| DBMS | File System |
|---|---|
| DBMS is a collection of data. In DBMS, the user is not required to write the procedures.<br><br>Searching data is easy in Dbms | File system is a collection of data. In this system, the user has to write the procedures for managing the database.<br><br>Searching is difficult in File System |
| Dbms is structured data<br><br>No data redundancy in Dbms<br><br>Memory utilisation well in dbms<br><br>No data inconsistency in dbms | Files are unstructured data<br><br>Data redundancy is there in file system<br><br>Memory utilisation poor in file system<br><br>Inconsistency in file system |

| | |
|---|---|
| DBMS gives an abstract view of data that hides the details. | File system provides the detail of the data representation and storage of data. |
| DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure. | File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost. |
| DBMS provides a good protection mechanism. | It is very difficult to protect a file under the file system. |
| DBMS contains a wide variety of sophisticated techniques to store and retrieve the data. | File system can't efficiently store and retrieve the data. |
| DBMS takes care of Concurrent access of data using some form of locking. | In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information. |

**DATABASE APPLICATIONS – DBMS:** Applications where we use Database Management Systems are:

Telecom: There is a database to keeps track of the information regarding calls made, network usage, customer details etc.

Industry: Where it is a manufacturing unit, warehouse or distribution centre, each one needs a database to keep the in & out.

Banking System: For storing customer info, tracking day to day credit and debit transactions, generating bank statements etc.

Sales: To store customer information, production information and invoice details.

Airlines: To travel though airlines, we make early reservations; this reservation information along with flight schedule is stored in database.

Education sector: Database systems are frequently used in schools and colleges to store and retrieve the data regarding student details, staff details, course details, exam details, payroll data, attendance details, fees details etc.



**Purpose of Database System:**

- The main purpose of database systems is to manage the data.
- Consider a university that keeps the data of students, teachers, courses, books etc.
- To manage this data we need to store this data somewhere where we can add new data, delete unused data, update outdated data, retrieve data, to perform these operations on data we need a Database management system that allows us to store the data in such a way so that all these operations can be performed on the data efficiently.

**Characteristics of Database System:**

- Data stored into Tables: Data is never directly stored into the database. Data is stored into tables, created inside the database.

- Reduced Redundancy: In the modern world hard drives are very cheap, but earlier when hard drives were too expensive, unnecessary repetition of data in database was a big problem. But DBMS follows Normalization which divides the data in such a way that repetition is minimum.

- Data Consistency: On Live data, i.e. data that is being continuously updated and added, maintaining the consistency of data can become a challenge. But DBMS handles it all by itself.

- Support Multiple user and Concurrent Access: DBMS allows multiple users to work on it(update, insert, delete data) at the same time and still manages to maintain the data consistency.

- Query Language: DBMS provides users with a simple Query language, using which data can be easily fetched, inserted, deleted and updated in a database.

**Advantages of DBMS**

- Controls database redundancy: It can control data redundancy because it stores all the data in one single database file and that recorded data in database.
- Data sharing: In DBMS, the authorized users of an organization can share the data among multiple users.
- Easily Maintenance: It can be easily maintainable due to the centralized nature of the database system.
- Reduce time: It reduces development time and maintenance need.
- Backup: It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
- Multiple user interface: It provides different types of user interfaces like GUI, application program interfaces



**Disadvantages of DBMS**

- Cost of Hardware and Software: It requires a high speed of data processor and large memory size to run DBMS software.
- Size: It occupies a large space of disks and large memory to run them efficiently.
- Complexity: Database system creates additional complexity and requirements.
- Higher impact of failure: Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

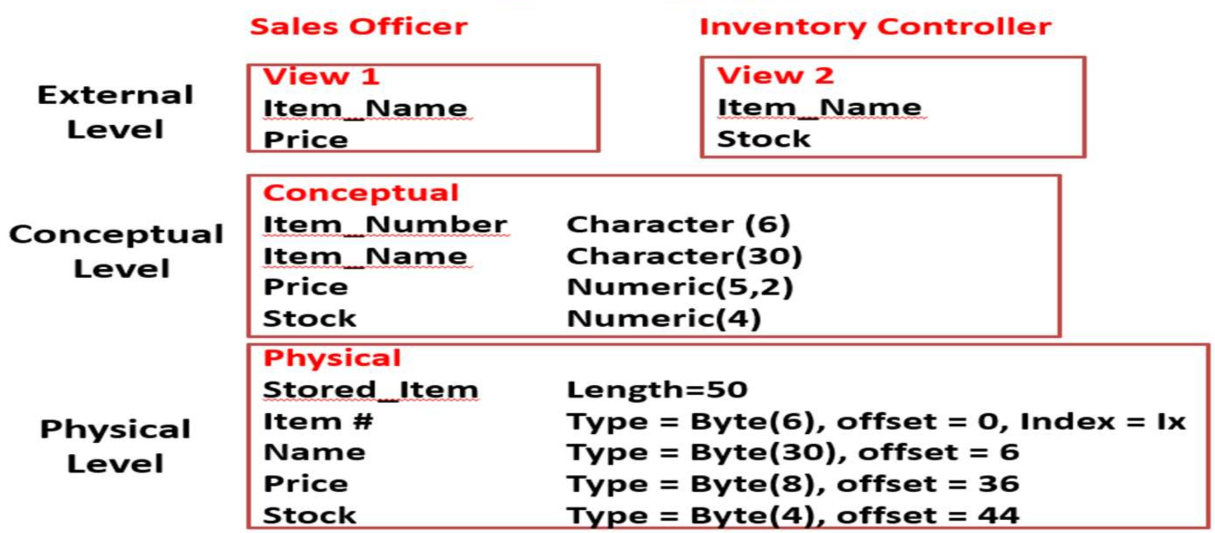
## DBMS ARCHITECTURE:

