विचार संगम

भारतीय दर्शन, विज्ञान, साहित्य-कला और तकनीकी के विशेष संदर्भ में

सम्पादक
डॉ. दिनेश चन्द्र शर्मा

विचार संगम

भारतीय दर्शन, विज्ञान, साहित्य-कला और तकनीकी के विशेष संदर्भ में

सम्पादक :

डॉ. दिनेश चन्द्र शर्मा

Ph.D., D.Lit. (Sub.)

निदेशक, कॉम्पिटिशन प्लस ग्रुप ऑफ इन्स्टीट्यूशन,

अजमेर

Books n Books

51/1509, आचमन, वेदविहार,

लोहाखान, अजमेर-305001

प्रकाशक :

पुष्पा शर्मा

Books n Books

51/1509, आचमन, वेदविहार,

लोहाखान, अजमेर-305001

ई-मेल : bnbajmer@gmail.com

संस्करण

प्रथम 2025

ISBN : 978-81-993854-0-5

मूल्य - 400/-

रचना - विचार संगम-भारतीय दर्शन, विज्ञान, साहित्य-कला और तकनीकी के विशेष संदर्भ में

सम्पादक - डॉ. दिनेश चन्द्र शर्मा

मुद्रक - श्रीनिवास प्रिन्टोग्राफिक्स, अजमेर

10

DBMS (DATABASE MANAGEMENT SYSTEM) INTRODUCTION

Shahin Parveen
Asst. Professor (Department of Science and Technology)
Hukumchand Noble Institute of Science and Technology, Ajmer

Introduction of DBMS (Database Management System) :

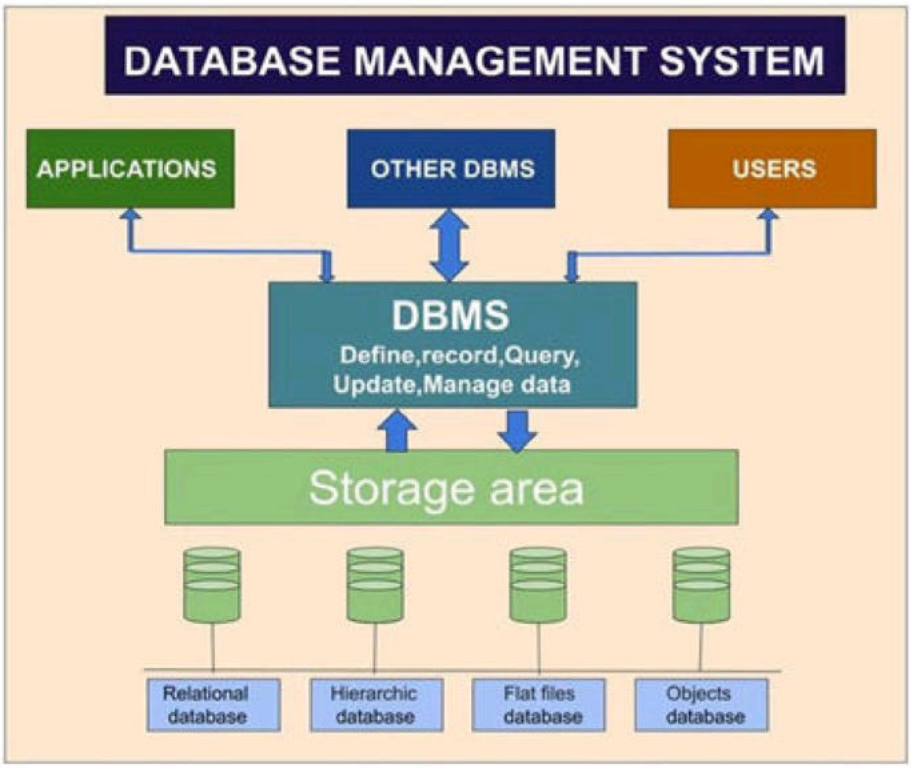
DBMS is a software system that manages, stores, and retrieves data efficiently in a structured format.

* It allows users to create, update, and query databases efficiently.

* Ensures data integrity, consistency, and security across multiple users and applications.

* Reduces data redundancy and inconsistency through centralized control.

* Supports concurrent data access, transaction management, and automatic backups



DBMS acts as a bridge between a central database and multiple

clients-including apps and users. It uses APIs to handle data requests, enabling apps and users to interact with the database securely and efficiently without directly accessing the data.

