Database Users

Knowing the database Users - Actors on the Scene , Workers behind the Scene

Database Users

Database users are the one who really use and take the benefits of database. There will be different types of users depending on their need and way of accessing the database.They can be categorized into:

1.Actors on the Scene

2.Actors behind the Scene

Actors on the Scene

1. Actors on the scene

The persons whose jobs involve the day to day use of the large database are called actors on the scene.

Database Administrators (DBA):

DBA is the chief administrator to oversee and manage the resources.He is responsible for authorizing access to the database for coordinating and monitoring its uses for acquiring software and hardware resources as needed.He is responsible for breach of security poor response time etc. service packs/ hot fixes/ patches to the DBMS servers.A DBA has many responsibilities. and a good performing database is in the hands of DBA.

- **Installing and upgrading the DBMS Servers: –** DBA is responsible for installing a new DBMS server for the new projects. He is also responsible for upgrading these servers as there are new versions comes in the market or requirement. If there is any failure in upgradation of the existing servers, he should be able revert the new changes back to the older version, thus maintaining the DBMS working. He is also responsible for updating the service packs/ hot fixes/ patches to the DBMS servers.
- **Design and implementation: –** Designing the database and implementing is also DBA's responsibility. He should be able to decide proper memory management, file organizations, error handling, log maintenance etc for the database.
- **Performance tuning: –** Since database is huge and it will have lots of tables, data, constraints and indices, there will be variations in the performance from time to time. Also, because of some designing issues or data growth, the database will not work as expected. It is responsibility of the DBA to tune the database performance. He is responsible to make sure all the queries and programs works in fraction of seconds.
- **Migrate database servers: –** Sometimes, users using oracle would like to shift to SQL server or Netezza. It is the responsibility of DBA to make sure that migration happens without any failure, and there is no data loss.

- **Backup and Recovery: –** Proper backup and recovery programs needs to be developed by DBA and has to be maintained him. This is one of the main responsibilities of DBA. Data/objects should be backed up regularly so that if there is any crash, it should be recovered without much effort and data loss.
- **Security: –** DBA is responsible for creating various database users and roles, and giving them different levels of access rights.
- **Documentation: –** DBA should be properly documenting all his activities so that if he quits or any new DBA comes in, he should be able to understand the database without any effort. He should basically maintain all his installation, backup, recovery, security methods. He should keep various reports about database performance.

Database Designers:

They are responsible for identifying data to be stored in the database and for choosing appropriate structures to represent and store the data.They have to study the requirements of the various users to come up with the design that meets the requirements.This task is done before the database is implements populated with data.They also talk to the perspective users & develop the view for different users t meet their processing requirements.

End users:

End users are the people where jobs require the access to the Database for querying, updating & generating reports several categories of end users.

Casual End users:

They occasionally access the database but they may need different information each time.

They use a sophisticated database query language to specify their requests & are typically middle of high level managers or occasional browsers.

Naïve or parametric End users:

these are the users who use the existing application to interact with the database. For example, online library system, ticket booking systems, ATMs etc which has existing application and users use them to interact with the database to fulfill their requests.They job revolves around constantly querying and updating the database using standard types of queries & updates called canned transactions.

Eg. Bank tellers, reservation clerks etc.

Sophisticated End users:

They are database developers, who write SQL queries to select/insert/delete/update data. They do not use any application or programs to request the database. They directly interact with the database by means of query language like SQL. These users will be scientists, engineers, analysts

who thoroughly study SQL and DBMS to apply the concepts in their requirement. They familiarize themselves with the facilities of the DBMS, to implement their applications to meet their complex requirements.

Eg. engineers, scientists etc.

Stand alone users:

These users will have stand –alone database for their personal use. These kinds of database will have readymade database packages which will have menus and graphical interfaces.These are also sophisticated users, but they write special database application programs. They are the developers who develop the complex programs to the requirement.They maintain personal database by using ready made program packages. They are easy to use menu of graphical Interfaces.

System analysts and Application programmers:

System Analyst: System analyst is responsible for the design, structure and properties of database. All the requirements of the end users are handled by system analyst. Feasibility, economic and technical aspects of DBMS is the main concern of system analyst.They study the requirements of End users specially having users and make the specifications for canned transactions.

Application programmer: They implement the specifications as programs. i.e., test debug, document & maintain the canned transactions. Both together are called Software engineers.They are the developers who interact with the database by means of DML queries. These DML queries are written in the application programs like C, C++, JAVA, Pascal etc. These queries are converted into object code to communicate with the database. For example, writing a C program to generate the report of employees who are working in particular department will involve a query to fetch the data from database. It will include a embedded SQL query in the C Program

Workers behind the Scene
Workers behind the scene are those are typically not interested in the **database** content itself. They include the following categories: DBMS system designers and implementer .They design and implement the DBMS modules and interfaces as a software package.

DBMS system designers & Implementers

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.These persons designs and implement the DBMS modules and interfaces as a software package.DBMS has several modules like recovery, security, concurrency control etc.

