Unit 1: Database and Database Users

Syllabus: Introduction; Characteristics of the Database Approach; Actors on the Scene; Workers behind the Scene; Advantages of Using the DBMS Approach [2 Hrs]

Introduction

The known facts that can be recorded and have some implicit meaning are called data. The organized collection of logically related data is called database. A database can be paper based (manual) or computerized. A computerized database must represent the aspects of real world which is also known as miniworld or the universe of discourse. Any event that occur in real world must be reflected in the computerized database. A database has some specific purpose, intended user group and an application with which user can interact with the database.

Normal users interact with database through some application program whereas some advance users like database designer or database administrator can interact with database directly. In either case a special software is needed that helps to access database known as DBMS.

Database Management System (DBMS) is a general purpose software system that allows the users and applications to define and maintain the computerized database. Defining a database means specifying the data types, storage structure and the constraints of the data to be stored in the database. A DBMS maintains a catalog containing the information about the database itself, known as metadata (data about database).

A DBMS helps to construct the database which means to store data in computers such that it can be manipulated later. Manipulating a database means to perform different database operations such as add, retrieve, update etc. to reflect any event in the miniworld.

A telephone diary, contact list, flight schedule, list of users etc. are examples of database. Some common examples of DBMS are MS-Access, MySQL, SQL server, Oracle, MongoDB, SyBase, FoxBase, SQLite etc.

A simplified database system environment is shown in the following diagram. The end users access the database through application program. The application program generates the query and passes these query to the DBMS system. DBMS system processes the query and accesses the actual database stored physically in the computer system. There are different ways of storing physical data in the computers. The most common approach is storing data in the form of tables. This approach is known as Relational Database Management System (RDMBS) or simply database approach.

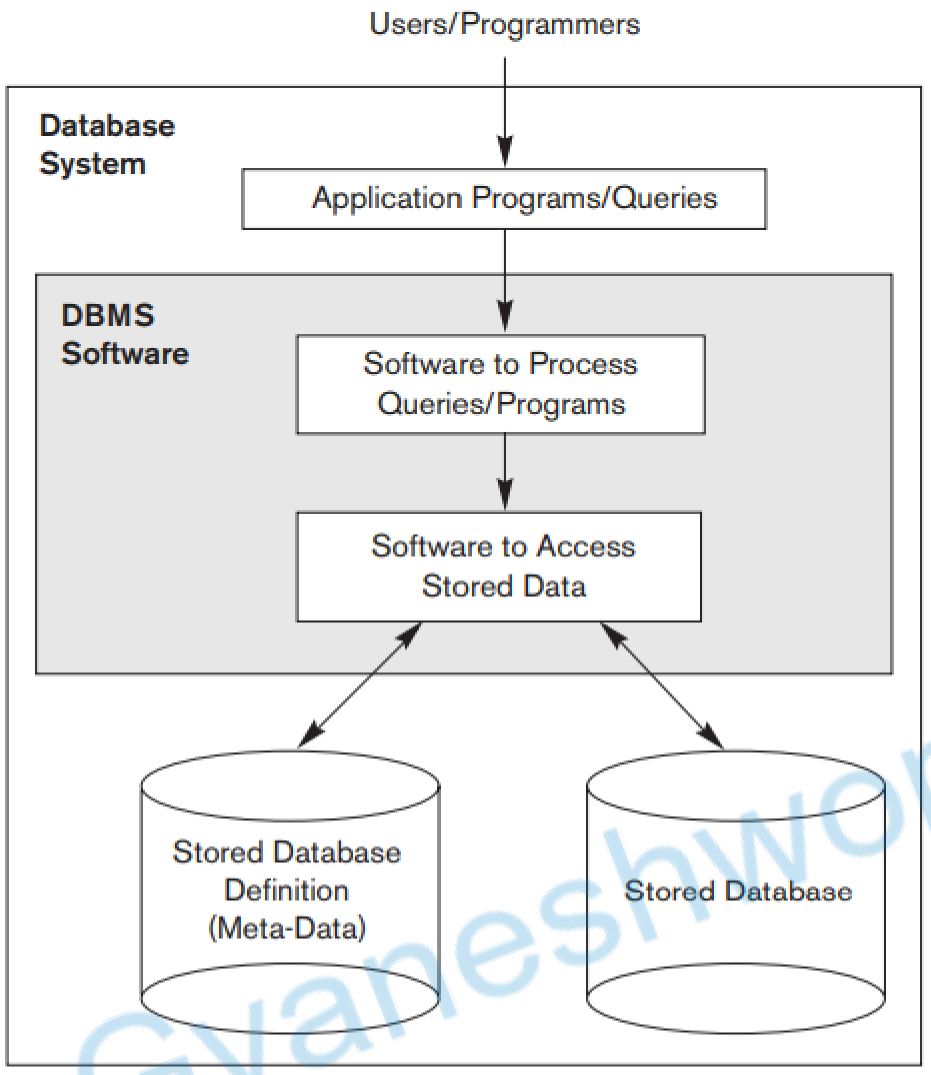
Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Fig: A simplified database system environment

