Syllabus: Introduction to Databases: Introduction, An Example, Characteristics of the Database Approach, Actors on Scene, Workers behind the scene, Advantages of Using the DBMS Approach, A Brief History of Database Applications, When Not to Use a DBMS Overview of Database Languages and Architectures: Data Models, Schemas and Instances, Three Schema Architecture and Data Independence, Database Languages and Interfaces, The Database System Environment, Centralized and Client/Server Architecture for DBMSs, Classification of Database Management Systems.

Introduction to Databases:

- A database is a collection of related data or information. Data means any raw facts such as its combination of A to Z alphabetic, 0-9 combinational numbers, characters , images, videos, audios etc…
- A database is **an organized collection of structured information, or data**, typically stored electronically in a computer system.
- A database may be generated and maintained manually or it may be computerized.
- A database can be of any size and complexity. For example, the list of names and addresses referred to earlier may consist of only a few hundred records, each with a simple structure.
- An example of a large commercial database is Amazon.com. It contains data for over 20 million books, CDs, videos, DVDs, games, electronics, apparel, and other items.

Database Management System (DBMS):

A database management system is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.

- **Defining** a database involves specifying the data types, structures, and constraints of the data to be stored in the database.
- **Constructing** the database is the process of storing the data on some storage medium that is controlled by the DBMS.
- **Manipulating** a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the data.
- **Sharing** a database allows multiple users and programs to access the database simultaneously.
- An **application** program accesses the database by sending queries or requests for data to the DBMS.
- Other important functions provided by the DBMS include protecting the database and maintaining it over a long period of time.
- Protection includes system protection against hardware or software malfunction (or crashes) and security protection against unauthorized or malicious access.
- A typical large database may have a life cycle of many years, so the DBMS must be able to maintain the database system by allowing the system to evolve as requirements change over time.

****

An Example:

Consider the example of university system. UNIVERSITY database for maintaining information concerning students, courses, and grades in a university environment. The STUDENT file stores data on each student, the COURSE file stores data on each course, the SECTION file stores data on each section of a course, the GRADE_REPORT file stores the grades that students receive in the various sections they have completed each course.

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

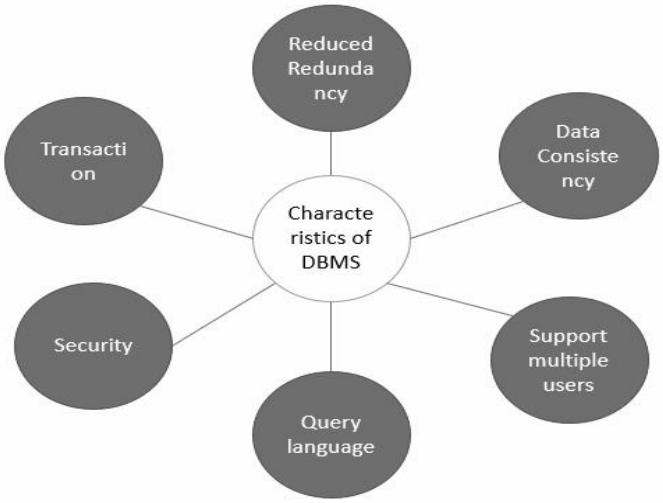
****

Characteristics of the Database Approach:

The main **characteristics** of the database approaches are the following:

- Self-describing nature of a database system.
- Insulation between programs and data, and data abstraction.
- Support of multiple views of the data.
- Sharing of data and multiuser transaction processing.

The characteristics of DBMS are explained below in pictorial form:



- A transaction is a very small unit of a program and it may contain several low level tasks. A transaction in a database system must maintain Atomicity, Consistency, Isolation, and Durability − commonly known as ACID properties − in order to ensure accuracy, completeness, and data integrity.
- Data redundancy occurs when the same piece of data exists in multiple places, it occurs data inconsistency and reduce redundancy maintain consistency.
- Security is another important characteristic of the database approach, as it ensures that the data is protected from unauthorized access, modification, or deletion.
- The Database management system allows so many users to access databases at the same time.
- It also provides users query language, helpful to insert, retrieve, update, and delete the data in a database.
- Data consistency means that each user sees a consistent view of the data, including visible changes made by the user's own transactions and transactions of other users.

****

Actors on Scene:

- Database users are categorized based up on their interaction with the database. These are seven types of database users in DBMS.