Tool Developers

Tool developers design and implement tools—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.These persons design & implement the tools i.e. the software packages that facilitate database system design & use and help improve performance.

Operators and maintenance personnel

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.

Even though these categories of workers behind the scene and are instrumental in making the database system available to end users, they typically do not use the database contents for their own purposes.

Characteristics and Implications of Database Approach

Characteristics of the Database Approach

There are number of characteristics that distinguish the database approach from the much older approach of programming with files. In traditional file processing, each user defines and implements the files needed for a specific software application as part of programming the application.For example, one user, the grade reporting office, may keep files on students and their grades. Programs to print a student's transcript and to enter new grades are implemented as part of the application. A second user, the accounting office, may keep track of students' fees and their payments. Although both users are interested in data about students, each user maintains separate files— and programs to manipulate these files—because each requires some data not available from the other user's files. This redundancy in defining and storing data results in wasted storage space and in redundant efforts to maintain common up-to-date data.In the database approach, a single repository maintains data that is defined once and then accessed by various users. In file systems, each application is free to name data elements independently. In contrast, in a database, the names or labels of data are defined once, and used repeatedly by queries, transactions, and applications.

The following are the characteristics of database approach:

1. Self describing nature of the Database system:

A fundamental characteristics of 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.

The catalog is used by the DBMS software and also by database users who need information about the database structure. A general-purpose DBMS software package is not written for a specific database application. Therefore, it must refer to the catalog to know the structure of the files in a specific database, such as the type and format of data it will access. The DBMS software must work equally well with any number of database applications—for example, a university database, a banking database, or a company database—as long as the database definition is stored in the catalog.

In traditional file processing, data definition is typically part of the application programs themselves. Hence, these programs are constrained to work with only one specific database, whose structure is declared in the application programs.

For example, an application program written in C++/Java may have struct or class declarations, to define its files. Whereas file-processing software can access only specific databases, DBMS software can access diverse databases by extracting the database definitions from the catalog and

using these definitions. The DBMS catalog will store the definitions of all the files shown. Figure 1.3 shows some sample entries in a database catalog. These definitions are specified by the database designer prior to creating the actual database and are stored in the catalog. Whenever a request is made to access, say, the Name of a STUDENT record, the DBMS software refers to the catalog to determine the structure of the STUDENT file and the position and size of the Name data item within a STUDENT record. By contrast, in a typical file-processing application, the file structure and, in the extreme case, the exact location of Name within a STUDENT record are already coded within each program that accesses this data item.

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

Summary:

The database system contains not only the database itself but also a complete definition or description of the database structure and constraints which is stored in system catalog called metadata.

2. Insulation between programs and data, 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. By contrast, DBMS access programs do not require such changes in most cases. The structure of

data files is stored in the DBMS catalog separately from the access programs. We call this property **program-data independence.**

For example, a file access program may be written in such a way that it can access only STUDENT records of the structure shown in Figure 1.4. If we want to add another piece of data to each STUDENT record, say the Birth_date , such a program will no longer work and must be changed. By contrast, in a DBMS environment, we only need to change the description of STUDENT records in the catalog (Figure 1.3) to reflect the inclusion of the new data item Birth_date ; no programs are changed.

The next time a DBMS program refers to the catalog, the new structure of STUDENT records will be accessed and used. In some types of database systems, such as object-oriented and object-relational systems, users can define operations on data as part of the database definitions. An operation (also called a function or method) is specified in two parts. The interface (or signature) of an operation includes the operation name and the data types of its arguments (or parameters). The implementation (or method) of operation is specified separately and can be changed without affecting the interface. User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented. This may be termed **program-operation independence.**

The characteristic that allows **program-data independence and program-operation independence is called data abstraction.** A DBMS provides users with a conceptual representation of data that does not include many of the details of how the data is stored or how the operations are implemented. Informally, a data model is a type of data abstraction that is used to provide this conceptual representation. The data model uses logical concepts, such as objects, their properties, and their interrelationships, that may be easier for most users to understand than computer storage concepts. Hence, the data model hides storage and implementation details that are not of interest to most database users.

For example, The internal implementation of a file may be defined by its record length—the number of characters (bytes) in each record—and each data item may be specified by its starting byte within a record and its length in bytes. The STUDENT record would thus be represented as shown in Figure 1.4. But a typical database user is not concerned with the location of each data item within a record or its length; rather, the user is concerned that when a reference is made to Name of STUDENT , the correct value is returned. Many other details of file storage organization—such as the access paths specified on a file—can be hidden database users by the DBMS;

In the database approach, the detailed structure and organization of each file are stored in the catalog. Database users and application programs refer to the conceptual representation of the files, and the DBMS extracts the details of file storage from the catalog when these are needed by the DBMS file access modules. Many data models can be used to provide this data abstraction to database users. A major part of this book is devoted to presenting various data models and the concepts they use to abstract the representation of data. In object-oriented and object-relational

databases, the abstraction process includes not only the data structure but also the operations on the data. These operations provide an abstraction of miniworld activities commonly understood by the users.