Traditional File Processing System

A file processing system is a method for storing and organizing computer files and the data they contain to make it easy to find and access them. File systems may use a storage device such as a hard disk or CD-ROM and involve maintaining the physical location of the files. In this approach, we used to store information in flat files which are maintained by the file system under the operating system's control. Here, files (or filefiles) are containing records having no structured relationship among them. The file handling which we learn under C/C ++ is the example of file processing system. The Application programs written in C/C ++ like programming languages go through the file system to access these flat files. The traditional file processing approach works well for small applications that doesn’t need dynamic data processing capabilities. This approach is not adequate for the large and complex applications that change time to time.

Following are some important characteristics of traditional file processing system:

- It uses a group of flat files to store data.
- Each file is independent of one another.

- Each file contains a particular type of data.
- Files are explicitly created by using programming languages like C/C++ etc.
- Any change in database design needs hard work of reprogramming.
- The database becomes less flexible, hard to scale and has more constraints.
- Proper maintaining the database becomes difficult task.

Limitations of the file based approach

Following are the major limitations/problems associated with file based systems.

1. Difficulty in data access: In file processing system, each file is isolated and contains a specific type of data. If we need to retrieve data from a number of files, we need to write large amount of code just to handle the necessary files. This increases application development time and hence cost.

2. Data redundancy: Often the same information is stored in more than one file. Uncontrolled duplication of data is not required for several reasons, such as:

- Duplication is wasteful. It costs time and money to enter the data more than once.
- It takes up additional storage space.
- Duplication can lead to loss of data integrity.
- Database becomes less consistent.

3. Data Dependency: In the file based approach, application programs are data dependent. It means that, with the change in the physical representation (how the data is physically represented in disk) or access technique (how it is physically accessed) of data, application programs are also affected and needs modification. In other word, application programs are dependent on the how the data is physically stored and accessed.

4. Difficulty in combining data from different files: To create useful applications for the user, often data from various files must be combined. In file processing it is difficult to determine relationships between isolated data in order to meet user requirements.

5. Scalability: The application programs that has used file based approach for its database, are less scalable. Upgrading the database and application becomes very difficult.

6. Data Security. The security of data is low in file based system because, the data is maintained in the flat file(s) is easily accessible. For Example: Consider the Banking System. The Customer Transaction file has details about the total available balance of all customers. A Customer wants information about his account balance. In a file system it is difficult to give the Customer access to only his data in the· file. Thus enforcing security constraints for the entire file or for certain data items are difficult.

7. Transactional Problems. The File based system approach does not satisfy transaction properties like Atomicity, Consistency, Isolation and Durability properties commonly known as ACID properties.

8. Concurrency problems. When multiple users access the same piece of data at same interval of time then it is called as concurrency of the system. When two or more users read the data simultaneously there is no problem, however when they like to update a file simultaneously, it may lead the database to inconsistent state.

9. Poor data modeling of real world. The file based system is not able to represent the complex data and interfile relationships, thus it is unable to represent the real world data model.

Characteristics of the Database Approach

Following are the main characteristics of the database approach versus the file-processing approach (advantages of database approach over file based approach):

1. **Self-describing nature of a database system:** A fundamental characteristic of the database approach is that the database system contains not only the database itself but also a complete definition or description of the database structure and constraints. This definition is stored in the DBMS catalog, which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data. The information stored in the catalog is called meta-data, and it describes the structure of the primary database.

   A general-purpose DBMS program is not written for a specific database application. Therefore, it must refer to the catalog (metadata) to know the structure of the files in a specific database, such as the type and format of data it will access.