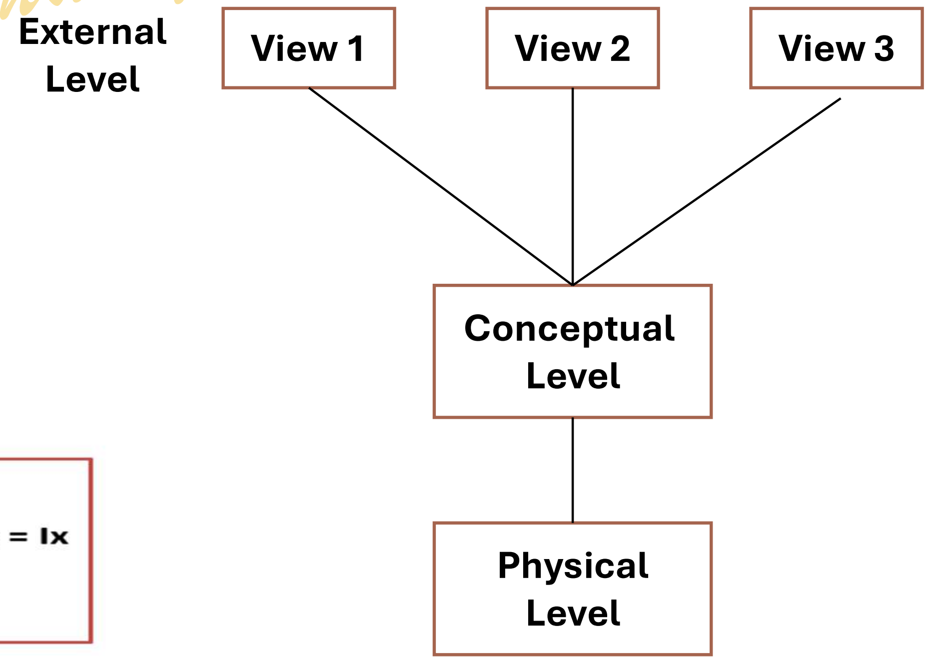
- **Logical DBMS architecture:** The logical architecture deals with the way data is stored and presented to users.
- **Physical DBMS architecture:** The physical architecture is concerned with the s/w components that make up a DBMS.

**THREE LEVEL ARCHITECTURE OF DBMS:** The objective of the three-level architecture is **to separate each user's view of the database from the way the database is physically represented**. There are three levels.

1. External or View Level
2. Conceptual Level
3. Internal or Physical Level





## External Level or View level:

- It is the **users' view** of the database.
- This level describes **that part of the database that is relevant** to each user.
- External level is the one which is **closest to the end users.**
- This level deals with the way in which individual users view data.
- A view involves only those portions of a database which are **of concern to a user.**
- The external view **insulates users from the details of the internal and conceptual levels.**
- External level is also known as the **view level.**
- In addition different views may have **different representations of the same data.** For example, one user may view dates in the **form (day, month, year), while another may view dates as (year, month, day).**

## Conceptual Level or Logical level:

- The **middle level** in the three-level architecture is the **conceptual level**.
- This level describes **what data is stored** in the database and the **relationships** among the data.
- This level contains the **logical structure of the entire database** as seen by the DBA.
- The users of this level are not concerned with **how these logical data structures will be implemented at the physical level**, rather they just are concerned about **what information is to be kept in the database.**
- Conceptual level is also known as the, **logical level.**

- The conceptual level represents:
- All entities, their attributes, and their relationships

An **Entity is an object whose information is stored in the database.** For example, in student database the entity is student.

An attribute is a characteristic of interest about an entity.

For example, in case of student database Roll No, Name, Class, Address etc. are attributes of entity student.

- Data types of attributes (for example, integer, real, character) and their length (such as the maximum number of digits or characters),
- The constraints on the data
- Security and integrity information



**Internal Level or Storage Level or Physical Level:**

- It is the physical representation of the database on the computer.
- This level describes how the data is stored in the database.
- The internal level is the one that concerns the way the data are physically stored on the hardware.
- The internal level covers the physical implementation of the database to achieve optimal runtime performance and storage space utilization.
- It covers the data structures and file organizations used to store data on storage devices.
- It interfaces with the operating system access methods to place the data on the storage devices, build the indexes, retrieve the data, and so on.

## Mapping between Views:

**Two** mappings are required in a database system with **three different views**.

1. **External/Conceptual Mapping**
2. **Conceptual/Internal Mapping**

## External/Conceptual Mapping:

- Each external schema is related to the conceptual schema by the external/conceptual mapping.
- A mapping between the external and conceptual views gives the correspondence among the records and the relationships of the external and conceptual views.
- There is a mapping from a particular logical record in the external view to one (or more) conceptual record(s) in the conceptual view.



