Introduction of DBMS (Database Management System)

A **Database Management System (DBMS)** is a software solution designed to efficiently **manage**, **organize**, and **retrieve data** in a structured manner. It serves as a critical component in **modern computing**, **enabling** organizations to **store**, **manipulate**, and secure their data effectively. From small applications to enterprise systems, DBMS plays a vital role in supporting **data-driven decision-making** and operational efficiency.

What is a DBMS?

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**. It organizes data into **tables**, views, **schemas**, and reports, providing a structured approach to data management.

Example:

A university database can store and manage student information, faculty records, and administrative data, allowing seamless retrieval, insertion, and deletion of information as required.

Key Features of DBMS

1. **Data Modelling**: 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.

Types of DBMS

There are several types of Database Management Systems (DBMS), each tailored to different data structures, scalability requirements, and application needs. The most common types are as follows:

1. Relational Database Management System (RDBMS)

RDBMS organizes data into tables (relations) composed of rows and columns. It uses primary keys to uniquely identify rows and foreign keys to establish relationships between tables. Queries are written in **SQL (Structured Query Language)**, which allows for efficient data manipulation and retrieval.

Examples: MySQL, Oracle, Microsoft SQL Server and Postgre SQL.

2. NoSQL DBMS

NoSQL systems are designed to handle **large-scale data** and provide high performance for scenarios where **relational models** might be restrictive. They store data in various non-relational formats, such as **key-value pairs**, **documents**, **graphs**, or **columns**. These flexible data models enable rapid scaling and are well-suited for unstructured or semi-structured data.

Examples: MongoDB, Cassandra, DynamoDB and Redis.

3. Object-Oriented DBMS (OODBMS)

OODBMS integrates object-oriented programming concepts into the **database environment**, allowing data to be stored as objects. This approach supports complex data types and relationships, making it ideal for applications requiring advanced data modeling and **real-world simulations**.

Examples: ObjectDB, db4o.

Database Languages

Database languages are specialized sets of **commands** and **instructions** used to define, manipulate, and control data within a database. Each language type plays a distinct role in database management, ensuring efficient **storage**, **retrieval**, and security of data. The primary database languages include:

1. Data Definition Language (DDL)

DDL is the short name for Data Definition Language, which deals with database schemas and descriptions, of how the data should reside in the database.

- **CREATE:** to create a database and its objects like (table, index, views, store procedure, function, and triggers)
- **ALTER:** alters the structure of the existing database
- **DROP:** delete objects from the database
- **TRUNCATE:** remove all records from a table, including all spaces allocated for the records are removed
- **COMMENT:** add comments to the data dictionary
- **RENAME:** rename an object

2. Data Manipulation Language (DML)

DML focuses on manipulating the data stored in the database, enabling users to retrieve, add, update, and delete data.

- **SELECT:** retrieve data from a database
- **INSERT:** insert data into a table
- **UPDATE:** updates existing data within a table
- **DELETE:** Delete all records from a database table
- **MERGE:** UPSERT operation (insert or update)
- **CALL:** call a PL/SQL or Java subprogram
- **EXPLAIN PLAN:** interpretation of the data access path
- **LOCK TABLE:** concurrency Control

3. Data Control Language (DCL)

DCL commands manage access permissions, ensuring data security by controlling who can perform certain actions on the database.

- **GRANT**: Provides specific privileges to a user (e.g., SELECT, INSERT).
- **REVOKE**: Removes previously granted permissions from a user.

4. Transaction Control Language (TCL)

TCL commands oversee transactional data to maintain consistency, reliability, and atomicity.

- **ROLLBACK**: Undoes changes made during a transaction.
- **COMMIT**: Saves all changes made during a transaction.
- **SAVEPOINT**: Sets a point within a transaction to which one can later roll back.

5. Data Query Language (DQL)

DQL is a subset of DML, specifically focused on data retrieval.

- **SELECT**: The primary DQL command, used to query data from the database without altering its structure or contents.