2. **Insulation between programs and data and data abstraction:** In traditional file processing, the structure of data files is embedded in the application programs, so any changes to the structure of a file may require changing all programs that access that file. Thus the application program is directly dependent on the storage structure of data. By contrast, in database approach the data and application logic are maintained separately, so the application program and data are well insulated from one another. This property is also known as program data independence.

   In DBMS approach, the application programs do not access data directly, rather they access data through some functions (also called transactions). If the application program has to access different set of data, then they can call the same functions with different set of arguments. In other words, in DBMS approach the application programs need not define how to access the desired data from the database. The application program simply declares what data to access, and what operation to perform on that data item. How to access that data item and how to perform the desired operation is the function of DBMS. This property is called data abstraction.

3. **Support of multiple views of the data:** A database typically has many types of users, each of whom may require a different perspective or view of the database. A view is a subset of the database or it contains virtual data that is derived from the database files but is not explicitly stored in the database. Some users may not need to be aware of whether the data they refer to is stored or derived. A multiuser DBMS whose users have a variety of distinct applications must provide facilities for defining multiple views. In other words, DBMS allows to create multiple views for different database users having different privilege.

4. **Sharing of data and multiuser transaction processing:** A multiuser DBMS, as its name implies, must allow multiple users to access the database at the same time. This is essential if data for multiple applications is to be integrated and maintained in a single database. The DBMS must include concurrency control mechanism to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct. For example, when several reservation agents try to assign a seat on an airline flight, the DBMS should ensure that each seat can be accessed by only one agent at a time for assignment to a passenger. These types of applications are generally called online transaction processing (OLTP) applications. A fundamental

role of multiuser DBMS software is to ensure that concurrent transactions operate correctly and efficiently.

Actors on the Scene

The people whose jobs involve the day-to-day use of a large database are called actors on the scene. They are the people who are directly involved in designing, constructing and manipulation the database. The important actors on the scene are:

1. **Database administrator:** In any organization where many people use the same resources, there is a need for a chief administrator to oversee and manage these resources. In a database environment, the primary resource is the database itself, and the secondary resource is the DBMS and related software. Administering these resources is the responsibility of the database administrator (DBA). The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed. The DBA is accountable for problems such as security breaches and poor system response time. In large organizations, the DBA is assisted by a staff that carries out these functions.

2. **Database designers:** Database designers are responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data. These tasks are mostly undertaken before the database is actually implemented and populated with data. It is the responsibility of database designers to communicate with all prospective database users in order to understand their requirements and to create a design that meets these requirements. In many cases, the designers are on the staff of the DBA and may be assigned other staff responsibilities after the database design is completed. Database designers typically interact with each potential group of users and develop views of the database that meet the data and processing requirements of these groups. Each view is then analyzed and integrated with the views of other user groups. The final database design must be capable of supporting the requirements of all user groups.

3. **End users:** End users are the people who access database by generating queries. They are the prime users of the database. Depending open how they access the database, there are different types of end users. Following are the types of end users according to their database access behavior and privilege.

   a. **Casual end users:** These are the users who occasionally access the database but they require different information each time. They use a sophisticated database query language basically to specify their request and are typically middle or level managers or other occasional browsers. These users learn very few facilities that they may use repeatedly from the multiple facilities provided by DBMS to access it.

   b. **Naive or parametric end users:** These are the users who basically make up a sizeable portion of database end users. The main job function revolves basically around constantly querying and updating the database for this we basically use a standard type of query known as **canned transaction**that have been programmed and tested.

These users need to learn very little about the facilities provided by the DBMS they basically have to understand the users' interfaces of the standard transaction designed and implemented for their use. The following tasks are basically performed by Naive end users:

- The person who is working in the bank will basically tell us the account balance and post-withdrawal and deposits.
- Reservation clerks for airlines, railway, hotels, and car rental companies basically check availability for a given request and make the reservation.
- Clerks who are working at receiving end for shipping companies enter the package identifies via barcodes and descriptive information through buttons to update a central database of received and in transit packages

c. **Sophisticated end users:** These users basically include engineers, scientist, business analytics and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their application to meet their complex requirement. These users try to learn most of the DBMS facilities in order to achieve their complex requirements.

d. **Standalone users:** These are those users whose job is basically to maintain personal databases by using a ready-made program package that provides easy to use menu-based or graphics-based interfaces, an example is the user of a tax package that basically stores a variety of personal financial data of tax purposes. These users become very proficient in using a specific software package.