## Conceptual/Internal Mapping:

- Conceptual view is related to the internal view by the conceptual/internal mapping.
- This enables the DBMS to find the actual record or combination of records in physical storage that constitute a logical record in conceptual view.
- **Mapping between the conceptual and the internal levels specifies the method of deriving the conceptual record from the physical database.**

## Concept of Schema:

- A schema is defined as an **outline or a plan** that describes the records and relationships existing at the particular level.
- The External view is described by means of a schema called **external schema** that correspond to different views of the data.
- The Conceptual view is defined by **conceptual schema,** which describes all the entities, attributes, and relationship together with integrity constraints.
- Internal View is defined by **internal schema**, which is a complete description of the internal model, containing definition of stored records, the methods of representation, the data fields, and the indexes used.
- There is only **one conceptual schema and one internal schema per database**.



**STUDENT**

| Name | Student_number | Class | Major |
|---|---|---|---|

**COURSE**

| Course_name | Course_number | Credit_hours | Department |
|---|---|---|---|

**PREREQUISITE**

| Course_number | Prerequisite_number |
|---|---|

**SECTION**

| Section_identifier | Course_number | Semester | Year | Instructor |
|---|---|---|---|---|

**GRADE_REPORT**

| Student_number | Section_identifier | Grade |
|---|---|---|

- The data in the database at any particular point in time is called a **database instance.**
- Therefore, **many** database instances can correspond to the **same** database schema.
- The **schema** is sometimes called the **intension** of the database, while an **instance is called an extension (or state) of the database.**

## Database Instance:

1. **Empty state:** When a new database is defined, only its schema is specified. At this point, the database is said to be in **empty state** as it contains no data.
2. **Initial state:** When the database is loaded with data for the first time, it is said to be in initial state.
3. **Current state:** The data in the database is updated frequently. Thus, **at any point of time, the database is said to be in the current state.**



The DBMS is responsible to check whether the database state is valid state. **Thus, each time the database is updated, DBMS ensures that the database remains in the valid state.** The DBMS refers to DBMS catalog where the metadata is stored in order to check whether the **database state satisfies the structure and constraints specified in the schema.**

**Data Abstraction in DBMS**

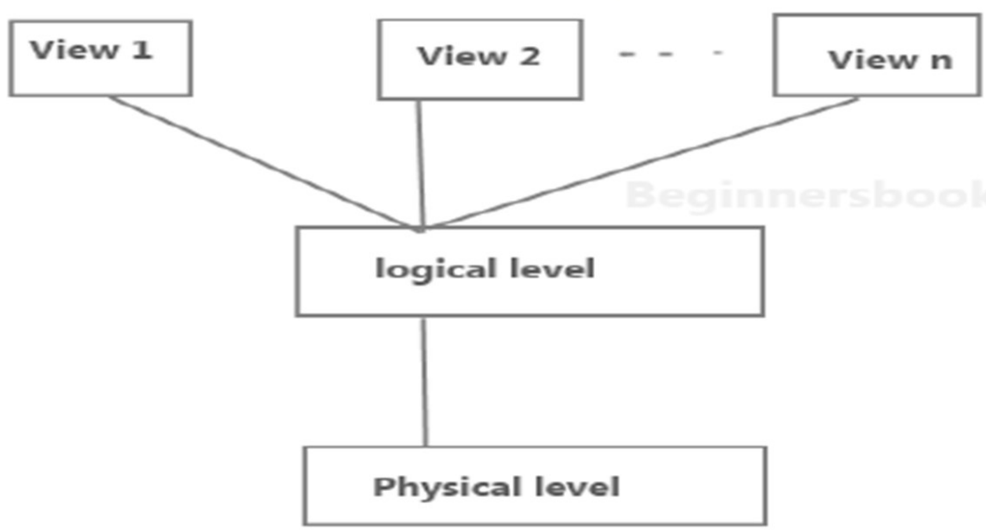
Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called data abstraction.

We have three levels of abstraction:

**Physical level**: This is the lowest level of data abstraction. It describes how data is actually stored in database. You can get the complex data structure details at this level.

**Logical level**: This is the middle level of 3-level data abstraction architecture. It describes what data is stored in database.

**View level**: Highest level of data abstraction. This level describes the user interaction with database system.




Three Levels of data abstraction

**DBMS ARCHITECTURE:**

- Database management systems architecture will help us understand the components of database system and the relation among them.

- The architecture of DBMS depends on the computer system on which it runs.

- the basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.

- The client/server architecture consists of many PCs and a workstation which are connected via the network.

- DBMS architecture depends upon how users are connected to the database to get their request done.



**TYPES OF DBMS ARCHITECTURE**

There are three types of DBMS architecture:

1. Single tier architecture
2. Two tier architecture
3. Three tier architecture

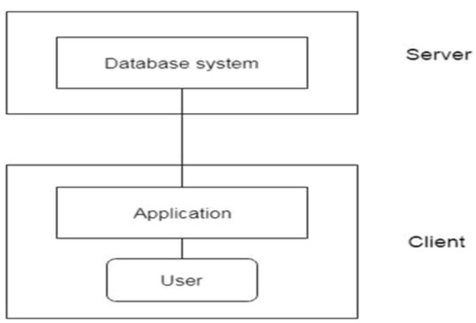
## One Tier Architecture

- In this type of architecture, the database is readily available on the client machine, any request made by client doesn't require a network connection to perform the action on the database.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.