For example, an operation CALCULATE_GPA can be applied to a STUDENT object to calculate the grade point average. Such operations can be invoked by the user queries or application programs without having to know the details of how the operations are implemented. In that sense, an abstraction of the miniworld activity is made available to the user as an abstract operation.

Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
Student_number	31	4
Class	35	1
Major	36	4

Figure 1.4
Internal storage format for a STUDENT record, based on the database catalog in Figure 1.3.

Summary:

DBMS Access programs do not require undergoing changes when the structure of the data files is changed because the structure of data file is stored separately in DBMS catalog separately. This property is called program data Independence.

In object oriented database users can define operations on data as part of database definitions. The operation has the interface and implementation. The implementation is specified separately and can be changed without affecting the interface. User programs can operate on the data by invoking these operations through the interface regardless of how the operations are implemented. This property is called program operation independence.

These characteristics of program data independence and program operation independence are called data abstraction.

3. Support of multiple views of data:

A database typically has many users, each of whom may require a different perspective or view of the database. A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored. Some users may not need to be aware of whether the data they refer to is stored or derived. A multi-user DBMS whose users have a variety of distinct applications must provide facilities for defining multiple views. For example, one user of the database may be interested only in accessing and printing the transcript of each student; the view for this user is shown in Figure 1.5(a). A second user, who is interested only in checking that students have taken all the prerequisites of each course for which they register, may require the view shown in Figure 1.5(b).

(a)

TRANSCRIPT

Student_name	Student_transcript				
	Course_number	Grade	Semester	Year	Section_id
Smith	CS1310	C	Fall	08	119
	MATH2410	B	Fall	08	112
Brown	MATH2410	A	Fall	07	85
	CS1310	A	Fall	07	92
	CS3320	B	Spring	08	102
	CS3380	A	Fall	08	135

(b)

COURSE_PREREQUISITES

Course_name	Course_number	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data Structures	CS3320	CS1310

Figure 1.5
Two views derived from the database in Figure 1.2. (a) The TRANSCRIPT view. (b) The COURSE_PREREQUISITES view.

Summary:

A database has many users and each of them may require a different perspective or view of database.

View is the subset of the database or it is containing the virtual data derived from the database files but is not explicitly stored.

4. Sharing of Data & 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 software 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.

The concept of a transaction has become central to many database applications. A transaction is an executing program or process that includes one or more database accesses, such as reading or updating of database records. Each transaction is supposed to execute a logically correct database access if executed in its entirety without interference from other transactions.

The DBMS must enforce several transaction properties. The **isolation property** ensures that each transaction appears to execute in isolation from other transactions, even though hundreds of transactions may be executing concurrently. The **atomicity property** ensures that either all the database operations in a transaction are executed or none are. The preceding characteristics are important in distinguishing a DBMS from traditional file-processing software.

Summary:

A multiuser DBMS 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 software 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.

The implications of using the database approach that can benefit most organizations are:

1. Potential for Enforcing Standards

Database applications are meant to reduce the efforts of data management for any organization by providing well organized and managed data. Such applications will help in faster data processing, efficient reporting system and quicker decision making.The database approach permits the DBA to define and enforce standards among database users in a large organization. **Standards** can be defined for names and formats of data elements, display formats, report structures and so on.

When data is shared with the support of DBMS in secure environment it will help **DBA** and Developers to define and implement the standards for creation of data elements, defining and storing data and presentation of data to users. Well defined and well adopted standards make the use of data easier.

As DBA it will be easier for you to create database elements like tables, views, procedures, rules, fieldnames and constraints. Any new employee who replaces an outgoing DBA will not have to struggle in understanding the data items created by his predecessor. Standards also include how the data values will be displayed to users, what will be the structure and components of user interface or screens, what will be a generic format of any report created from the database. For example:

· Date data must be displayed in ***dd-mm-yyyy*** format

· Names will be displayed as last name, first name and middle name

· Currency value has to be displayed up to 2 figures after decimal.

2. Reduced Application Development Time

Database approach allows developers to incrementally development the application. Beginning with core application accepted and approved by users can be enhanced by adding layers of functionality and complexity demanded by the user who is now well aware of his needs, scope and capability of application. Since the previous versions are well designed, adding new features will be done in lesser time.The main advantage of database approach is that developing a new application takes a very little time. Once a database is up and running, less time is required to create new applications using DBMS facilities.

3. Flexibility to alter the data structures

DBMS allows certain types of changes to the structure of the database without affecting the stored data and the existing application programs.With time and use of data there may be need to store data in a different data structures. For example earlier the amount of data was thought to be fit to store in form of a binary search tree. But with increase in volume of data it is better to store it in an m-way tree for better search and insertion. Database approach allows this freedom to alter the data structure with changing requirements of the application or organization.