4. **System analysts and application programmers:** System analysts determine the requirements of end users, especially naive and parametric end users, and develop specifications for standard canned transactions that meet these requirements. Application programmers implement these specifications as programs; then they test, debug, document, and maintain these canned transactions. Such analysts and programmers—commonly referred to as software developers or software engineers they should be familiar with the full range of capabilities provided by the DBMS to accomplish their tasks.

<u>Workers Behind the Scene</u>

People who are instrumental in making the database system available to end users, but typically do not use the database contents for their own purposes are called workers behind the scene. Following are the major types of workers behind the scene.

1. **DBMS system designers and implementers:** Design and implement the DBMS modules and interfaces as a software package. A DBMS is a very complex software system that consists of many components, or modules, including modules for implementing the catalog, query language processing, interface processing, accessing and buffering data, controlling concurrency, and handling data recovery and security. The DBMS must interface with other system software such as the operating system and compilers for various programming languages.

2. **Tool developers:** Design and implement tools are the software packages that facilitate database modeling and design, database system design, and improved performance. Tools are optional packages that are often purchased separately. They include packages for database design, performance monitoring, natural language or graphical interfaces, prototyping, simulation, and test data generation. In many cases, independent software vendors develop and market these tools.

3. **Operators and maintenance personnel** (system administration personnel) are responsible for the actual running and maintenance of the hardware and software environment for the database system.

Advantages of Using the DBMS Approach

In this section we discuss some additional advantages of using a DBMS and the capabilities that a good DBMS should possess. These capabilities are in addition to the four main characteristics of database approach discussed above.

1. **Reduced Redundancy:**
   In traditional file processing system, same data is stored in multiple locations. This redundancy creates a number of problems. One problem may be data integrity problem. This means a single logical update may not be sufficient. The data needs to be updated in multiple locations. This in turn leads the database in an inconsistent state. DBMS uses the centralized approach with which data integrity becomes strong and the database becomes more consistent.
2. **Restricted unauthorized access:**
   When multiple users share a large database, it is likely that most users will not be authorized to access all information in the database. For example, financial data such as salaries and bonuses is often considered confidential, and only authorized persons are allowed to access such data. In addition, some users may only be permitted to retrieve data, whereas others are allowed to retrieve and update. Hence, the type of access operation—retrieval or update—must also be controlled. Typically, users or user groups are given account numbers protected by passwords, which they can use to gain access to the database. A DBMS provides a security and authorization system, which the DBA uses to create accounts and to specify account restrictions. Then, the DBMS enforces these restrictions automatically. Notice that we can apply similar controls to the DBMS software. For example, only the DBA's staff may be allowed to use certain privileged software, such as the software for creating new accounts. Similarly, parametric users may be allowed to access the database only through the predefined apps or canned transactions developed for their use.
3. **Providing persistence storage:**
   Persistence refers to the characteristics of data that continue to exit even after the process that created it ceases or the machine it is running on is powered off. When an object or state is created and needs to be persistent, it is saved in a non-volatile storage location like hard drive. In case of database, persistence means an object should not be erased unless it is really meant to be deleted. DBMS allows objects to be stored permanently in a form that is compatible to be used by a variety of programming languages.