## Two tier architecture

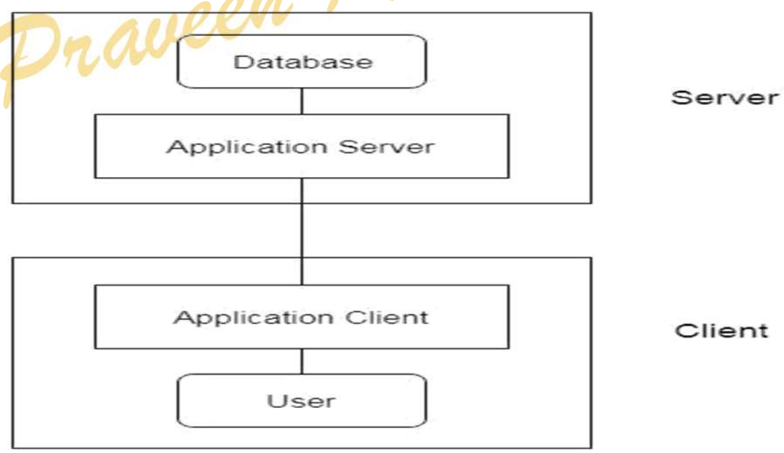
- In two-tier architecture, the Database system is present at the server machine and the DBMS application is present at the client machine, these two machines are connected with each other through a reliable network.
- Whenever client machine makes a request to access the database present at server using a query language like sql, the server perform the request on the database and returns the result back to the client.
- The application connection interface such as JDBC, ODBC are used for the interaction between server and client.



**Three Tier Architecture**

- In three-tier architecture, another layer is present between the client machine and server machine.
- In this architecture, the client application doesn't communicate directly with the database systems present at the server machine, rather the client application communicates with server application and the server application internally communicates with the database system present at the server.





**DATA MODELS:**

- Data Model is the modelling of the data description, data semantics, and consistency constraints of the data.
- It provides the conceptual tools for describing the design of a database at each level of data abstraction.
- Therefore, there are following four data models used for understanding the structure of the database:

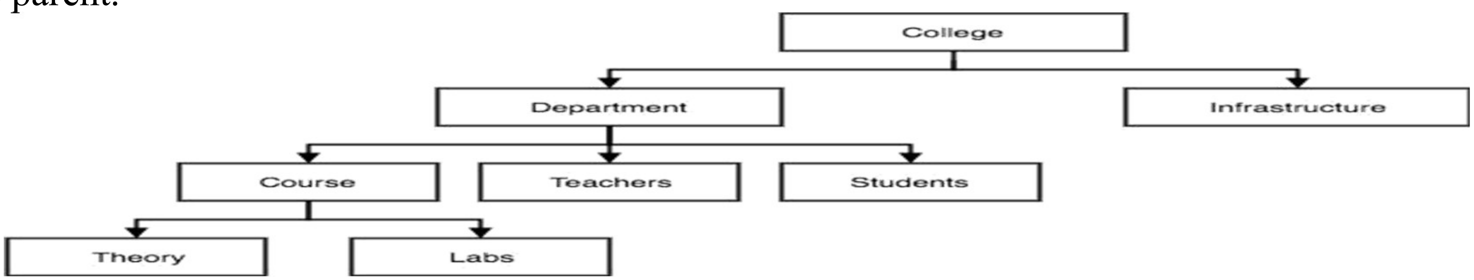
**Four Types of DBMS systems are:**

- Hierarchical database
- Network database
- Relational database
- ER model database



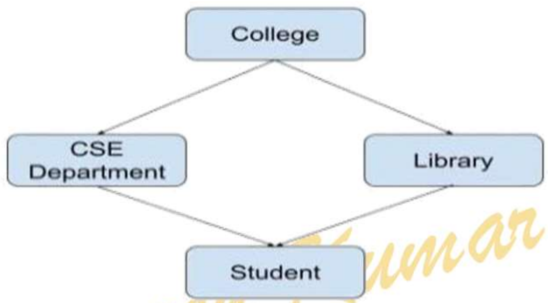
**Hierarchical DBMS**

In a Hierarchical database, model data is organized in a tree-like structure. Data is Stored Hierarchically (top down or bottom up) format. Data is represented using a parent-child relationship. In Hierarchical DBMS parent may have many children, but children have only one parent.



## Network Model

The network database model allows each child to have multiple parents. It helps you to address the need to model more complex relationships like as the orders/parts many-to-many relationship. In this model, entities are organized in a graph which can be accessed through several paths.




*Network Model*



## Relational model

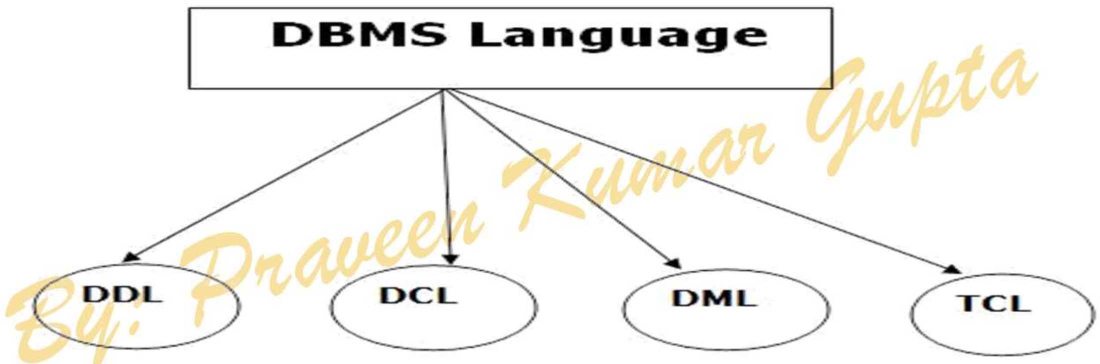
Relational DBMS is the most widely used DBMS model because it is one of the easiest. This model is based on normalizing data in the rows and columns of the tables. Relational model stored in fixed structures and manipulated using SQL.