1. **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. DBA is also responsible for providing security to the database.DBA also monitors the recovery and backup and provides technical support. DBA repairs damage caused due to hardware and/or software failures. DBA is the one having privileges to users.
2. **Naïve users:** who don't have any DBMS knowledge but they frequently use the database applications in their daily life. For examples, Railway's ticket booking users are naive users.
3. **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.
4. **System analysts:** 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.
5. **Application programmers**: Application programmers implement these specifications as programs; then they test, debug, document, and maintain these canned transactions.
6. **Sophisticated Users:** Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database.
7. **Casual Users / Temporary Users :** Casual Users are the users who occasionally use/access the database but each time when they access the database they require the new information,

****

Workers behind the scene:

Who design, use, and administer a database, others are associated with the design, development, and operation of the DBMS software and system environment. These persons are typically not interested in the database content itself. We call them the workers behind the scene.

- **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.

- **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.
- **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:

Some of the advantages of using a DBMS

1. **Controlling Redundancy:**

   In traditional software development utilizing file processing, every user group maintains its own files for handling its data-processing applications. This redundancy in storing the same data multiple times leads to several problems. First, there is the need to perform a single logical update—such as entering data on a new student—multiple times: once for each file where student data is recorded. This leads to duplication of effort. Second, storage space is wasted when the same data is stored repeatedly, and this problem may be serious for large databases. Third, files that represent the same data may become inconsistent.

2. **Restricting 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. A DBMS should provide a security and authorization subsystem, which the DBA uses to create accounts and to specify account restrictions. Then, the DBMS should enforce these restrictions automatically.

3. **Providing Persistent Storage for Program Objects:**

   The persistent storage of program objects and data structures is an important function of database systems. Traditional database systems often suffered from the so called mismatch problem, since the data structures provided by the DBMS were incompatible with the programming language's data structures. Object-oriented database systems typically offer data structure compatibility with one or more object-oriented programming languages.

4. **Providing Storage Structures and Search Techniques for Efficient Query Processing:**

   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 used for this purpose. Indexes are typically based on tree data structures. 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

5. **Providing 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.

6. **Providing Multiple User Interfaces:**

   DBMS should provide a variety of user interfaces. Access the data either command line interface or graphical interface or programming interface.

7. **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.

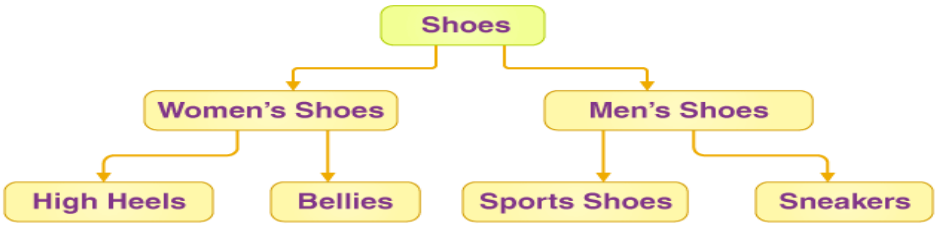
8. **Fast Data Sharing** :
   Database administration makes it possible for consumers to access more and better-managed data. Multiple users access the data parallel.

****

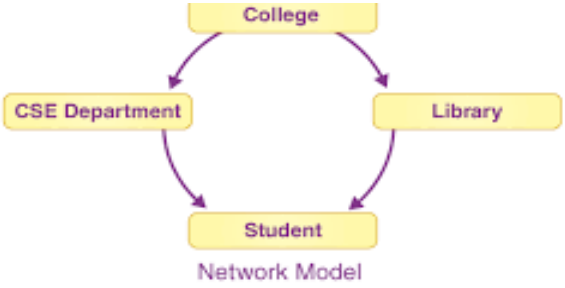
A Brief History of Database Applications:

A database management system (DBMS) allows a person to organize, store, and retrieve data from a computer. It is a way of communicating with a computer's "stored memory."

- In the very early years of computers, "punch cards" were used for input, output, and data storage. Punch cards offered a fast way to enter data and retrieve.
- 1950s and early 1960s: o Data processing using magnetic tapes for storage § Tapes provided only sequential access.
- Late 1960s and 1970s: o Hard disks allowed direct access to data.
- **Early Database Applications Using Hierarchical and Network Systems**:
  - **The Hierarchical Model** was the first database management system model. This concept uses a hierarchical tree structure to organize the data. The hierarchy begins at the root, which contains root data, and then grows into a tree as child nodes are added to the parent node.