Problems with Traditional File-Based Systems

Before the introduction of modern 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

* Data Redundancy: Duplicate entries across files
* Inconsistency: Conflicting or outdated information
* Difficult Access: Manual file search required
* Poor Security: No control over data access
* Single-User Access: No support for collaboration
* No Backup/Recovery: Data loss was often permanent

A university file-based system storing data in separate files (e.g., Academics, Results, Hostels) often faced these problems.

History of DBMS :

The first database management systems (DBMS) were created to handle complex data for businesses in the 1960s. These systems included Charles Bachman's Integrated Data Store (IDS) and IBM's Information Management System (IMS). Databases were first organized into tree-like structures using hierarchical and network models.

Edgar F. Codd popularized the relational model in the 1970s, transforming database management systems (DBMS) with the concept of arranging data in tables, or relations and utilizing SQL for queries. As a result, contemporary DBMS systems like Oracle and MySQL were established. These systems are still developing today, incorporating newer technologies like NoSQL databases to handle unstructured data.

What is DBMS?

A Database Management System (DBMS) is a Software system designed to handle, store, retrieve, and work with data in databases. It ensures effective data management by serving as an interface between databases and end users. A DBMS gives users/developers the ability to create, edit, and remove databases as well as organize the environment in which data is kept in tables, records, and fields.

DBMS supports multiple operations, including data storage,

retrieval, updating, and database management. Usually, DBMS communicates with the database using Structured Query Language (SQL). A database management system (DBMS) is primarily used to manage large amounts of data in an organized manner while maintaining data security, consistency, and integrity.

Two important characteristics of DBMSs are data independence, which guarantees that modifications to data structures do not impact application layers, and data abstraction, which makes database interactions simpler by shielding users from the complexity of data storage. Additionally, it provides concurrency control, which permits several users to access data concurrently and peacefully

Evolution of Data Storage and Processing Techniques

* **1950s and early 1960s:** Magnetic tapes were developed for data storage. Data processing tasks such as payroll were automated, with data stored on tapes. Processing of data consisted of reading data from one or more tapes and writing data to a new tape. Data could also be input from punched card decks, and output to printers.

* **Late 1960s and 1970s:** Widespread use of hard disks in the late 1960s changed the scenario for data processing greatly, since hard disks allowed direct access to data. The position of data on disk was immaterial, since any location on disk could be accessed in just tens of milliseconds. With disks, network and hierarchical databases could be created that allowed data structures such as lists and trees to be stored on disk. Programmers could construct and manipulate these data structures. A landmark paper by Codd [1970] defined the relational model and nonprocedural ways of querying data in the relational model, and relational databases were born.

* **1980s:** Although academically interesting, the relational model was not used in practice initially, because of its perceived performance disadvantages; relational databases could not match the performance of existing network and hierarchical databases. That changed with System R, a ground-breaking project at IBM Research that developed techniques for the construction of an efficient relational database system. Excellent overviews of System R are provided by Astrakhan et al. [1976] and Chamberlin et al. [1981]. The fully functional System R prototype led to IBM's first relational database product, SQL/DS. At the same time, the Ingres system was being developed at the University of California at Berkeley. It led

to a commercial product of the same name. Initial commercial relational database systems, such as IBM DB2, Oracle, Ingres, and DEC Rdb, played a major role in advancing techniques for efficient processing of declarative queries. By the early 1980s, relational databases had become competitive with network and hierarchical database systems even in the area of performance. Relational databases were so easy to use that they eventually replaced network and hierarchical databases; programmers using such databases were forced to deal with many low-level implementation details, and had to code their queries in a procedural fashion. Most importantly, they had to keep efficiency in mind when designing their programs, which involved a lot of effort. In contrast, in a relational database, almost all these low-level tasks are carried out automatically by the database, leaving the programmer free to work at a logical level.

* **Early 1990s:** The SQL language was designed primarily for decision support applications, which are query-intensive, yet the mainstay of databases in the 1980s was transaction-processing applications, which are update-intensive. Decision support and querying re-emerged as a major application area for databases. Tools for analyzing large amounts of data saw large growths in usage. Many database vendors introduced parallel database products in this period. Database vendors also began to add object-relational support to their databases.