4. Availability of up-to-date Information

A DBMS makes the database available to all users. When one user updates a database, all other users can immediately see this update. This availability of up-to-date information is essential for reservation systems, banking databases etc. Data residing at one location and available at different locations through networked systems is very essential. Database approach fulfills this need due to its centralized repository feature. Moreover the data consistency and real time update implies that the data is readily and quickly available for users going for on-line reservation like airlines and transactions like banking or online shopping.

5. Economy of scalability

With database approach the organizations can cutoff expenses by minimizing resources and employees. Data duplicity is considerably reduced so no need to retain copies of data files in multiple computer systems. Data stored in centralized database server will be available to all the users.A person or team devoted for database administration can take care of the information needs across the organization instead of departmental management of localized data. If the data storage and management needs increase with time, instead of replacing all the computing resources across organization only the servers can be upgraded to match the increase in volume of data.Wasteful overlap of resources and personnel can be avoided by consolidating data and applications across departments.

Advantages of DBMS,When not to use DBMS?

Advantages of DBMS

The following are the advantages of using DBMS

Controlling redundancy:

In DBMS approach the data required for different users can be stored in a centralized manner without duplicating the data multiple times.When the data is stored redundantly several problems like the following arise,

- Duplication of effect as several data has to be entered many times
- Wastage of storage space as the same data is stored many times

- Data may become inconsistent as the updating requires to be done separately on all files.

Restricting unauthorized access:

As number of users increases data transferring or data sharing rate also increases thus increasing the risk of data security. It is widely used in corporation world where companies invest money, time and effort in large amount to ensure data is secure and is used properly. A Database Management System (DBMS) provide a better platform for data privacy and security policies thus, helping companies to improve Data Security.When multiple users share the database it is required that the users be authorized and given certain operation permissions. This is taken care by the authorization and security subsystem, where the DBA creates accounts and specifies the account restrictions.

Providing persistent storage for program objects and data structures:

Database provides persistent storage for program objects and data structures. This is one of the main reasons for the emergence of object oriented database. The data structures provided by DBMS are compatible with the programming language data structures.

Permitting inferencing & actions using rules:

Some data bases provide capabilities for defining deduction rules for inferencing new information from the stored database facts. Such systems are called deductive database systems. As the mini-world rules changes the declared deduction rules can be changed than to recode the procedural programs. Active database systems provide the active rules that can automatically initiate actions.

Representing complex relationships among data:

DBMS must have the capability to represent a variety of complex relationships among the data as well as to retrieve and update related data easily & efficiently.

Providing multiple user interfaces:

Since person with varying levels of technical knowledge use the database DBMS should provide a variety of user interface. They include query language for naïve users menu & natural language interfaces for stand alone users.

Enforcing integrity constraints:

DBMS must provide capabilities for defining and enforcing the constraints (rules). Constraints include specifying the data type, uniqueness of data item etc.

Providing backup & recovery:

DBMS must provide facilities for recovering from hardware & software failures. The back up and recovery subsystem is responsible for recovery.

Minimized Data Inconsistency:
Data inconsistency occurs between files when different versions of the same data appear in different places.If a database is properly designed then Data inconsistency can be greatly reduced hence minimizing data inconsistency.

Better data integration:
Due to Database Management System we have an access to well managed and synchronized form of data thus it makes data handling very easy and gives integrated view of how a particular organization is working and also helps to keep a track on how one segment of the company affects other segment.

Faster data Access:
The Data base management system (DBMS) helps to produce quick answers to database queries thus making data accessing faster and more accurate. For example, to read or update the data.

Better decision making:
Due to DBMS now we have Better managed data and Improved data accessing because of which we can generate better quality information hence on this basis better decisions can be made.
Better Data quality improves accuracy, validity and time it takes to read data.
DBMS does not guarantee data quality, it provides a framework to make it is easy to improve data quality .

Increased end-user productivity:

The data which is available with the help of combination of tools which transform data into useful information, helps end user to make quick, informative and better decisions that can make difference between success and failure in the global economy.

Simple:
Data base management system (DBMS) gives simple and clear logical view of data. Many operations like insertion, deletion or creation of file or data are easy to implement.

Disadvantages of DBMS
Although the database system provides many advantages these database systems do carry considerable disadvantages.

Increased costs

One of the disadvantages of DBMS is database systems require sophisticated hardware, software, and highly skilled personnel. The cost of maintaining these requisites and managing a database system can be substantial. Training, licensing, and regulation compliance costs are often unheeded when database systems are employed.

Management intricacy

Database systems interface with many different technologies and have a significant impact on a company's resources and culture. The changes introduced by the adoption of a database system must be properly managed to ensure that they help advance the company's objectives. Given the fact that database systems hold crucial company data that are accessed from multiple sources, security issues must be assessed constantly.

Maintaining currency

To maximize the efficiency of the database system, you must keep your system current. Therefore, you must perform frequent updates, apply the latest patches, and security measures to all components.As database technology advances rapidly, personnel training costs tend to be significant. Given the heavy investment in technology and personnel training, companies might be reluctant to change database vendors.As a consequence, vendors are less likely to offer pricing point advantages to existing customers, and those customers might be limited in their choice of database system components.