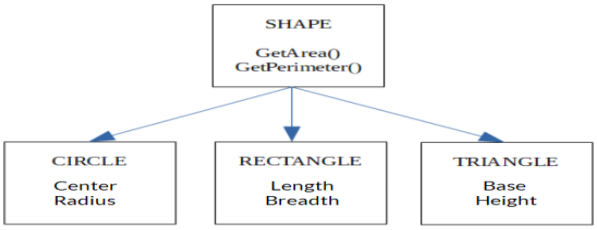
  - **Network Model in DBMS** is a hierarchical model that is used to represent the many-to-many relationship among the database constraints. It is a simple and easy-to-construct database model. The Network Model in DBMS is based on the set of nodes and links.



- **Relational Databases:**
  - The relational data model also introduced high-level query languages that provided an alternative to programming language interfaces, making it much faster to write new queries.
  - In relational model, the data and relationships are represented by collection of inter-related tables. Each table is a group of column and rows, where column represents attribute of an entity and rows represents records.

Student-id	Student-name	branch	Phone number
101	Rekha	MCA	9999555522
102	Geetha	MBA	9933446688
103	Kumar	MCA	8844998877
104	Veeru	MBA	1702271702

- **Object-Oriented Applications**: The real world entities and situations are represented as objects in the Object oriented database model.



- **Interchanging Data on the Web for E-Commerce Using XML:**
  - In the 1990s, electronic commerce (e-commerce) emerged as a major application on the Web.
  - Currently, Extended Markup Language (XML) is considered to be the primary standard for interchanging data among various types of databases and Web pages.
- **Extending Database Capabilities for New Applications:**
  - Database systems now offer extensions to better support the specialized requirements for some of these applications. The following are some examples of these applications:
    - Scientific applications
    - Storage and retrieval of images
    - Storage and retrieval of videos
    - Data mining applications that analyze large amounts of data.
    - Spatial applications that store spatial locations of data.
    - Time series applications.

****

When Not to Use a 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.

****

Data Models:

A data model—a collection of concepts that can be used to describe the structure of a database. By structure of a database we mean the data types, relationships, and constraints that apply to the data. Most data models also include a set of basic operations for specifying retrievals and updates on the database.

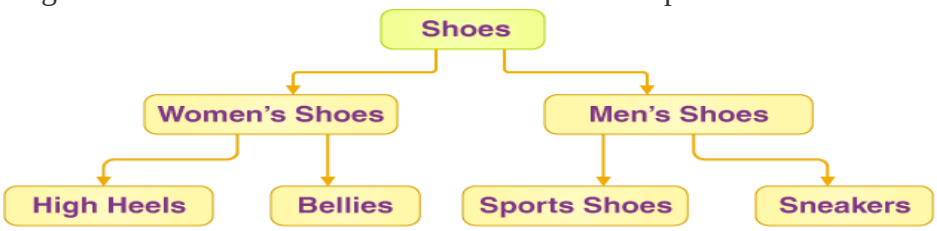
- **Categories of Data Models**:

  Many data models have been proposed, which we can categorize according to the types of concepts they use to describe the database structure.

1. Hierarchical Model
2. Network Model
3. Entity-Relationship Model
4. Relational Model
5. Object-Oriented Data Model

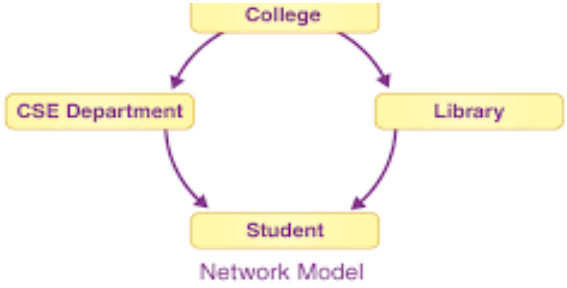
Hierarchical Model:

- The Hierarchical Model was the first database management system model. This concept uses a hierarchical tree structure to organize the data. The hierarchy begins at the root, which contains root data, and then grows into a tree as child nodes are added to the parent node.



Network Model in DBMS:

- Network Model is a hierarchical model that is used to represent the many-to-many relationship among the database constraints. It is a simple and easy-to-construct database model. The Network Model in DBMS is based on the set of nodes and links.



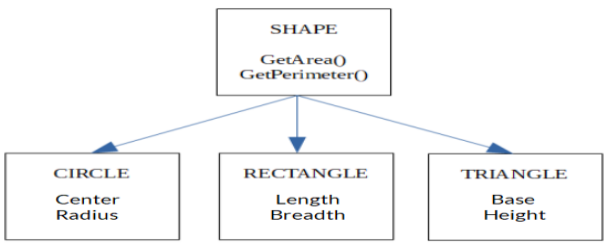
Relational Databases:

- The relational data model also introduced high-level query languages that provided an alternative to programming language interfaces, making it much faster to write new queries. In relational model, the data and relationships are represented by collection of inter-related tables. Each table is a group of column and rows, where column represents attribute of an entity and rows represents records.