Paradigm Shift from File System to DBMS

Before the advent of **modern Database Management Systems** (DBMS), data was managed using basic file systems on **hard drives**. While this approach allowed users to **store**, **retrieve**, and **update files** as needed, it came with numerous challenges.

A typical example can be seen in a file-based **university management system**, where data was stored in separate sections such as Departments, Academics, Results, Accounts, and Hostels. Certain information like student names and phone numbers was repeated across **multiple files**, leading to the following issues:

1. Redundancy of data

When the same data exists in multiple places, any update must be **manually repeated everywhere**. For instance, if a student changes their phone number, it must be updated across all sections. Failure to do so leads to **unnecessary duplication** and wasted storage.

2. Inconsistency of Data

Data is said to be inconsistent if multiple copies of the same data do not match each other. If the Phone number is different in Accounts Section and Academics Section, it will be inconsistent. Inconsistency may be because of typing errors or not updating all copies of the same data.

3. Complex Data Access

A user should know the exact location of the file to access data, so the process is very cumbersome and tedious. If the user wants to search the student hostel allotment number of a student from 10000 unsorted students' records, how difficult it can be.

4. Lack of Security

File systems provided limited control over who could access certain data. A student who gained access to a file with grades might easily alter it without proper authorization, compromising data integrity.

5. No Concurrent Access

File systems were not designed for multiple users working at the same time. If one user was editing a file, others had to wait, which hindered collaboration and slowed down workflows.

6. No Backup and Recovery

File systems lacked built-in mechanisms for creating backups or recovering data after a loss. If a file was accidentally deleted or corrupted, there was no easy way to restore it, potentially causing permanent data loss.

Advantages of DBMS

1. **Data organization:** A DBMS allows for the organization and storage of data in a structured manner, making it easy to retrieve and query the data as needed.
2. **Data integrity:** A DBMS provides mechanisms for enforcing data integrity constraints, such as constraints on the values of data and access controls that restrict who can access the data.
3. **Concurrent access:** A DBMS provides mechanisms for controlling concurrent access to the database, to ensure that multiple users can access the data without conflicting with each other.
4. **Data security:** A DBMS provides tools for managing the security of the data, such as controlling access to the data and encrypting sensitive data.
5. **Backup and recovery:** A DBMS provides mechanisms for backing up and recovering the data in the event of a system failure.

6. **Data sharing:** A DBMS allows multiple users to access and share the same data, which can be useful in a collaborative work environment.

Disadvantages of DBMS

1. **Complexity:** DBMS can be complex to set up and maintain, requiring specialized knowledge and skills.
2. **Performance overhead:** The use of a DBMS can add overhead to the performance of an application, especially in cases where high levels of concurrency are required.
3. **Scalability:** The use of a DBMS can limit the scalability of an application, since it requires the use of locking and other synchronization mechanisms to ensure data consistency.
4. **Cost:** The cost of purchasing, maintaining and upgrading a DBMS can be high, especially for large or complex systems.
5. **Limited Use Cases:** Not all use cases are suitable for a DBMS, some solutions don't need high reliability, consistency or security and may be better served by other types of data storage.

Applications of DBMS

1. **Enterprise Information:** Sales, accounting, human resources, Manufacturing, online retailers.
2. **Banking and Finance Sector:** Banks maintaining the customer details, accounts, loans, banking transactions, credit card transactions. Finance: Storing the information about sales and holdings, purchasing of financial stocks and bonds.
3. **University:** Maintaining the information about student course enrolled information, student grades, staff roles.
4. **Airlines:** Reservations and schedules.
5. **Telecommunications:** Prepaid, postpaid bills maintance.

DBMS Architecture 1-level, 2-Level, 3-Level

A Database stores a lot of critical information to access data quickly and securely. Hence it is important to select the correct architecture for efficient data management. Database Management System (DBMS) architecture is crucial for efficient data management and system performance. It helps users to get their requests done while connecting to the database. It focuses on how the database is designed, built and maintained, shaping how users access and interact with it. This article explains different DBMS architectures like client/server systems and database models.