Frequent upgrade/replacement cycles

DBMS vendors frequently upgrade their products by adding new functionality. Such new features often come bundled in new upgrade versions of the software. Some of these versions require hardware upgrades. Not only do the upgrades themselves cost money, but it also costs money to train database users and administrators to properly use and manage the new features.

When not to use DBMS?
In spite of the advantages of using a DBMS, there are a few situations in which a DBMS may involve unnecessary overhead costs that would not be incurred in traditional file processing.The overhead costs of using a DBMS are due to the following:

- High initial investment in hardware, software, and training
- The generality that a DBMS provides for defining and processing data
- Overhead for providing security, concurrency control, recovery, and integrity functions

Therefore, it may be more desirable to use regular files under the following circumstances:

- Simple, well-defined database applications that are not expected to change at all.

- Stringent, real-time requirements for some application programs that may not be met because of DBMS overhead . Embedded systems with limited storage capacity, where a general-purpose DBMS would not fit.
- No multiple-user access to data

Certain industries and applications have chosen not to use general-purpose DBMSs. For example, many computer-aided design (CAD) tools used by mechanical and civil engineers have proprietary file and data management software that is geared for the internal manipulations of drawings and 3D objects. Similarly, communication and switching systems designed by companies like AT&T were early manifestations of database software that was made to run very fast with hierarchically organized data for quick access and routing of calls. Similarly, GIS implementations often implement their own data organization schemes for efficiently implementing functions related to processing maps, physical contours, lines, polygons, and so on. General-purpose DBMSs are inadequate for their purpose.

Database System Concepts and Architecture

Data Models

Data models

DBMS allows a user to specify the data to be stored in terms of a data model. A **data model** is a collection of higher level data description constraints that hides lower level storage details. It is the collection of concepts that can be used to describe the structure of a database.

Data models are of three types:

1. Object based / High-level data models

2. Record based / Representational data models

3. Physical data model / low-level data models

1) High Level or Conceptual Data Model:

It provides the concepts that are close to the way many users perceive the data. Conceptual Data Model uses the concepts like entity, attributes, relationship.

- **Entity:** It is the real world object.
- **Attribute:** property of the entity.
- **Relationship:** Interaction between entities.

2) **Representational or Implementation Data Model:**

It provides the concepts understood by the end users and not far from the way data is organized on the computer. It includes the relational model, hierarchical and network model.

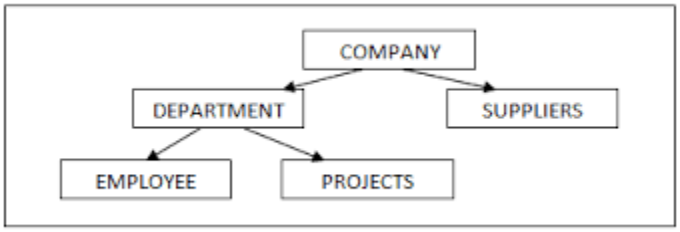
There are 3 types of record based data model. They are:
i. Hierarchical data model.

ii. Network data model.

iii. Relational data model.

i. Hierarchical data model

The data is sorted hierarchically, using a downward tree. This model uses pointers to navigate between stored data. It was the first DBMS model.



ii. Network data model

Like the hierarchical model, this model uses pointers toward stored data. However, it does not necessarily use a downward tree structure.

iii. Relational data model (**RDBMS**, *Relational database management system*)

The data is stored in two-dimensional tables (rows and columns). The data is manipulated based on the relational theory of mathematics.

Roll_No.	Name	Department
101	Steive	Comp. Sci.
265	Jhoson	Finance
505	Margret	Biology
325	Jenny	Social Sci.
256	Davis	Comp. Sci.
453	Sheryl	Biology
365	Emma	Maths

Student Relation in Relational Model

3) Low Level or Physical Data Model:

It provides the concepts that describe the details of how data is stored in the computer. Physical Data Model describes the storage of data in the computer by representing information such as record formats, record orderings and access path. Concepts provided by these models are generally meant for computer specialists, not for typical end users.

Schema , Instance , Database State -Definitions

Database Schema: It is the description of the database specified during database design and is not expected to change frequently.

Schema Diagram: The schema displayed diagrammatically is called schema diagram.

Schema Construct: Each object in the schema is called schema construct.