Student-id	Student-name	branch	Phone number
101	Rekha	MCA	9999555522
102	Geetha	MBA	9933446688
103	Kumar	MCA	8844998877
104	Veeru	MBA	1702271702

Object-Oriented Applications:

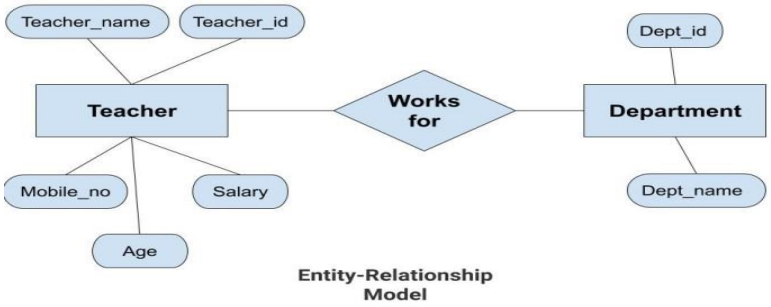
- The real world entities and situations are represented as objects in the Object oriented database model. In this model, both the data and relationship are present in a single structure known as an object. In this model, two are more objects are connected through links.



Entity Relational model:

- Entity-Relationship Model or simply ER Model is a high-level data model diagram. In this model, we represent the real-world problem in the pictorial form to make it easy for the stakeholders to understand. We use the ER diagram as a visual tool to represent an ER Model. ER diagram has the following three components:

- **Entities:** Entity is a real-world thing. It can be a person, place, or even a concept. Example: Teachers, Students, Course, Building, Department, etc are some of the entities of a School Management System.
- **Attributes:** An entity contains a real-world property called attribute. This is the characteristics of that attribute. Example: The entity teacher has the property like teacher id, salary, age, etc.
- **Relationship:** Relationship tells how two attributes are related. Example: Teacher works for a department.



Entity-Relationship Model

****

Schemas and Instances:

Schemas:

- It refers to an overall description that we get for any given database. In simpler words, schema refers to the basic structure of how one needs to store data in any database. There are basically two types of Schema: Physical Schema and Logical Schema.
  Physical Schema – This schema describes the database designed at a physical level.
  Logical Schema – This schema describes the database designed at a logical level.

Instances:

- In DBMS, the data is stored for a particular amount of time and is called an instance of the database. The database schema defines the attributes of the database in the particular DBMS. The value of the particular attribute at a particular moment in time is known as an instance of the DBMS.

The major differences between schema and instance are as follows:

Database Schema	Database Instance
It is the definition of the database, or it is defined as the description of the database.	It is a snapshot of a database at a specific moment
This corresponds to the variable declaration of a programming language.	The value of the variable in a program at a point in time corresponds to an instance of the database schema.
It rarely changes.	It changes frequently.
This corresponds to the variable declaration of a programming language.	The value of the variable in a program at a point in time corresponds to an instance of the database schema.
Defines the basic structure of the database, i.e., how the data will be stored in the database.	It is the set of Information stored at a particular time.
Schema is same for whole database.	Data in instances can be changed using addition, deletion, updating.

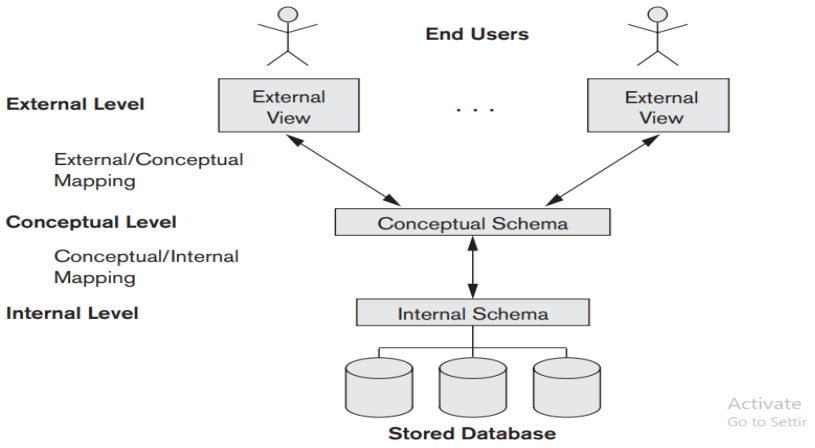
****

Three-Schema Architecture:

The goal of the three-schema architecture, is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels:

1. Internal Level
2. Conceptual Level
3. External Level

- **The internal level** has an internal schema, which 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 paths for the database.
- **The conceptual level** has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.
- **The external or view level** includes a number of 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. As in the previous level, each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level data model



****

Data Independence:

- Data Independence can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level.