Types of DBMS Architecture

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

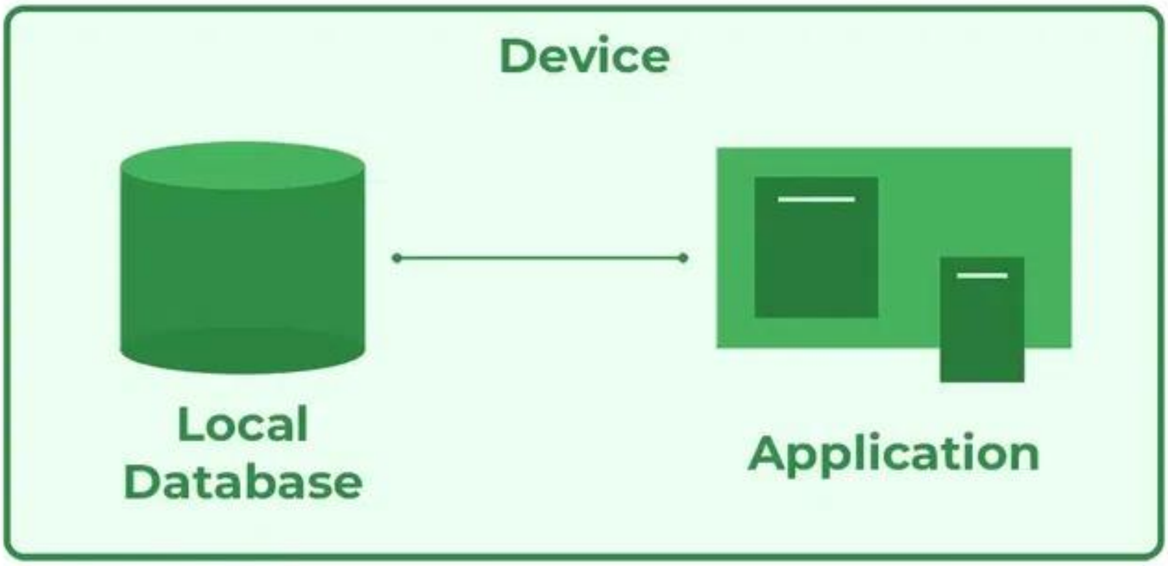
- 1-Tier Architecture
- 2-Tier Architecture
- 3-Tier Architecture

1-Tier Architecture

In 1-Tier Architecture the database is directly available to the user, the user can directly sit on the DBMS and use it that is, the client, server, and Database are all present on the same machine. This setup is simple and is often used in personal or standalone applications where the user interacts directly with the database.
For Example: A Microsoft Excel spreadsheet is a great example of one-tier architecture.

- Everything—the user interface, application logic and data is handled on a single system.
- The user directly interacts with the application, performs operations like calculations or data entry and stores data locally on the same machine.

This architecture is simple and works well for personal, standalone applications where no external server or network connection is needed.



DBMS 1-Tier Architecture

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.