## Entity-Relationship Model

Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them. While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes and constraints.

**Database languages**

Database languages are used to read, update and store data in a database. There are several such languages that can be used for this purpose; one of them is SQL (Structured Query Language).




1. DDL – Data Definition Language: (CREATE,DROP,ALTER,TRUNCATE,COMMENT,RENAME)
2. DML – Data Manipulation Language: (INSERT, UPDATE,DELETE)
3. DCL – Data Control Language: (GRANT,REVOKE)
4. TCL-Transaction Control Language: (COMMIT,ROLLBACK)

## Data Independence

The ability to modify a schema definition in one level without affecting a scheme definition in the next higher level is called *DATA INDEPENDENCE.*

1. Physical Data Independence
2. Logical Data Independence

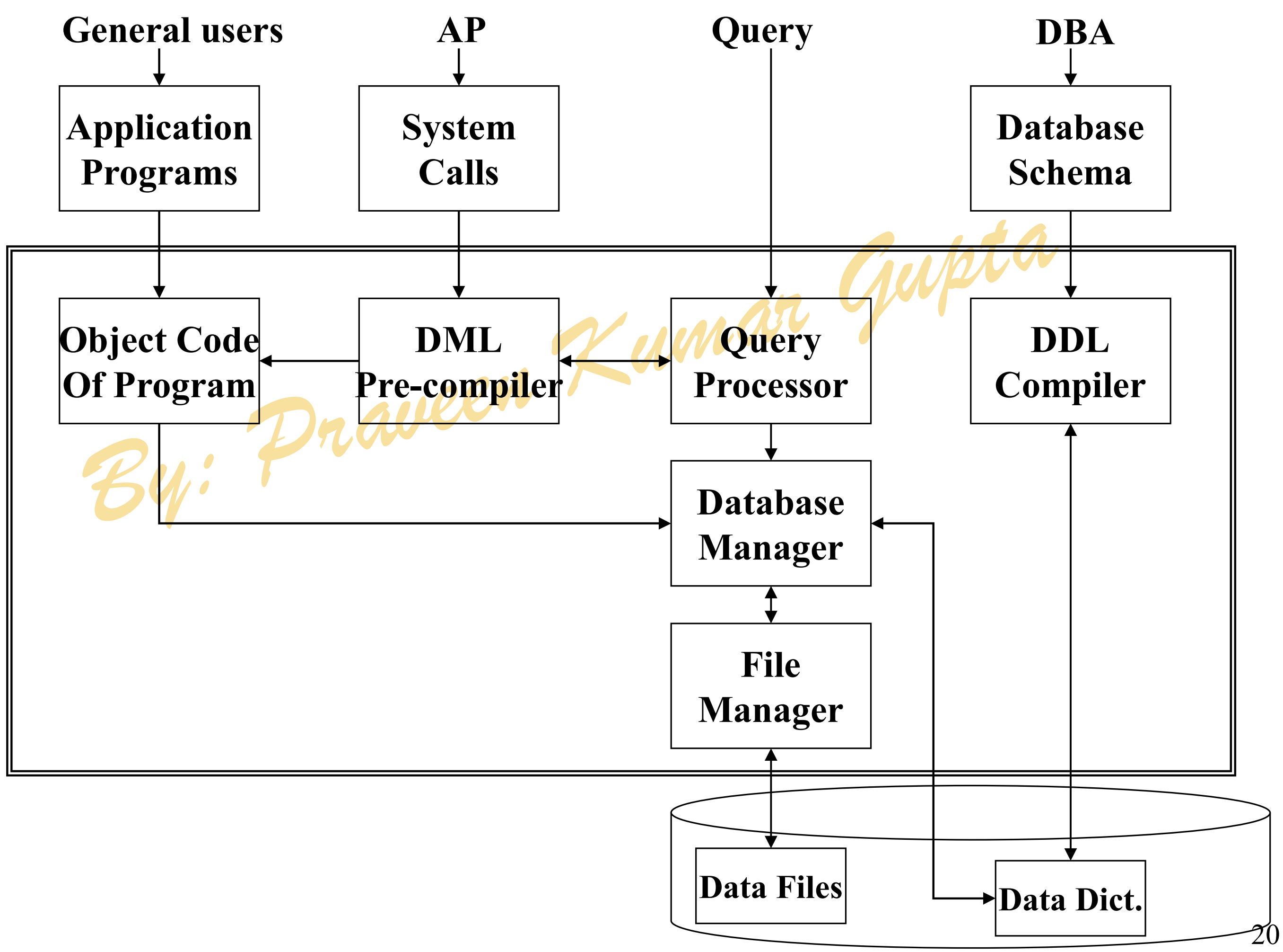