* **1990s:** The major event of the 1990s was the explosive growth of the World Wide Web. Databases were deployed much more extensively than ever before. Database systems now had to support very high transaction-processing rates, as well as very high reliability and 24 × 7 availability (availability 24 hours a day, 7 days a week, meaning no downtime for scheduled maintenance activities). Database systems also had to support Web interfaces to data.

* **2000s:** The first half of the 2000s saw the emerging of XML and the associated query language XQuery as a new database technology. Although XML is widely used for data exchange, as well as for storing certain complex data types, relational databases still form the core of a vast majority of large-scale database applications. In this time period we have also witnessed the growth in "autonomic-computing/auto-admin" techniques for minimizing system administration effort. This period also saw a significant growth in

use of open-source database systems, particularly PostgreSQL and MySQL. The latter part of the decade has seen growth in specialized databases for data analysis, in particular column-stores, which in effect store each column of a table as a separate array, and highly parallel database systems designed for analysis of very large data sets. Several novel distributed data-storage systems have been built to handle the data management requirements of very large Web sites such as Amazon, Facebook, Google, Microsoft and Yahoo!, and some of these are now offered as Web services that can be used by application developers.

Components of DBMS Applications

Any DBMS based applications is made up of six key components that work together to handle data effectively.

Components of DBMS Applications

1. Hardware

* Physical devices like servers, disks, input-output devices (keyboard, monitor, printer).

* Stores and processes data; interfaces between real-world inputs and digital systems.

* Examples: Personal computer hard disk, RAM, network devices used for DBMS operations.

2. Software

* Actual DBMS software like MySQL, Oracle, PostgreSQL.

* Includes the database engine, OS, network software, and application tools.

* Translates database access languages into operations.

3. Data

* Raw facts stored in structured or unstructured formats.
* Operational Data: Actual user data (e.g., name, age).
* Metadata: Data about data (e.g., storage time, size, data type).
* Core reason DBMS exists-to manage and store data efficiently.

4. Procedures

* Instructions and rules for using DBMS effectively.
* Covers setup, login/logout, data validation, backup, access control, and report generation.
* Helps ensure consistent and secure use of the system.

5. Database Access Language

* Used to interact with the database (create, read, update, delete data).
* Examples: SQL, MyAccess, Oracle PL/SQL.
* DDL (Data Definition Language) - CREATE, ALTER, DROP
* DML (Data Manipulation Language) - INSERT, UPDATE, DELETE

6. People

* Users interacting with DBMS at different levels:
* Database Administrators (DBA) - Manage security, performance, user access.
* Developers - Build applications using the database.
* End Users - Use applications to access the database (e.g., students, employees).

Difference between File System and DBMS :

A file system and a DBMS are two kinds of data management systems that are used in different capacities and possess different characteristics. A File System is a way of organizing files into groups and folders and then storing them in a storage device. It provides the media that stores data as well as enables users to perform procedures such as reading, writing, and even erasure.

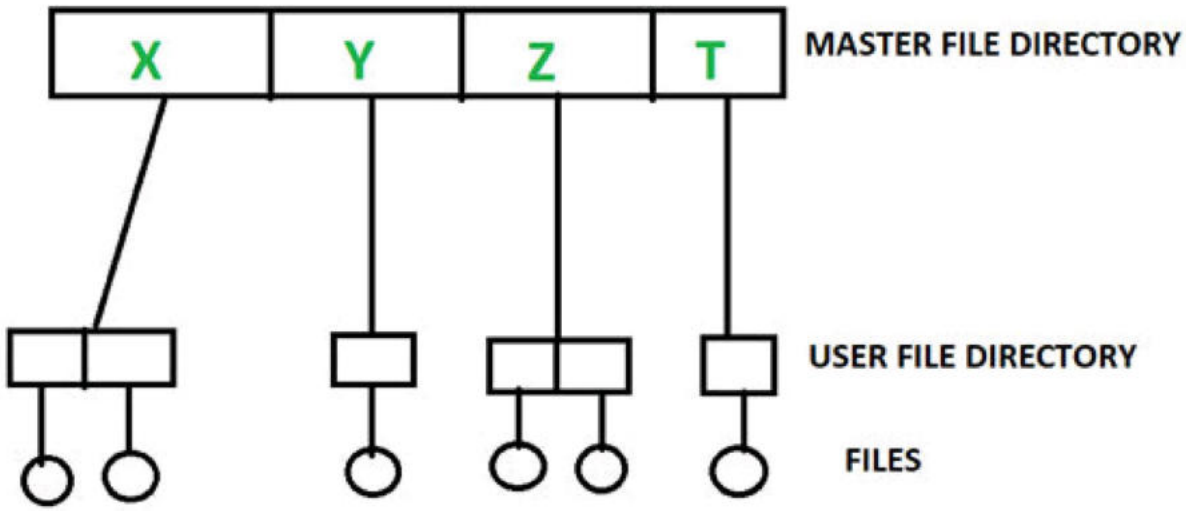
On the other hand, DBMS is a more elaborate software application that is solely charged with the responsibility of managing large amounts of structured data. It provides functionalities such as query, index, transaction, as well as data integrity. Although the file

