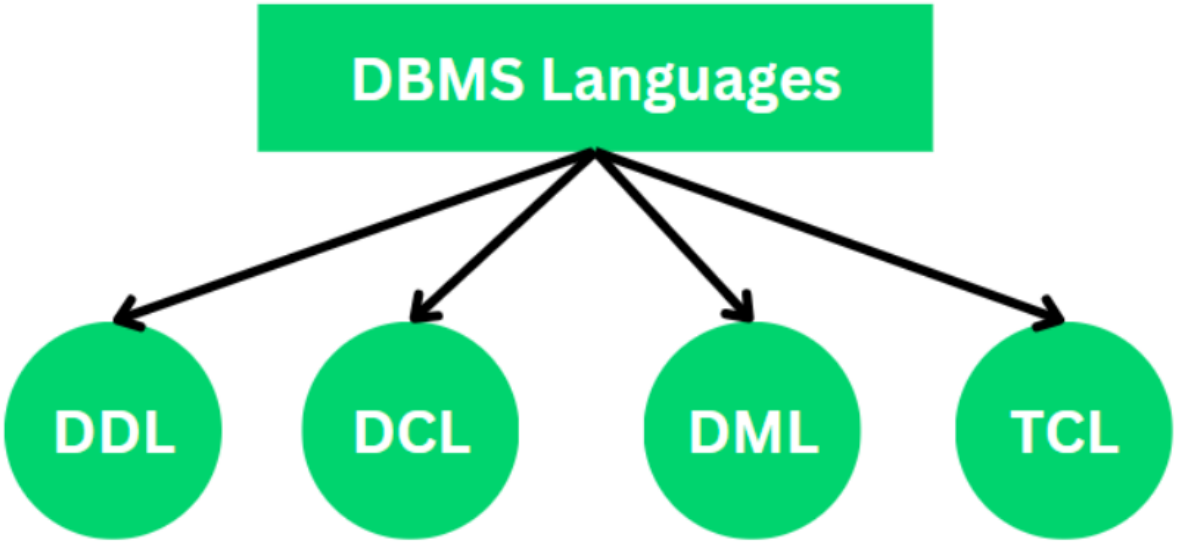
Benefits of Data Independence:

- **Flexibility:** Allows for changes to the database without impacting applications.

- **Maintainability:** Simplifies database maintenance and updates.
- **Scalability:** Enables efficient scaling of the database without requiring major application changes.
- **Cost Reduction:** Reduces the cost of changes and upgrades.

Database Languages in DBMS

Database languages are specialized languages used to interact with a database. They allow users to perform different tasks such as defining, controlling and manipulating the data. There are several types of database languages in DBMS, categorized into the following four main types:



1. DDL (Data Definition Language)
2. DCL (Data Control Language)
3. DML (Data Manipulation Language)
4. TCL (Transaction Control Language)

1. DDL (Data Definition Language)

The DDL stands for Data Definition Language, Which is used to define the database's internal structure and Pattern of the Database. It is used to define and modify the structure of the database itself, including the tables, views, indexes and other schema-related objects. It deals with the creation and modification of database schema, but it doesn't deal with the data itself. Following are the five DDL commands in SQL:

- **CREATE:** Used to create database objects like tables, indexes or views.
- **ALTER:** Used to modify the structure of an existing database object, such as adding a new column to a table.
- **DROP:** Used to delete database objects.

- **TRUNCATE:** Used to remove all rows from a table, without affecting the structure.
- **RENAME:** Used to change the name of a database object.

2. DCL (Data Control Language)

DCL stands for Data Control Language. It is used to control the access permissions of users to the database. DCL commands help grant or revoke privileges to users, determining who can perform actions like reading or modifying data. DCL commands are transactional, meaning they can be rolled back if necessary. The two main DCL commands are:

- **Grant:** Gives user access to the database
- **Revoke:** Removes access or permissions from the user

3. DML (Data Manipulation Language)

The DML (Data Manipulation Language) is used toto manage and manipulate data within a database. With DML, you can perform various operations such as inserting, updating, selecting and deleting data. These operations allow you to work with the actual content in your database tables. Here are the key DML commands:

- **SELECT:** Retrieves data from the table based on specific criteria.
- **INSERT:** Adds new rows of data into an existing table.
- **UPDATE:** Modifies existing data in a table.
- **DELETE:** Removes data from a table.
- **MERGE:** Performs an "upsert" operation, which means updating existing records or inserting new ones if they don’t exist.
- **CALL:** Executes stored procedures or functions.
- **LOCK TABLE:** Prevents other users from accessing the table while changes are being made.

4. TCL (Transaction Control Language)

The TCL full form is Transaction Control Language commands are used to manage and control transactions in a database, grouping them into logical units. These commands help ensure the integrity of data and consistency during complex operations. Here are the two main commands in this category:

- **Commit:** Saves all the changes made during the current transaction to the database. These are very useful in the banking sector.
- **Rollback:** used to restore the database to its original state from the last commit. This command also plays an important role in Banking Sectors.