## Physical Data Independence

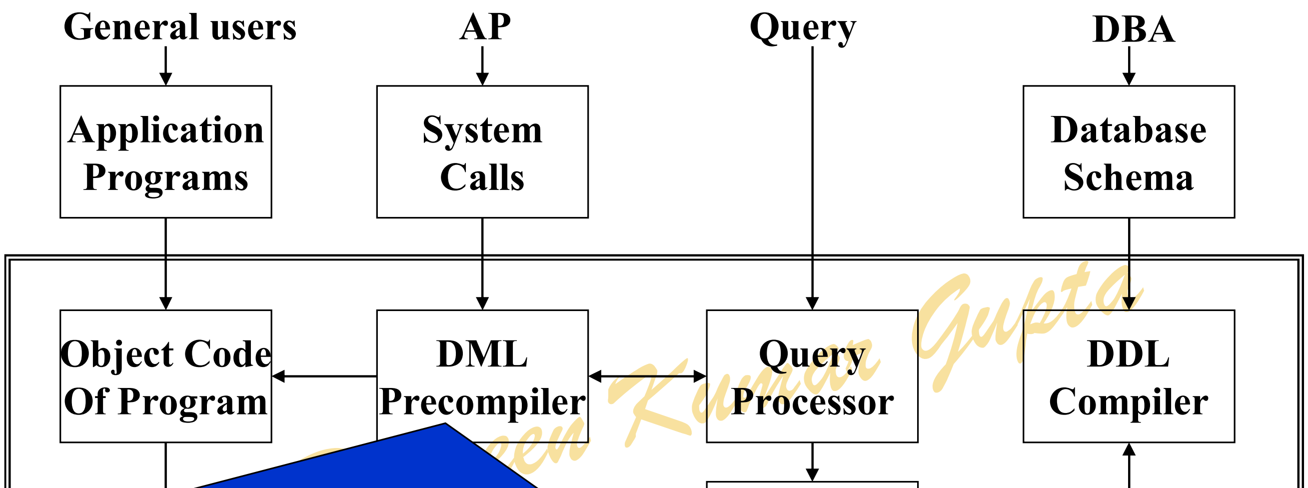
- It refers to **the ability to modify the scheme followed at the physical level without affecting the scheme followed at the conceptual level.**
- The application programs remain the same even though the scheme at the physical level gets modified.
- Modifications at the physical level are occasionally necessary in order to improve the performance.

## Logical Data Independence

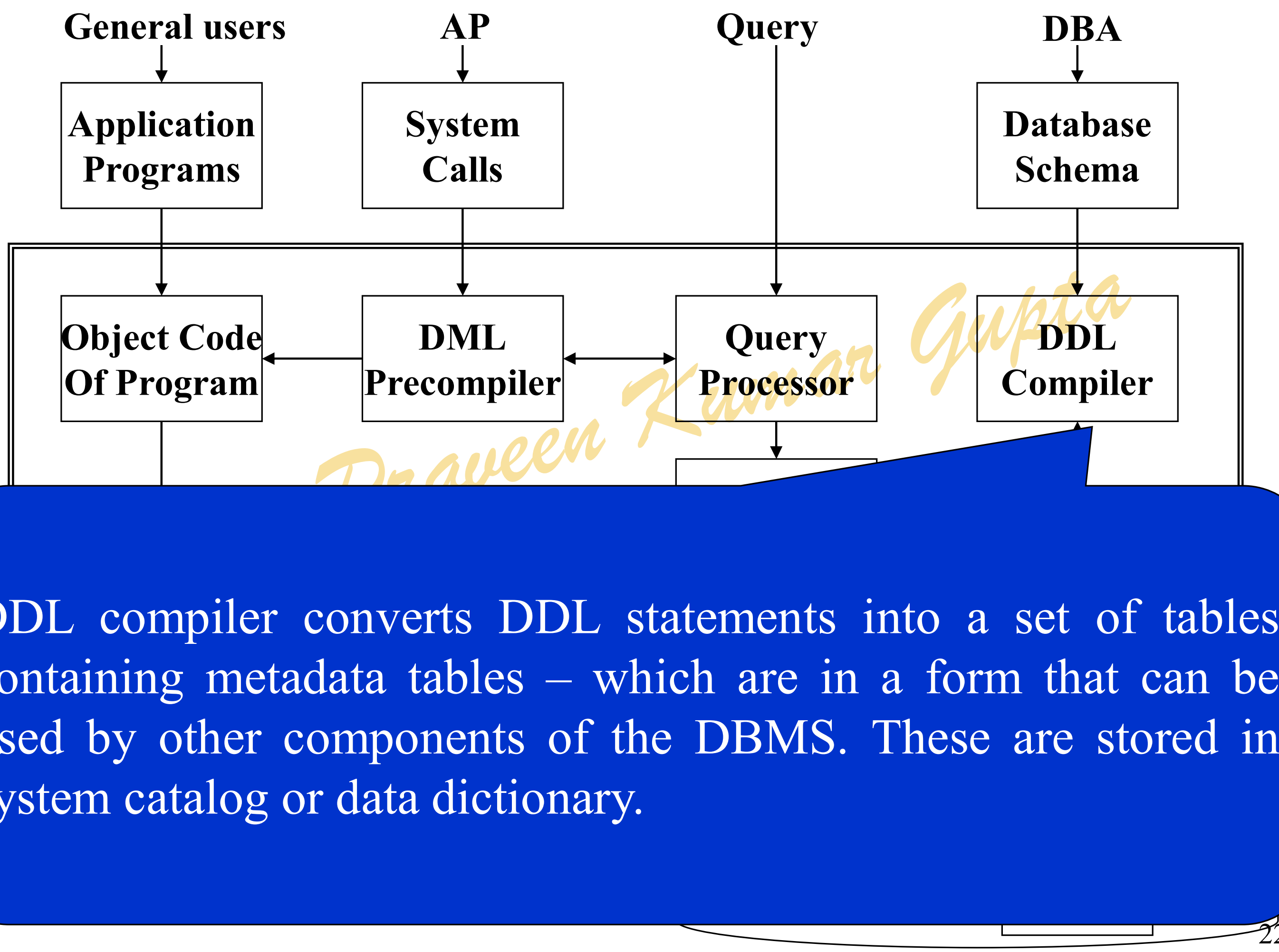
- **It refers to the ability to modify the conceptual scheme without causing any changes in the schemes followed at view levels.**
- The logical data independence ensures that the application programs remain the same.
- **Modifications at the conceptual level are necessary whenever logical structures of the database get altered** because of some unavoidable reasons.