4. **Efficient query processing capability:**
   Database systems must provide capabilities for efficiently executing queries and updates. Because the database is typically stored on disk, the DBMS must provide specialized data structures and search techniques to speed up disk search for the desired records. Auxiliary files called indexes are often used for this purpose. Indexes are typically based on tree data structures or hash data structures that are suitably modified for disk search. In order to process the database records needed by a particular query, those records must be copied from disk to main memory. Therefore, the DBMS often has a buffering or caching module that maintains parts of the database in main memory buffers. In general, the operating system is responsible for disk-to-memory buffering. However, because data buffering is crucial to the DBMS performance, most DBMSs do their own data buffering. The query processing and optimization module of the DBMS is responsible for choosing an efficient query execution plan for each query based on the existing storage structures. The choice of which indexes to create and maintain is part of physical database design and tuning, which is one of the responsibilities of the DBA staff.
5. **Backup and recovery:**
   A DBMS must provide facilities for recovering from hardware or software failures. The backup and recovery subsystem of the DBMS is responsible for recovery. For example, if the computer system fails in the middle of a complex update transaction, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the transaction started executing. Disk backup is also necessary in case of a catastrophic disk failure
6. **Multiuser interface:**
   Because many types of users with varying levels of technical knowledge use a database, a DBMS should provide a variety of user interfaces. These include apps for mobile users, query languages for casual users, programming language interfaces for application programmers, forms and command codes for parametric users, and menu-driven interfaces and natural language interfaces for standalone users. Both forms-style interfaces and menu-driven interfaces are commonly known as graphical user interfaces (GUIs). Many specialized languages and environments exist for specifying GUIs. Capabilities for providing Web GUI interfaces to a database—or Web-enabling a database—are also quite common.
7. **Representation of complex relationship among data:**
   A database may include numerous varieties of data that are interrelated in many ways. A DBMS must have the capability to represent a variety of complex relationships among the data, to define new relationships as they arise, and to retrieve and update related data easily and efficiently.
8. **Enforcing integrity constraints:**
   Most database applications have certain integrity constraints that must hold for the data. A DBMS should provide capabilities for defining and enforcing these constraints. The simplest type of integrity constraint involves specifying a data type for each data item. For example, we may specify the constraint such that the value of the Class data item within each STUDENT record must be a one-digit integer and that the value of Name must be a string of no more than 30 alphabetic characters. A more complex type of constraint that frequently occurs involves specifying that a record in one file must be related to records

in other files, this is known as a referential integrity constraint. Another type of constraint specifies uniqueness on data item values, the email address of each student must be unique. This is known as a key or uniqueness constraint. These constraints are derived from the meaning or semantics of the data and of the miniworld it represents. It is the responsibility of the database designers to identify integrity constraints during database design.

9. **Permitting Inferencing and Actions Using Rules and Triggers**

Some database systems provide capabilities for defining deduction rules for inferencing new information from the stored database facts. Such systems are called deductive database systems. In such deductive database systems, database designers can write some stored procedure which will automatically perform some desired action depending on some trigger. In today's relational database systems, it is possible to associate triggers with tables. A trigger is a form of a rule activated by updates to the table, which results in performing some additional operations to some other tables, sending messages, and so on. More involved procedures to enforce rules are popularly called stored procedures; they become a part of the overall database definition and are invoked appropriately when certain conditions are met. More powerful functionality is provided by active database systems, which provide active rules that can automatically initiate actions when certain events and conditions occur.

10. **Additional implications of using the database approach:**

Few additional implications of using the database approach are:

a. **Potential for enforcing standards:** The database approach permits the DBA to define and enforce standards among database users in a large organization. This facilitates communication and cooperation among various departments, projects, and users within the organization. Standards can be defined for names and formats of data elements, display formats, report structures, terminology, and so on. The DBA can enforce standards in a centralized database environment more easily than in an environment where each user group has control of its own data files and software.

b. **Reduced application development time:** A prime selling feature of the database approach is that developing a new application takes very little time. Designing and implementing a large multiuser database from scratch may take more time than writing a single specialized file application. However, once a database is up and running, substantially less time is generally required to create new applications using DBMS facilities. Development time using a DBMS is estimated to be one sixth to one-fourth of that for a file system.

c. **Flexibility:** It may be necessary to change the structure of a database as requirements change. For example, in file based approach, a new user group may appear that needs information not currently in the database. As a result, it may be necessary to add a file to the database or to extend the data elements in an existing file. Modern DBMSs allow certain types of evolutionary changes to the structure of the database without affecting the stored data and the existing application programs.

d. **Availability of up-to-date information:** A DBMS makes the database available to all users. As soon as one user's update is applied to the database, all other users can immediately see this update. This availability of up-to-date information is essential for many transaction-processing applications, such as online ticket reservation systems or banking databases, and it is made possible by the concurrency control and recovery subsystems of a DBMS.

e. **Economies of scale:** DBMS approach reduces the overall cost of operation and management for an organization. As the application program and the database are decoupled, the application development time is reduced, any change needed can be implemented with less effort and time. As a result, this enables the whole organization to invest in more powerful processors, storage devices, or networking gear, rather than having each department purchase its own (lower performance) equipment.