We can define two types of data independence:

1. Logical data independence.
2. Physical data independence.

Logical data independence:

- Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item).

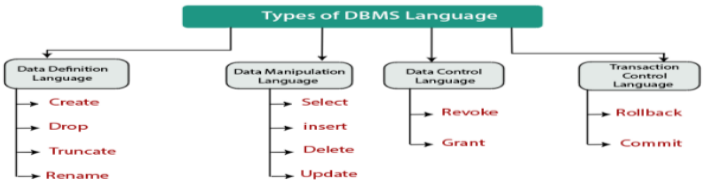
Physical data independence:

- Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized. for example, by creating additional access structures to improve the performance of retrieval or update. If the same data as before remains in the database, we should not have to change the conceptual schema.

****

Database Languages and Interfaces:

- Once the design of a database is completed and a DBMS is chosen to implement the database, the first step is to specify conceptual and internal schemas for the database and any mappings between the two. Database languages are
  1. Data Definition Language.
  2. Data Manipulation Language.
  3. Data Control Language.
  4. Transaction Control Language



Data Definition Language:

- Data definition language (DDL), is used by the DBA and by database designers to define both schemas. It used to define and modify the structure of the database.

1. **Create:**
   Create command is used create any data base object like table, database, view…
   Syntax: create table tablename
   (filed1 datatype(size),
   filed2 datatype(size),
   ……..);
   Ex: Create table student
   (id number(5),
   name varchar2(10)),
   branch varchar2(5));
2. **Drop:**
   Drop command is used to remove database object .
   Syntax: Drop table tablename;
   Ex: Drop table Student;
3. **Alter:**
   Alter command is used to change the structure of the database.
   - **Alter-add**: Alter add command is used to add new column in table
     Syntax: alter table tablename add newcolumn datatype;
     Ex: alter table student add phno number(10);
   - **Alter-modify:** Alter modify command is used to change the type and size of the datatype.
     Syntax: alter table tablename modify column datatype(size);
     Ex: alter table student modify branch char(6);
   - **Alter-drop:** Alter drop is used to remove the table column.
     Syntax: alter table tablename drop column columnname;
     Ex: alter table student drop column phno;
   - **Alter –rename:** Alter rename is used to change the column name
     Syntax: alter table tablename rename column oldname to newname;
     Ex: alter table student rename column branch to section;

4. **Truncate :**

   Truncate command is used to remove all the data from table.

   Syntax: truncate table tablename;

   Ex: truncate table student;

5. **Rename:**

   Rename command is used to change the table name.

   Syntax: rename oldtablename to newtablename

   Ex: rename student to mentee

Data Manipulation Language:

- Data manipulation language is used to manipulate the data on database. Data manipulation language is used to perform operation on data not structure.

1. **Insert:** Insert command is used to insert the data into table.

   Syntax: insert into tablename (field1,field2……)values(value1,value2…..);

   Ex: insert into student (id,name,section)values(101,'rekha','MCA');

   Or

   Syntax: insert into tablename values(value1,value2…..);

   Ex: insert into student values(101,'rekha','MCA');

2. **Select:**

   Select is used to retrieve information from table

   - **Select-all**: select all is used to retrieve the all information from table.

     Syntax: select * from tablename;

     Ex: select * from student;

   - **Selected-fields**: select fields are is used to retrieve the only selected fields from table

     Syntax: select filed1,filed2 from tablename;

     Ex: select id, name from student;

   - **Selected-record**: select records are is used to retrieve the only selected records from table

     Syntax: select filed1,filed2 from tablename where condition;

     Ex: select id, name from student where id=101;

3. **Update :**

   Update command is used to change the data in table.

   - **Update –all**: update all command change all records at a time

     Syntax: update tablename set column=value;

     Ex: update student set branch='MCA';

   - **Update –selected** : update all command change all records at a time

     Syntax: update tablename set column=value;

     Ex: update student set branch='MCA';

4. **Delete :**

   Delete command is used to remove data from tables.

   - **Delete-all**: Delete all the information from table.