system serves well for the purpose of data storage for applications where data is to be stored simply and does not require any great organization, DBMS is more appropriate for applications where data needs to be stored and optimized for organizational and structural needs, security, etc.

File System

The file system is basically a way of arranging the files in a storage medium like a hard disk. The file system organizes the files and helps in the retrieval of files when they are required. File systems consist of different files which are grouped into directories. The directories further contain other folders and files. The file system performs basic operations like management, file naming, giving access rules, etc.

Example: NTFS(New Technology File System) , EXT(Extended File System).

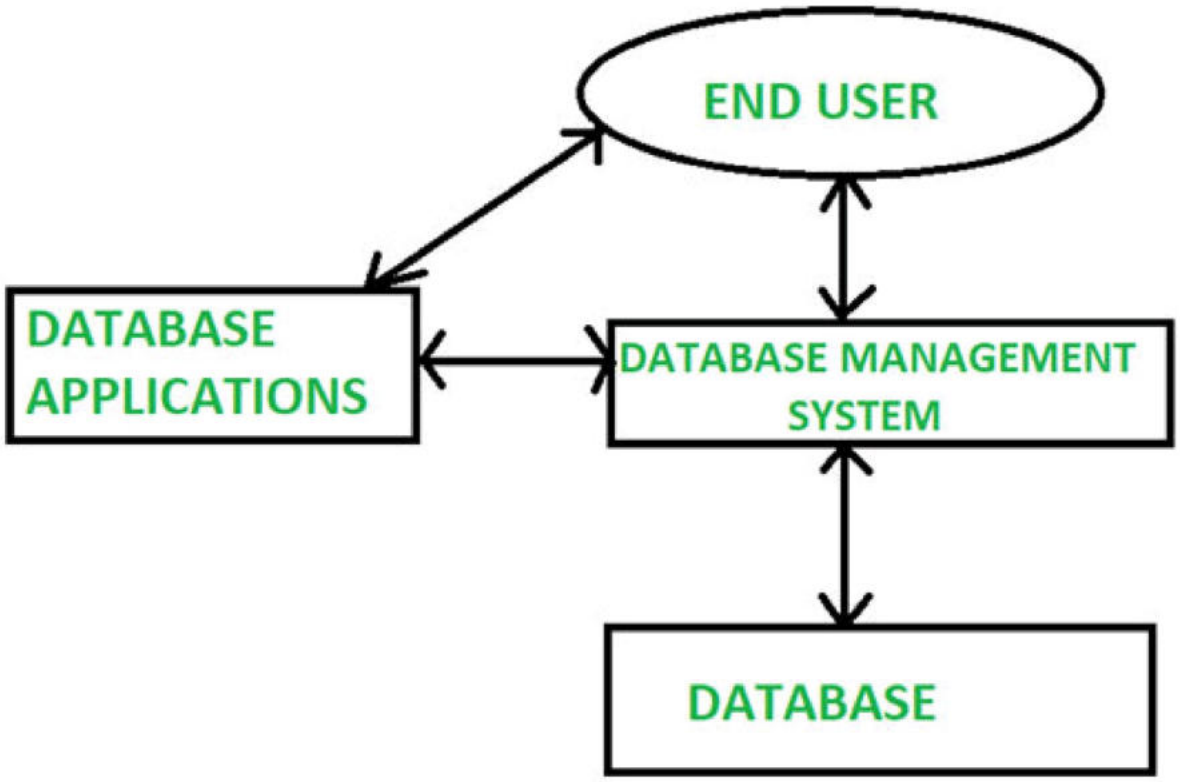


File System

DBMS (Database Management System) :

Database Management System is basically software that manages the collection of related data. It is used for storing data and retrieving the data effectively when it is needed. It also provides proper security measures for protecting the data from unauthorized access. In Database Management System the data can be fetched by SQL queries and relational algebra. It also provides mechanisms for data recovery and data backup.