E.g.) of schema diagram

Student

Name	Id	phone

(Student & course are schema constraints)

Course

Cid	Cname	dept

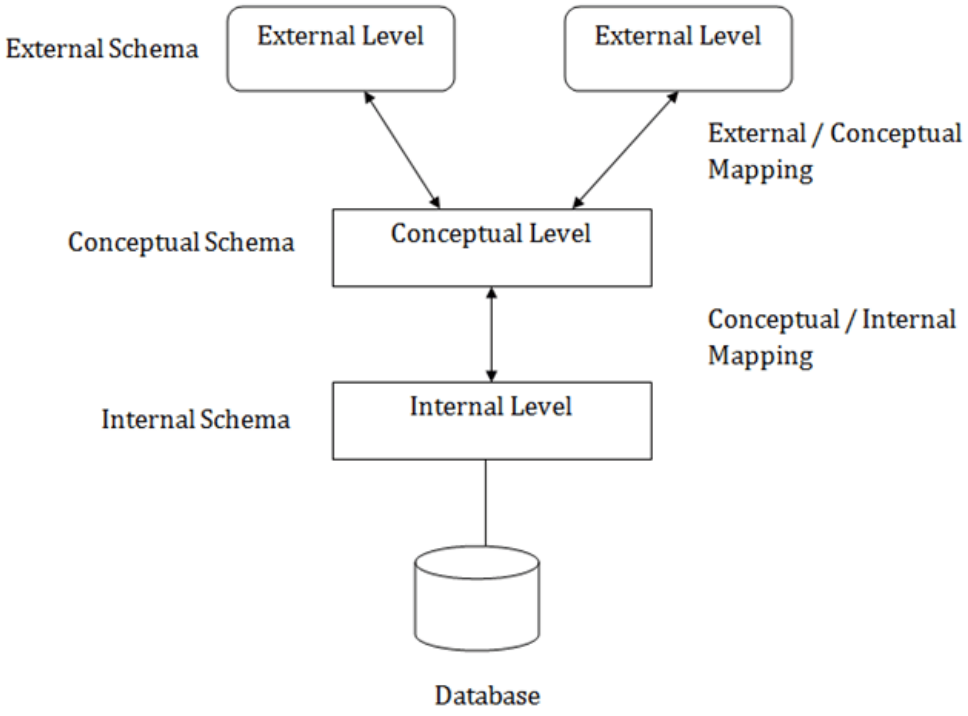
Database State or Snapshot:

The data in the database at a particular moment of time is called database state or snapshot. Also called current set of occurrences or instances in the database.

Meta Data: The DBMS stores the description of the schema constructs and constraints called meta data in **DBMS catalog**.

Schema is called intension and database state is called the extension.

Three Schema Architecture
The architecture is defined at three levels as follows:



Internal level:

- This level has an internal schema.
- It describes the physical storage structure of the database
- The internal schema uses a physical data model and describes the complete details of data storage and access path of the database.

Conceptual Level:

- This level has a conceptual schema, which describes the structure of the whole database for a community of users.
- It hides the details of physical storage structure and concentrates on describing entities, data types, relationships, constraints and user operations.

External or View Level:

- It includes the external schemas or user views.
- Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

Mappings:

The three schemas are only descriptions of data and the actual data exists only at physical level.

Each user group refers to only the external schema and the DBMS must transfer the request against the conceptual schema and then to the internal schema for processing over the stored database.

If the request is the database retrieval, the data extracted from the stored database should be reformatted to the external view of user. This process of transferring requests and results between levels is called mapping.

Data Independence:

It is the capacity to change the schema at one level of the database system without having to change the schema at the next higher level.

1. **Logical Data Independence:**

It is the capacity to change the conceptual schema without having to change external schemas or application programs.

2. **Physical Data Independence**:

It is the capacity to change the internal schema without to change the conceptual schemas.

Database Languages and Interfaces

DBMS Languages

A DBMS must provide appropriate languages and interfaces for each category of users to express database queries and updates. Database Languages are used to create and maintain database on computer. There are large numbers of database languages like Oracle, MySQL, MS Access, dBase, FoxPro etc. SQL statements commonly used in Oracle and MS Access can be categorized as data definition language (DDL), data control language (DCL) and data manipulation language (DML).

Data Definition Language (DDL):

It is a language that allows the users to define data and their relationship to other types of data. It is mainly used to create files, databases, data dictionary and tables within databases.It is also used to specify the structure of each table, set of associated values with each attribute, integrity constraints, security and authorization information for each table and physical storage structure of each table on disk.

The following table gives an overview about usage of DDL statements in SQL

S.No	Need and Usage	The SQL DDL statement
1	Create schema objects	CREATE
2	Alter schema objects	ALTER
3	Delete schema objects	DROP
4	Reneme schema objects	RENAME

The DDL is used by DBA and database designers to define the conceptual and internal schema when there is no clean separation between them. If a clear separation is maintained between the conceptual and internal schema then DDL is used to define only the conceptual schema.

Data Manipulation Languages (DML):

It is a language that provides a set of operations to support the basic data manipulation operations on the data held in the databases. It allows users to insert, update, delete and retrieve data from the database. The part of DML that involves data retrieval is called a query language. DML is used to perform the various manipulations on the database such as to retrieve, insert, delete, modify etc.The following table gives an overview about the usage of DML statements in SQL:

S. No	Need and Usage	The SQL DML statement
1	Remove rows from tables or views	DELETE
2	Add new rows of data into table or view	INSERT
3	Retrieve data from one or more tables	SELECT
4	change column values In existing rows of a table or view	UPDATE

There are two types of DML's:

a) High level or nonprocedural DML:

· It can be used to specify complex database operations in a concise manner.