     Syntax: delete from tablename;

     Ex: delete from student;

   - **Delete-selected:** Delete selected records from table.

     Syntax: delete from tablename where condition;

     Ex: delete from student where id=101;

Data Control Language:

- Data control language is used to control the data operations by DBA when user performs transactions. Data control language commands are
  1. **Grant:** Grant is used to give permissions to user.
     Syntax: grant privilege
     on object
     To{user-public-role}
     Ex: grant select on student to rekha;
  2. Revoke: Revoke is used to withdraw permissions from user
     Syntax: grant privilege
     on object
     from {user-public-role}
     Ex: revoke select on student from rekha;

Transactions Control Language:

- Transaction Control language is a language that manages transactions within the database. It is used to execute the changes made by the DML statements.
  1. **Commit:** It is used to save the transactions in the database.
     Syntax: commit;
     Ex: commit;
  2. **Rollback**: It is used to restore the database to that state which was last committed.
     syntax: rollback [to savepoint_name];
     Ex: rollback update5;
  3. **Savepoint:** Savepoint helps to save the transaction temporarily.
     Syntax: **savepoint [savepoint_name;]**
     Ex: savepoint table_create;

****

DBMS Interfaces:

User-friendly interfaces provided by a DBMS may include the following:

- **Menu-Based Interfaces for Web Clients or Browsing:**

  These interfaces present the user with lists of options (called menus) that lead the user through the formulation of a request. The query is composed step-bystep by picking options from a menu that is displayed by the system. Pull-down menus are a very popular technique in Web-based user interfaces. They are also often used in browsing interfaces.

  Ex: Oracle, MongoDB
- **Forms-Based Interfaces**:

  A forms-based interface displays a form to each user. Users can fill out all of the form entries to insert new data, or they can fill out only certain entries, in which case the DBMS will retrieve matching data for the remaining entries.

  Ex: Oracle, SQL Plus
- **Graphical User Interfaces:**

  A GUI typically displays a schema to the user in diagrammatic form. The user then can specify a query by manipulating the diagram. In many cases, GUIs utilize both menus and forms. Most GUIs use a pointing device, such as a mouse, to select certain parts of the displayed schema diagram.

  Ex: Oracle, MongoDB
- **Natural Language Interfaces**:

  These interfaces accept requests written in English or some other language and attempt to

understand them. A natural language interface usually has its own schema, which is similar to the database conceptual schema, as well as a dictionary of important words. A high-level query corresponding to the natural language request and submits it to the DBMS for processing.

- **Speech Input and Output:**

  Limited use of speech as an input query and speech as an answer to a question or result of a request is becoming commonplace. The speech input is detected using a library of predefined words and used to set up the parameters that are supplied to the queries. For output, a similar conversion from text or numbers into speech takes place.

  Ex: Inquiries for telephone directory, flight arrival/departure, and credit card account information are allowing speech for input and output to enable customers to access this information,

- **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 Database System Environment:

Database system components:

- Database system environment means DBMS is a complex software system, how the computer software and computer systems are interact with DBMS. DBMS components are divided three parts. Lower part it describes how data is stored in physical memory. Middle part DBMS responsible for storage of data and processing of transactions. Higher part related to user.
- The database and the DBMS catalog are usually stored on disk. Access to the disk is controlled primarily by the operating system (OS), which schedules disk read/write. Many DBMSs have their own buffer management module to schedule disk read/write, because this has a considerable effect on performance.
- A higher-level stored data manager module of the DBMS controls access to DBMS information that is stored on disk, user using interactive query interface. The DDL compiler processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog by DBA. User applies DML commands or application programs and interacts with data and manages data on database.

DBA manage the database system with common utilities.

- **Loading:** A loading utility is used to load existing data files—such as text files or sequential files—into the database.
- **Backup:** A backup utility creates a backup copy of the database, usually by dumping the entire database onto tape or other mass storage medium. The backup copy can be used to restore the database in case of catastrophic disk failure
- **Database storage reorganization:** This utility can be used to reorganize a set of database files into different file organizations, and create new access paths to improve performance.
- **Performance monitoring**. Such a utility monitors database usage and provides statistics to the DBA. The DBA uses the statistics in making decisions such as whether or not to reorganize files or whether to add or drop indexes to improve performance.
- **Other utilities:** May be available for sorting files, handling data compression, monitoring access by users, interfacing with the network, and performing other functions.

Tools, Application Environments, and Communications Facilities:

- Other tools are often available to database designers, users, and the DBMS. CASE tools12 are used in the design phase of database systems. Another tool that can be quite useful in large organizations is an

expanded data dictionary (or data repository) system.