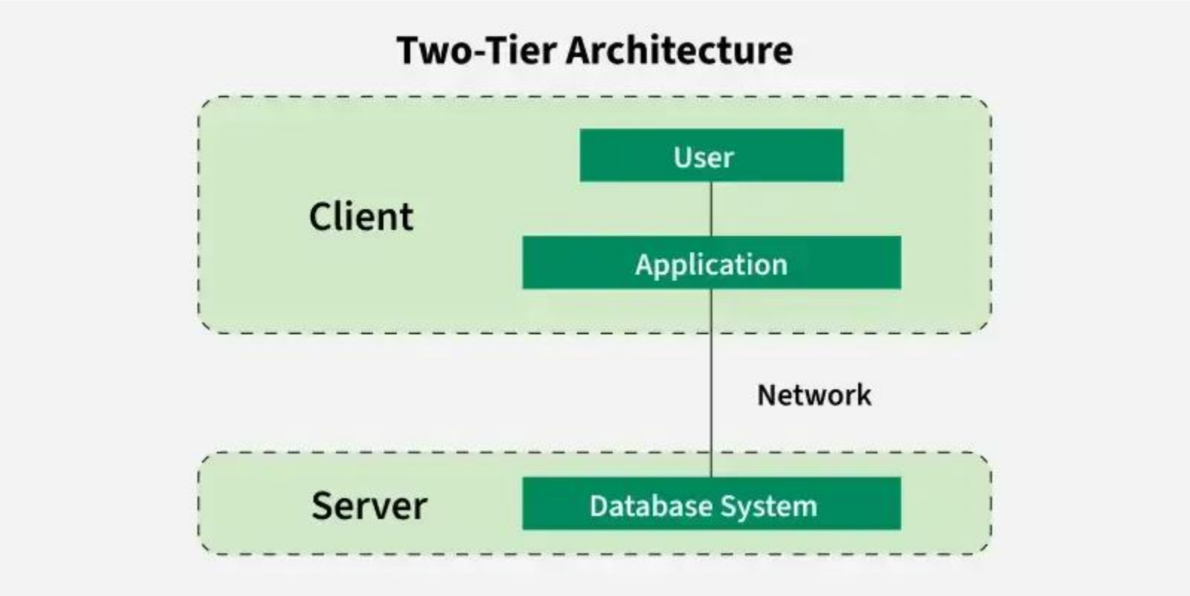
- **Easy to Implement:** 1-Tier Architecture can be easily deployed, and hence it is mostly used in small projects.

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.

1. **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.
2. **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.



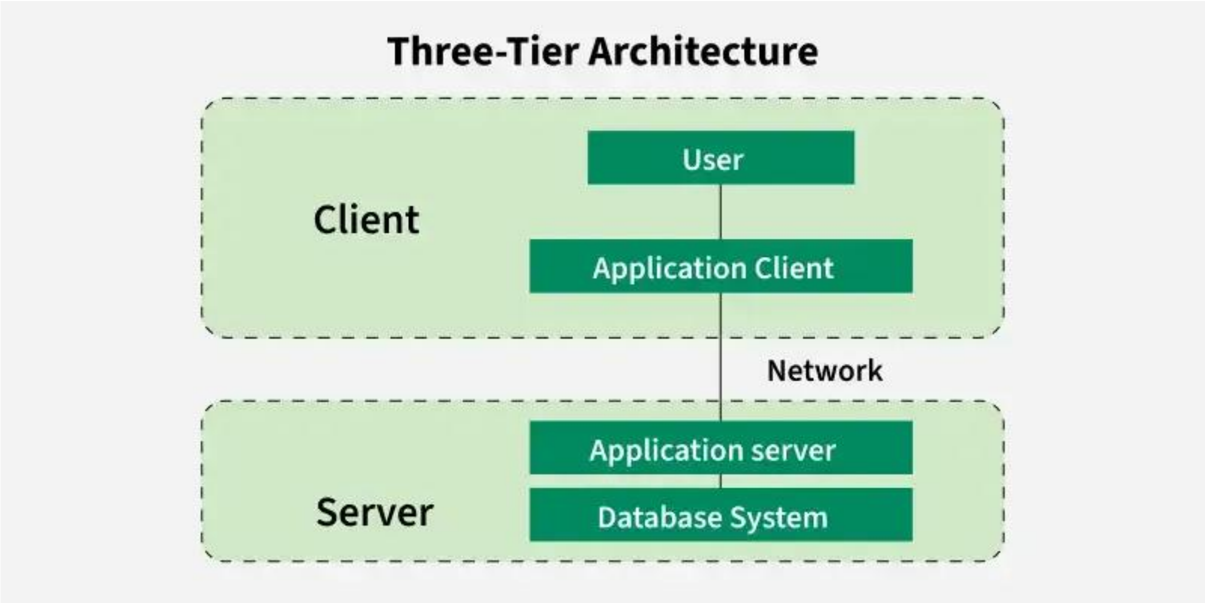
DBMS 2-Tier Architecture

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.

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.
For 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.