· High level DML's can either be entered interactively from a terminal or it can be embedded in a general purpose programming language.

· It is also called set-at-a time as many records are processed at a time.

b) Low level of procedural DML:

· This must be embedded in a general purpose programming language.

· This DML retrieves individual record from the database and processes each separately. Therefore it is called record at a time DML.

In the above both low level and high level DML's when embedded in a programming language. The language is called host language and the DML is called the data sublanguage.

Data Control Language (DCL):

DCL statements control access to data and the database using statements such as GRANT and REVOKE. A privilege can either be granted to a User with the help of GRANT statement. The privileges assigned can be SELECT, ALTER, DELETE, EXECUTE, INSERT, INDEX etc. In addition to granting of privileges, you can also revoke (taken back) it by using REVOKE command.

Transaction Control Language (TCL)

TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

- **Commit:** It is used to save the transaction on the database.
- **Rollback:** It is used to restore the database to original since the last Commit.

DBMS Interfaces:

1. Menu Based Interfaces:

- This Interface provides the user with a list of options called menu that leads the user through the formulation of a request.

- Query is composed in step by step by picking options from a menu that is displayed by the system.
- It is often used in browsing interfaces.

2. Form based interface:

- A form based interface displays a form to each user.
- Users can fill out the form entries or certain entries using which the queries will be created.
- DBMS's will have the form specification languages that help programmers to specify such form.

3. Graphical user interface (GUI):

- GUI displays schema to the user in diagrammatic form and user can specify the request by manipulating the diagram.
- GUI's use menu and forms.

4. Natural language interface:

- This interface accepts the request in English or some other language.
- Natural language interface has its own schema and a set of words to interpret the request.

5. Interfaces for parametric users:

- Parametric users like bank tellers have a small set of operations that must be performed repeatedly.
- The function keys in a terminal can be programmed to initiate various commands.

6. Interfaces for DBA:

- It is the set of privileged commands used by the DBA staff, to create accounts, grant authorizations etc.

DBMS Interfaces

A database management system (DBMS) interface is a user interface which allows for the ability to input queries to a database without using the query language itself. A DBMS interface could be a web client, a local client that runs on a desktop computer, or even a mobile app

1. Menu Based Interfaces:

This Interface provides the user with a list of options called menu that leads the user through the formulation of a request.Query is composed in step by step by picking options from a menu that is displayed by the system.It is often used in browsing interfaces.

2. Form based interface:

A form based interface displays a form to each user.Users can fill out the form entries or certain entries using which the queries will be created. DBMS's will have the form specification languages that help programmers to specify such form.

3. Graphical user interface (GUI):

GUI displays schema to the user in diagrammatic form and user can specify the request by manipulating the diagram.GUI's use menu and forms.

4. Natural language interface:

This interface accepts the request in English or some other language.Natural language interface has its own schema and a set of words to interpret the request.

5. Interfaces for parametric users:

Parametric users, such as bank tellers, often have a small set of operations that they must perform repeatedly. For example, a teller is able to use single function keys to invoke routine and repetitive transactions such as account deposits or withdrawals, or balance inquiries.The function keys in a terminal can be programmed to initiate various commands.

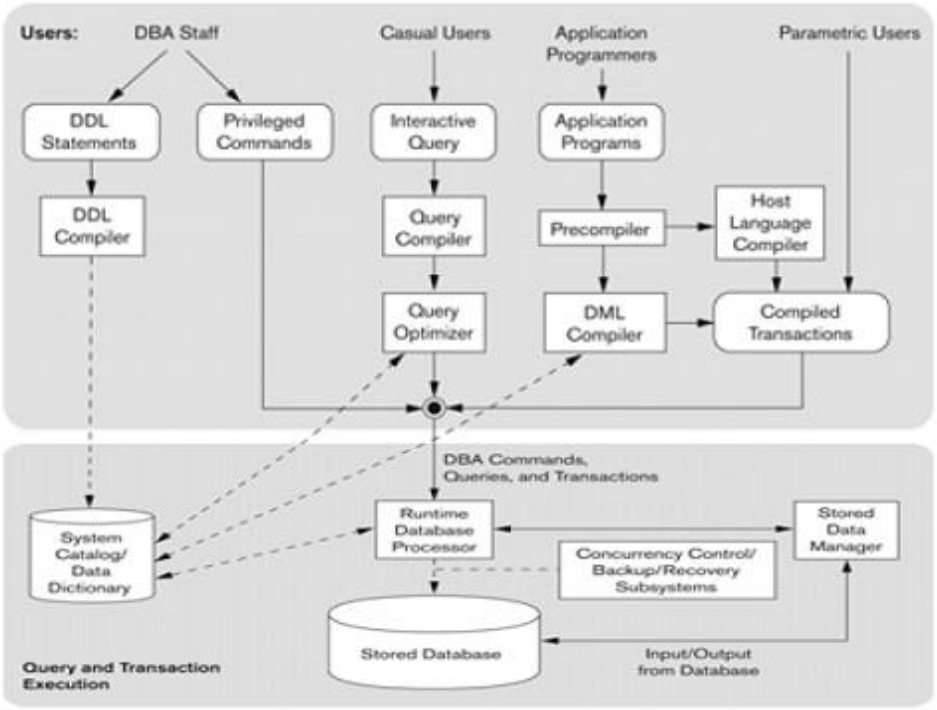
6. Interfaces for DBA:

It is the set of privileged commands used by the DBA staff, to create accounts, grant authorizations etc.Most database systems contain privileged commands that can be used only by the DBA staff. These include commands for creating accounts, setting system parameters, granting account authorization, changing a schema, and reorganizing the storage structures of a database

7.Apps for Mobile Devices:

These interfaces present mobile users with access to their data. For example, banking, reservations, and insurance companies, among many others, provide apps that allow users to access their data through a mobile phone or mobile device.

Database System Environment
Database System Environment is depicted below:



1. DBMS component Modules.

The database and DBMS catalog is stored on the disk. Access to the disk is controlled primarily by the operating system which schedules disk input/output.

A higher level stored data manager module of DBMS controls access to DBMS information that is stored on the disk (i.e.) database or DBMS catalog. It uses the basic operating system services for carrying out low level data transfer between disk and main storage.

The DDL compiler processes schema definitions specified in DDL and stores the descriptions of the schema in the DBMS catalog.

The run time database processor handles database accesses at run time. It receives the retrieval or update operations and carries them out on the database.

Access to the disk goes through **stored data manager.**

The Query Compiler handles high level queries that are entered interactively. It passes, analyzes and compiles or interprets a query by creating database access code and then generates calls to the run time processor for executing the code.

Pre-compiler extracts DML commands from an application program written in host programming language. These commands are send to DML Compiler for compilation into object code for database access.

The rest of the program is sent to **host language compiler**. The object code for DML commands and the rest of the program are linked forming canned transactions, whose executable code includes calls to the runtime database processor.

2. Database system utilities:

Common database utilities have these functions.

i. Loading:

It is used to load the existing data files such as text files or sequential files into the database when the source file format and target file format is different, this utility does the conversion automatically.

ii. Backup:

This creates a backup copy of the database, usually dumping the entire database on the tape. Incremental backups are also used to record the change to the previous backup.

iii. File Reorganization:

This is used to reorganize a database file into a different file organization to improve performance.

iv. Performance Monitoring:

It monitors database usage and provides statistics files to improve performance. Other utilities exist for sorting, file compressions etc.

3. Tools, Applications Environments and communications facilities:

a. CASE Tools :

It is used in the design phase to help speed up the development process.

b. Data dictionary system

It stores catalog information about schemas and constraints, as well as design decisions, usage standards, application program descriptions, user information. It is also called as information repository. It can be accessed directly by DBA or users when needed.

c. Application development environments

It provides an environment for developing database applications and includes facilities to help in database design, querying and updating, GUI development and application development. (i.e. JBuilder)

d. Communication software

It allows the users at remote locations to access the database through computer terminals, workstations or personal computers. It is connected to the database through data communications hardware such as phone lines, local area networks etc.

Classification of DBMS
DBMS can be classified in various ways:

1. **Based on the Data Model**

A. **Relational data model**

The relational Data Model represents a database as a collection of tables, where each table can be stored as a separate file.

B. **Hierarchical data model**

In a hierarchical data model, data is organized into a tree like structure.

C. **Network data model**

The network data model represents data in terms of records and also 1: N relationship.

D. **Object relational data model**

These add new object storage capabilities to the relational systems at the core of modern information systems, by encapsulating methods with data structures.

2. **Based on the number of Users**

A.**Single User systems**

A single user system supports one user at a time. It is used mainly in personal computers.

B. **Multi User systems**

A multiuser system supports multiple users at a time, which is implemented in most of the computers.

3. **Based on the number of Sites on which database is distributed**

A.**Centralized DBMS**

It contains the database at only one computer site, which can support multiple users.

B. **Distributed DBMS**

It contains the database and DBMS software stored over many sites connected by network.

C. **Homogeneous**

It uses the same DBMS software at multiple sites.

D. **Federated DBMS (Multiple database system)**

The participating DBMS's are loosely coupled and have a degree of local autonomy.

E. **Online transaction processing(OLTP)**

These support a large number of concurrent transactions without imposing excessive delays.

4. **Based on types of access path**

A.**Special purpose**

It is a package developed to suit the needs of a particular user, which cannot be used by other users.

B.**General purpose**

It can be used for any type of application.

5. **Based on Cost**

Majority of DBMS cost more. Single user low-end system that works with microcomputers cost relatively lesser between $100 a