## Physical DBMS Architecture

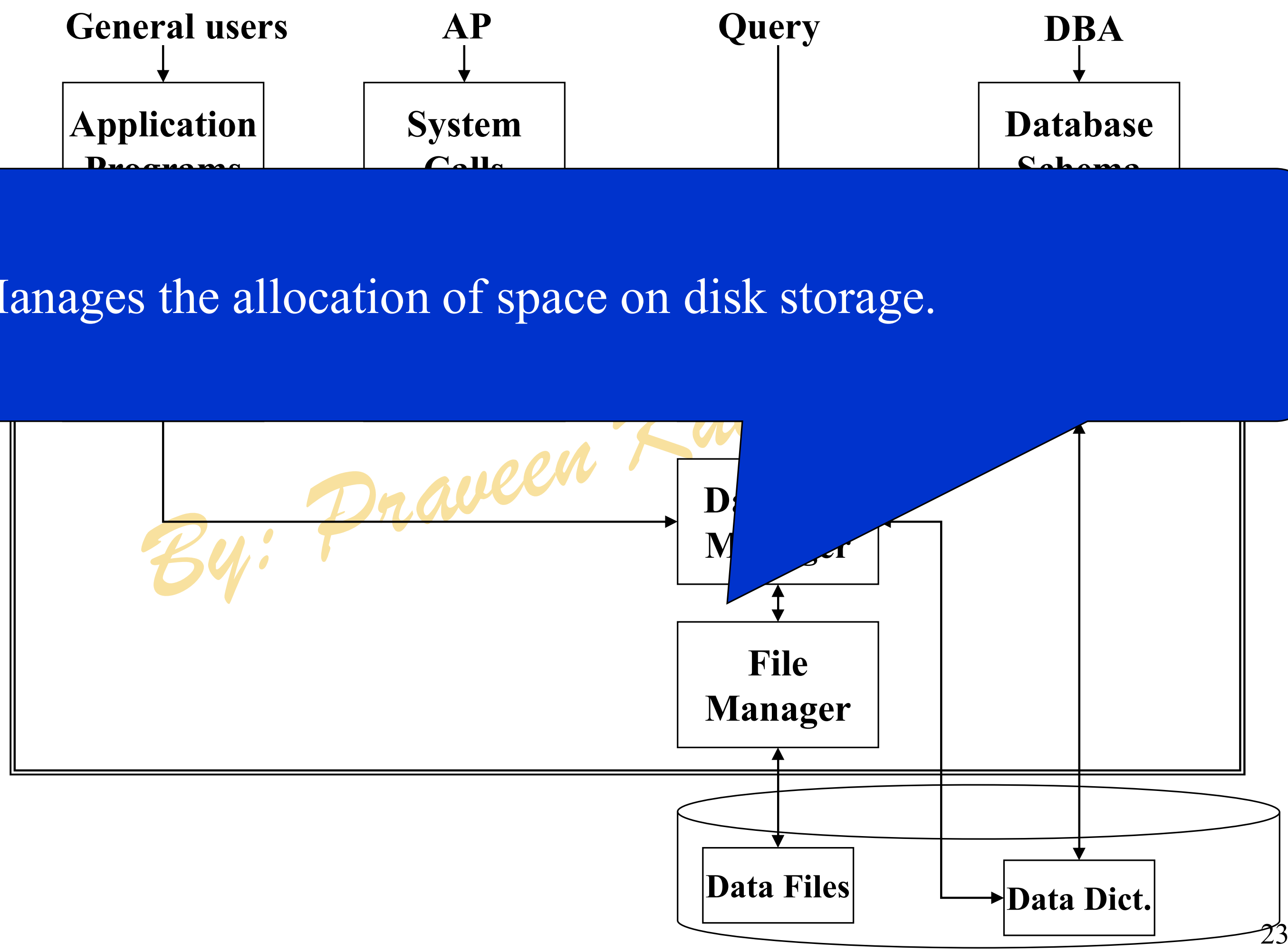




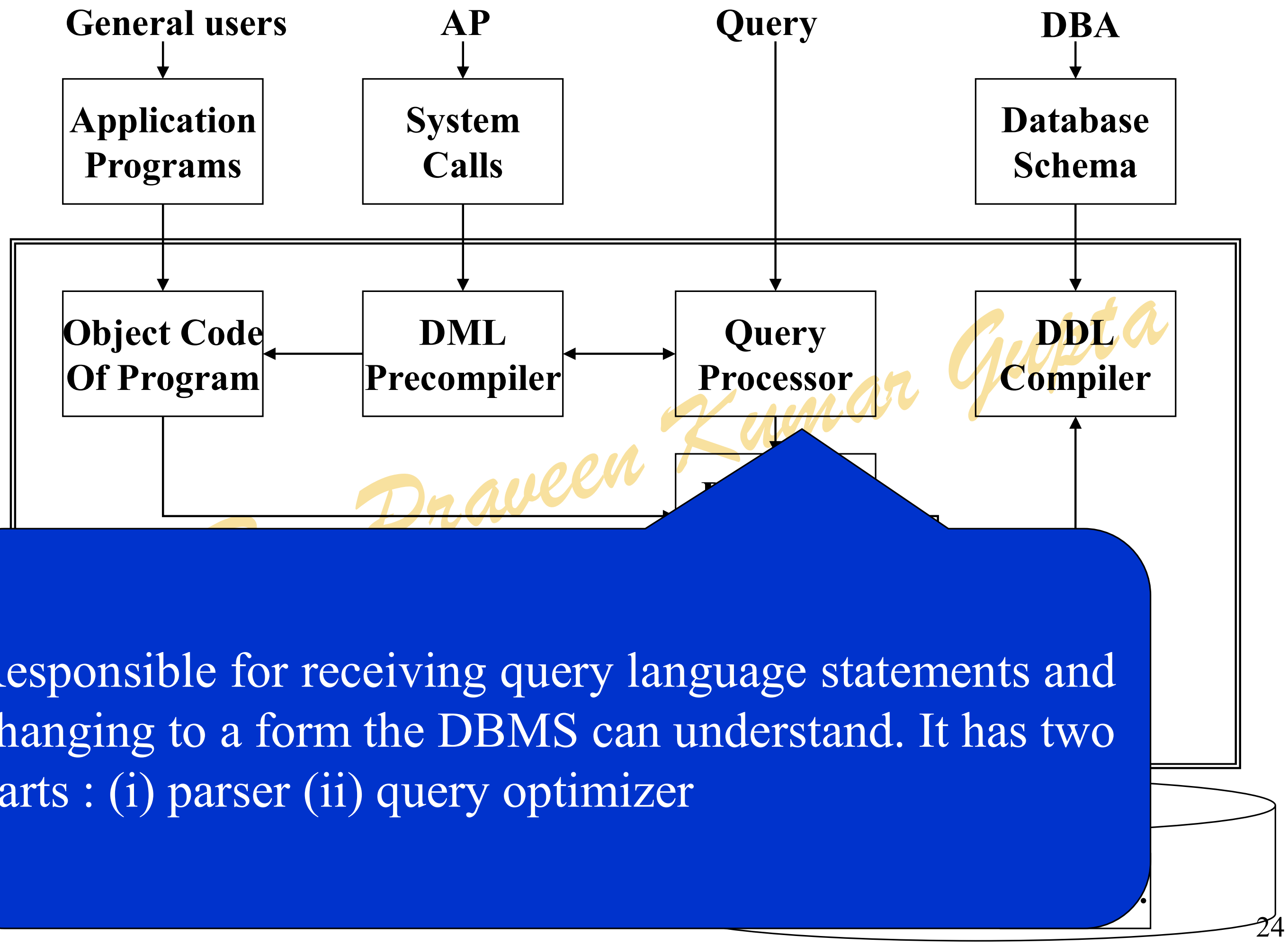

DDL – set of commands required to define the format of data.
DML – set of commands that modify, process data.
DML precompiler converts DML statements embedded in an application program to normal procedural calls in the host language. It interacts with the