Example: Oracle, MySQL, MS SQL server.



DBMS

Difference Between File System and DBMS :

Basics	**File System**	**DBMS**
Structure	The file system is a way of arranging the files in a storage medium within a computer.	DBMS is software for managing the database.
Data Redundancy	Redundant data can be present in a file system.	In DBMS there is no redundant data.
Backup and Recovery	It doesn't provide Inbuilt mechanism for backup and recovery of data if it is lost.	It provides in house tools for backup and recovery of data even if it is lost.
Query processing	There is no efficient query processing in the file system.	Efficient query processing is there in DBMS.
Consistency	There is less data consistency in the file system.	There is more data consistency because of the process of normalization .

Complexity	It is less complex as compared to DBMS.	It has more complexity in handling as compared to the file system.
Security Constraints	File systems provide less security in comparison to DBMS.	DBMS has more security mechanisms as compared to file systems.
Cost	It is less expensive than DBMS.	It has a comparatively higher cost than a file system.
Data Independence	There is no data independence.	In DBMS data independence exists, mainly of two types: 1) Logical Data Independence . 2)Physical Data Independence.
User Access	Only one user can access data at a time.	Multiple users can access data at a time.
Meaning	The users are not required to write procedures.	The user has to write procedures for managing databases
Sharing	Data is distributed in many files. So, it is not easy to share data.	Due to centralized nature data sharing is easy
Data Abstraction	It give details of storage and representation of data	It hides the internal details of Database
Integrity Constraints	Integrity Constraints are difficult to implement	Integrity constraints are easy to implement
Attribute s	To access data in a file, user requires attributes such as file name, file location.	No such attributes are required.
Example	Cobol , C++	Oracle , SQL Server

A file system manages storage, while a DBMS provides efficient data management. To learn more, the GATE CS Self-Paced Course covers these differences thoroughly.

The main difference between a file system and a DBMS (Database Management System) is the way they organize and manage data.

1. File systems are used to manage files and directories, and provide basic operations for creating, deleting, renaming, and accessing files. They typically store data in a hierarchical structure, where files are organized in directories and subdirectories. File systems are simple and efficient, but they lack the ability to manage complex data relationships and ensure data consistency.

2. On the other hand, DBMS is a software system designed to manage large amounts of structured data, and provide advanced operations for storing, retrieving, and manipulating data. DBMS provides a centralized and organized way of storing data, which can be accessed and modified by multiple users or applications. DBMS offers advanced features like data validation, indexing , transactions, concurrency control , and backup and recovery mechanisms. DBMS ensures data consistency, accuracy, and integrity by enforcing data constraints, such as primary keys, foreign keys, and data types.

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)

* It organizes data into tables (relations) composed of rows and columns.

* 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

* They 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)

* It integrates object-oriented programming concepts into the database environment, allowing data to be stored as objects.

* Supports complex data types and relationships, making it ideal for applications requiring advanced data modeling and real-world simulations.

Examples: ObjectDB, db4o.

4. Hierarchical Database

* Organizes data in a tree-like structure, where each record (node) has a single parent and have multiple children.

* This model is similar to a file system with folders and subfolders.

* It is efficient for storing data with a clear hierarchy, such as organizational charts or file directories.

* Navigation is fast and predictable due to the fixed structure.

* It lacks flexibility and difficult to restructure or handle complex many-to-many relationships.

Example: IBM Information Management System (IMS).

5. Network Database

* It uses a graph-like model to allow more complex relationships between entities.

* Unlike the hierarchical model, it permits each child to have multiple parents, enabling many-to-many relationships.

* Data is represented using records and sets, where sets define the relationships.

* It is more flexible than the hierarchical model and better suited for applications with complex data linkages.

Example: Integrated Data Store (IDS), TurboIMAGE.

6. Cloud-Based Database

* They are hosted on cloud computing platforms like AWS, Azure or Google Cloud.

* They offer on-demand scalability, high availability, automatic backups and remote accessibility.

* These databases can be relational (SQL) or non-relational (NoSQL) and are maintained by cloud service providers, reducing administrative overhead.

* They support modern application requirements, including distributed access and real-time analytics.

Example: Amazon RDS (for SQL), MongoDB Atlas (for NoSQL), Google BigQuery.