DBMS 3-Tier Architecture

Advantages of 3-Tier Architectures

- **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.
-

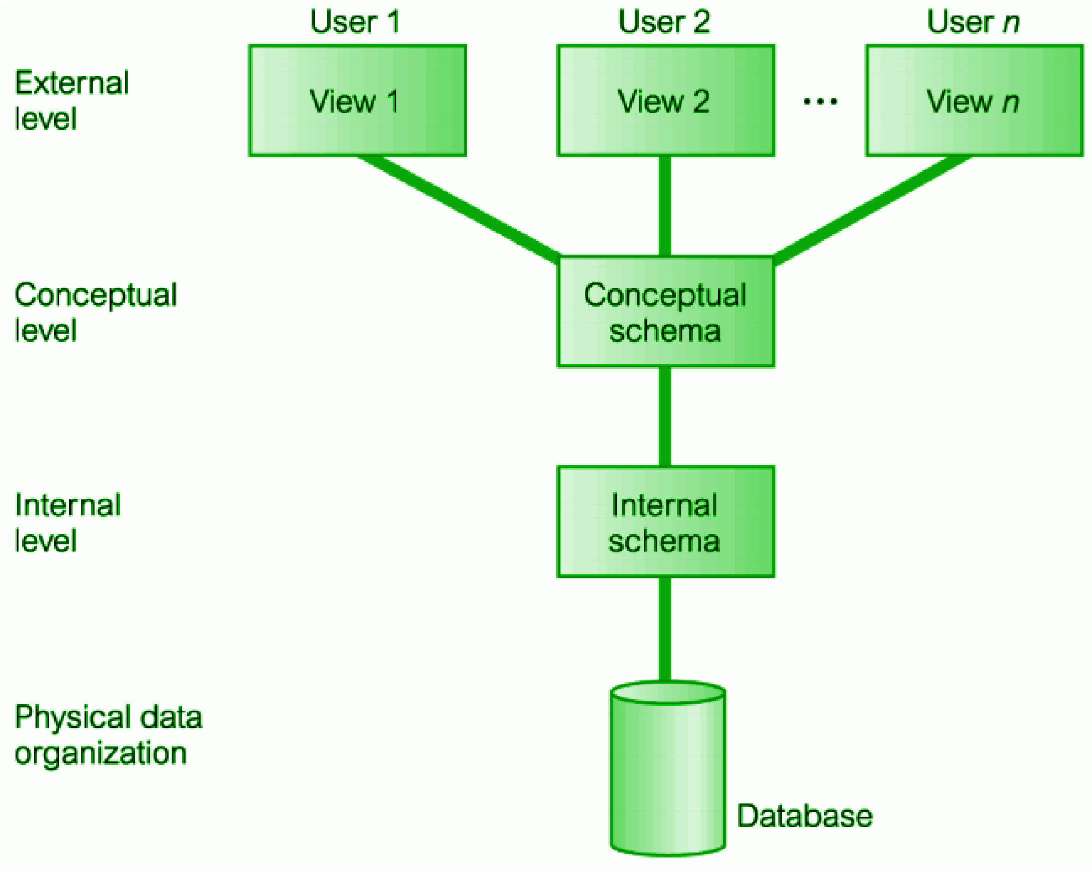
What is Data Independence in DBMS?

Data independence is a property of a database management system by which we can change the database schema at one level of the database system without changing the database schema at the next higher level. In this article, we will learn in full detail about data independence and will also see its types. If you read it completely, you will understand it easily.

What is Data Independence in DBMS?

In the context of a database management system, data independence is the feature that allows the schema of one layer of the database system to be changed without any impact on the schema of the next higher level of the database system. ” Through data independence, we can build an environment in which data is independent of all programs, and through the three schema architectures, data independence will be more understandable. Data via two card stencils along with centralized DBMS data is a form of transparency that has value for someone.

It can be summed up as a sort of immunity of user applications that adjusts correctly and does not change addresses, imparting the class of data and their order. I want the separate applications not to be forced to deal with data representation and storage specifics because this decreases quality and flexibility. DBMS permits you to see data with such a generalized sight. It actually means that the ability to change the structure of the lower-level schema without presenting the upper-level schema is called data independence.