- Application development environments, such as PowerBuilder (Sybase) or JBuilder (Borland), have been quite popular. These systems provide an environment for developing database applications and include facilities that help in many facets of database systems, including database design, GUI development, querying and updating, and application program development.
- The DBMS also needs to interface with communications software, whose function is to allow users at locations remote from the database system site to access the database through computer terminals, workstations, or personal computers.

****

Centralized and Client/Server Architecture for DBMS:

The Database Management System (DBMS) architecture shows how data in the database is viewed by the users. DBMS fallows three-tier architecture. The architecture of a database system is greatly influenced by the underlying computer system on which the database is running:

i. Centralized.
ii. Client-server.
iii. Parallel (multi-processor).
iv. Distributed

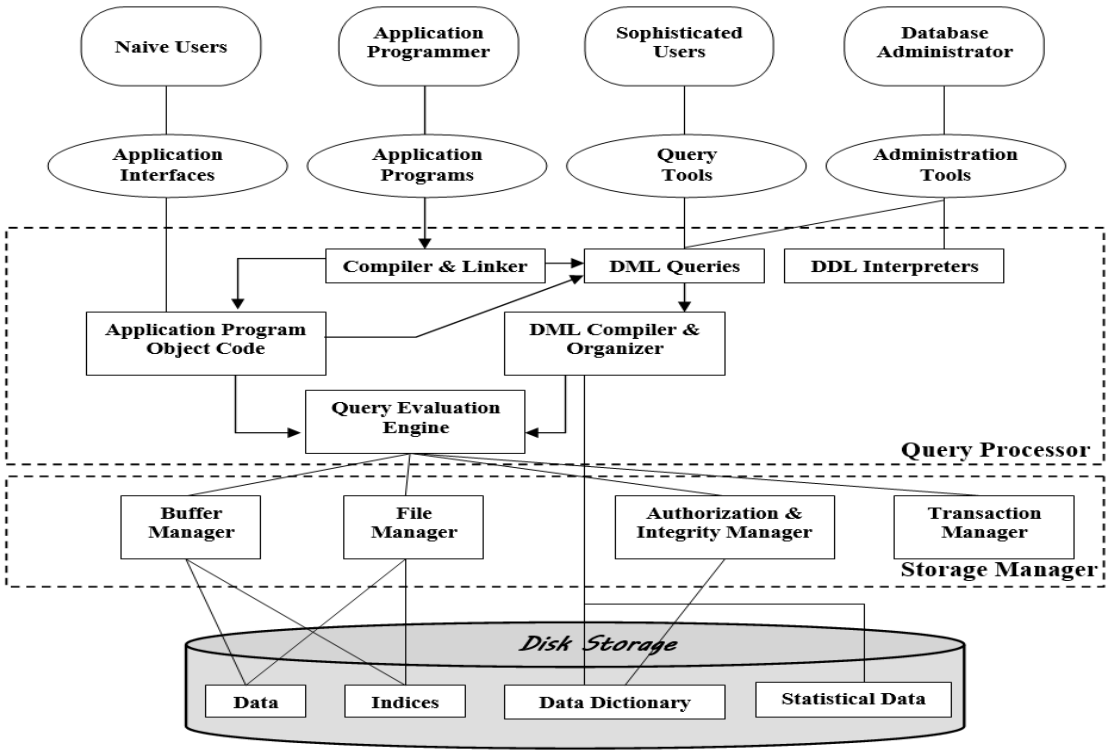


Figure: System Architecture

Database Users:

Users are differentiated by the way they expect to interact with the system:

- **Application programmers:**
  - Application programmers are computer professionals who write application programs.
  - Application programmers can choose from many tools to develop user interfaces.
- **Sophisticated users:**
  - Sophisticated users interact with the system application programs and database Languages.
  - Sophisticated users are having knowledge on multiple tools.
- **Naïve users :**
  - Naive users are unsophisticated users who interact with the system without any knowledge by invoking one of the application programming interfaces that have been written previously.

- **Database Administrator:**
  coordinates all the activities of the database system. Database administrator's duties include:
  - Schema definition: The DBA creates the original database schema by executing a set of data definition statements in the DDL.
  - Storage structure and access method definition.
  - Schema and physical organization modification
  - Granting user authority to access the database: By granting different types of authorization, the database administrator can regulate which parts of the database various users can access.
  - Specifying integrity constraints.
  - Monitoring performance and responding to changes in requirements.
  - Backup and recovery the data.
  - Providing security.