Types of Data Independence

There are two types of data independence.

- logical data independence
- Physical data independence

Logical Data Independence

- Changing the logical schema (conceptual level) without changing the external schema (view level) is called logical data independence.
- It is used to keep the external schema separate from the logical schema.
- If we make any changes at the conceptual level of data, it does not affect the view level.
- This happens at the user interface level.
- For example, it is possible to add or delete new entities, attributes to the conceptual schema without making any changes to the external schema.

Physical Data Independence

- Making changes to the physical schema without changing the logical schema is called physical data independence.
- If we change the storage size of the database system server, it will not affect the conceptual structure of the database.

- It is used to keep the conceptual level separate from the internal level.
- This happens at the logical interface level.
- Example – Changing the location of the database from C drive to D drive.

Difference Between Physical and Logical Data Independence

Physical Data Independence	Logical Data Independence
It mainly concerns how the data is stored in the system.	It mainly concerns about changes to the structure or data definition.
It is easier to achieve than logical independence.	It is difficult to achieve compared to physical independence.
To make changes at the physical level we generally do not require changes at the application program level.	To make changes at the logical level, we need to make changes at the application level.
It tells about the internal schema.	It tells about the conceptual schema.
There may or may not be a need for changes to be made at the internal level to improve the structure.	Whenever the logical structure of the database has to be changed, the changes made at the logical level are important.
Example- change in compression technology, hashing algorithm, storage device etc.	Example – adding/modifying or deleting a new attribute.

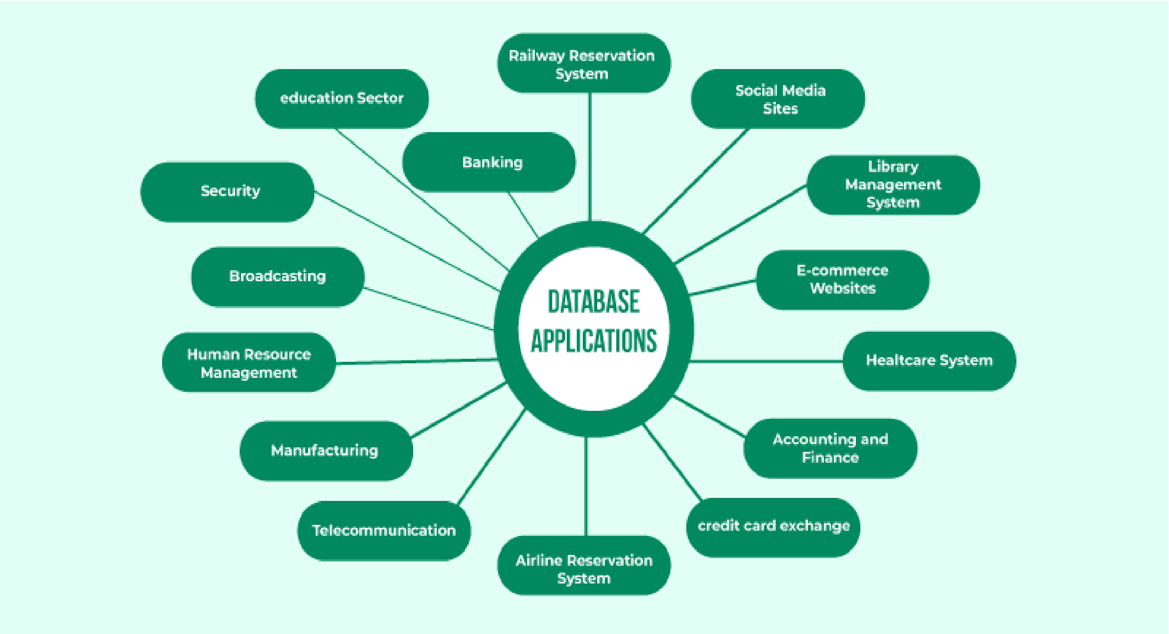
Application of DBMS

The efficient and safe management, saving and retrieval of data is made possible by the Database Management Systems. They provide strong solutions for the data management demands and are the foundation of the numerous applications used in a variety of the sectors. Recognizing the uses of DBMSs aids in understanding their significance in contemporary company operations and technology.

What is DBMS?

A Database Management System (DBMS) makes easier to create, maintain and work with the databases. It acts as the channel between end users and the database enabling the functions including administration, retrieval, updating and storing of data. By structuring data into the organized formats and controlling concurrent access the database management systems (DBMSs) contribute to the efficient handling, security and integrity of data.

There are different fields where a database management system is utilized. Following are a few applications that utilize the information base administration framework.



Applications of DBMS

DBMS applications range from banking to education, ensuring the effective management of large amounts of data. Understanding the various applications of DBMS is crucial for database professionals and students alike.

1. Railway Reservation System

In the rail route reservation framework, the information base is needed to store the record or information of ticket appointments, status of train's appearance, and flight. Additionally, if trains get late, individuals become acquainted with it through the information base update.

2. Library Management System

There are many books in the library so; it is difficult to store the record of the relative multitude of books in a register or duplicate. Along these lines, the data set administration framework (DBMS) is utilized to keep up all the data identified with the name of the book, issue date, accessibility of the book, and its writer.

3. Banking

Database the executive's framework is utilized to store the exchange data of the client in the information base.

4. Education Sector

Presently, assessments are led online by numerous schools and colleges. They deal with all assessment information through the data set administration framework (DBMS). In spite of that understudy's enlistments subtleties, grades, courses, expense, participation, results, and so forth all the data is put away in the information base.

5. Credit card exchanges

The database Management framework is utilized for buying on charge cards and age of month to month proclamations.

6. Social Media Sites

We all utilization of online media sites to associate with companions and to impart our perspectives to the world. Every day, many people group pursue these online media accounts like Pinterest, Facebook, Twitter, and Google in addition to. By the utilization of the data set administration framework, all the data of clients are put away in the information base and, we become ready to interface with others.

7. Broadcast communications

Without DBMS any media transmission organization can't think. The Database the executive's framework is fundamental for these organizations to store the call subtleties and month to month postpaid bills in the information base.

8. Accounting and Finance

The information base administration framework is utilized for putting away data about deals, holding and acquisition of monetary instruments, for example, stocks and bonds in a data set.

9. E-Commerce Websites

These days, web-based shopping has become a major pattern. Nobody needs to visit the shop and burn through their time. Everybody needs to shop through web based shopping sites, (for example, Amazon, Flipkart, Snapdeal) from home. So all the items are sold and added uniquely with the assistance of the information base administration framework (DBMS). Receipt charges, installments, buy data these are finished with the assistance of DBMS.

10. Human Resource Management

Big firms or organizations have numerous specialists or representatives working under them. They store data about worker's compensation, assessment, and work with the assistance of an information base administration framework (DBMS).

11. Manufacturing

Manufacturing organizations make various kinds of items and deal them consistently. To keep the data about their items like bills, acquisition of the item, amount, inventory network the executives, information base administration framework (DBMS) is utilized.

12. Airline Reservation System

This framework is equivalent to the railroad reservation framework. This framework additionally utilizes an information base administration framework to store the records of flight takeoff, appearance, and defer status.

13. Healthcare System

DBMS is used in healthcare to manage patient data, medical records, and billing information.

14. Security

DBMS provides security features to ensure that only authorized users have access to the data.

15. Telecommunication

Database Management Systems (DBMS) are essential to the telecommunications industry because they manage enormous volumes of data on billing, customer information, and network optimization.

In Conclusion, database management systems, or DBMS, are essential to effective, safe, and well-organized data management. They offer multiple user access, enable scalability, and guarantee integrity. Data recovery, decision support, and web-based platforms are just a few of the uses for **database management systems (DBMS)** that are essential to contemporary information systems and satisfy a wide range of business objectives.