Query Processor:

The query processor will accept query from user and solves it by accessing the database.

Parts of Query processor:

- **DDL interpreter**
  This will interprets DDL statements and fetch the definitions in the data dictionary.
- **DML compiler**
  - This will translates DML statements in a query language into low level instructions that the query evaluation engine understands.
  - A query can usually be translated into any of a number of alternative evaluation plans for same query result DML compiler will select best plan for query optimization.
- **Query evaluation engine**
  This engine will execute low-level instructions generated by the DML compiler on DBMS.

Storage Manager/Storage Management:

A storage manager is a program module which acts like interface between the data stored in a database and the application programs and queries submitted to the system. Thus, the storage manager is responsible for storing, retrieving and updating data in the database.

The storage manager components include:

- **Authorization and integrity manager:** Checks for integrity constraints and authority of users to access data.
- **Transaction manager:** Ensures that the database remains in a consistent state although there are system failures.
- **File manager:** Manages the allocation of space on disk storage and the data structures used to represent information stored on disk.
- **Buffer manager:** It is responsible for retrieving data from disk storage into main memory. It enables the database to handle data sizes that are much larger than the size of main memory.

Memory storage:

Data structures implemented by storage manager.

- **Data files:** Stored in the database itself.
- **Data dictionary:** Stores metadata about the structure of the database.
- **Indices:** Provide fast access to data items.
- **Statistical data:** Statistical analysis performed on memory.

****

Classification of Database Management Systems:

Classification of database management system is based on various parameters such as the kind of data model used to construct the DBMS, the number of users that will be using the database system, the way in which the database is distributed

1. Based on Data Model
2. Based on Number of Users
3. Based on Database Distribution
4. Based on Cost of Database
5. Based on Usage

Based on Data Model

. Depending upon how the data is structured, data models are further classified into:

- **Relational Data Model:**
  In the relational data model, we use tables to represent data and the relationship among that data. Each of the tables in the relational data model has a unique name. A table has multiple columns where each column name is unique. A table holds records which has value for each column of the table. The relational database model is the most currently used data model.
- **Entity-Relationship Model**
  The Entity-Relationship model (E-R data model) represents data using objects and the relationship among these objects. These objects are referred to as entities that represent the real 'thing' or 'object' in the real world.
- **Object-Based Data Model**
  The object-based data model is an extension of the E-R model which also include notion for encapsulation, methods. There is also an object-relational data model which is a combination of the object-oriented data model and relational data model.
- **Semi structured Data Model**
  The semi structured data model is different from what we have studied above. In the semi structured data model, the data items or objects of the same kind might have a different set of attributes. The Extensible Markup Language represents the semi structured data. The hierarchical data model stores the data in the form of records and uses a tree structure to represent these records. Network data model was introduced which allow the multiple parent record for a single child record.

Based on Number of Users:

The database management system can also be classified on the basis of its user. So, a DBMS can either be used by a single user or it can be used by multiple users.

- The database system that can be used by a single user at a time is referred to **as a single-user system**
- The database system that can be used by multiple users at a time is referred to as a **multiple usersystem**.

Based on Database Distribution:

Depending on the distribution of the database over numerous sites we can classify the database as:

- **Centralized DBMS**:
  In the centralized DBMS, the entire database is stored in a single computer site. Though the centralized database supports multiple users still the DBMS software and the data both are stores on a single computer site.
- **Distributed DBMS**
  In the distributed DBMS (DDBMS) the database and the DBMS software are distributed over many computer sites. These computer sites are connected via a computer network. The DDBMS is further classified as homogeneous DDBMS and heterogeneous DDBMS.
  - **Homogeneous DDBMS:** The homogeneous DDBMS has the same DBMS software at all the distributed sites.
  - **Heterogeneous DDBMS:** The heterogeneous DDBMS has different DBMS software for different sites.

Based on Cost of Database:

Well, it is quite difficult to classify the database on the basis of its cost as nowadays you can have free open source DBMS products such as MySQL and PostgreSQL. Although the personal version of RDBMS can cost up to $100.You may also have to pay millions of dollars for the installation and maintenance of a large database system.

Based on Usage:

On the basis of the access path that is used to store the files, the database can be classified as general-purpose DBMS and special-purpose DBMS.

- The **special-purpose DBMS** is the one that is designed for a specific application and it can not be used for another application .
- **Online transaction processing (OLTP):** The OLTP system supports a large number of transactions concurrently without any delay.
- The **general-purpose** DBMS is the one that is designed to meet the need of as many applications as possible

****