query processor in order to generate the appropriate code.




DDL compiler converts DDL statements into a set of tables containing metadata tables – which are in a form that can be used by other components of the DBMS. These are stored in system catalog or data dictionary.

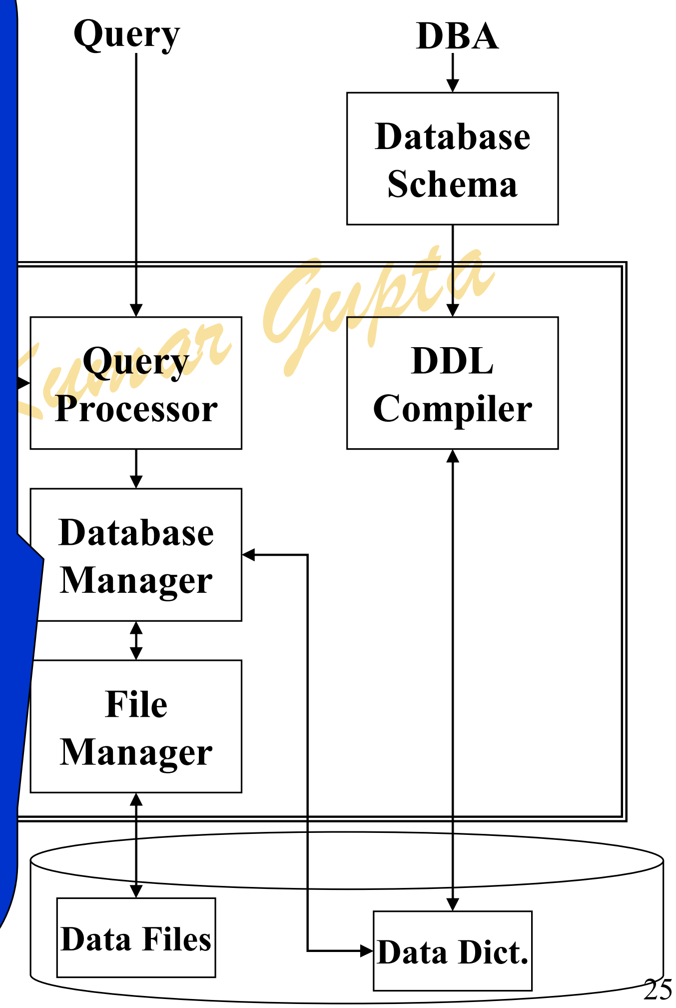



Manages the allocation of space on disk storage.



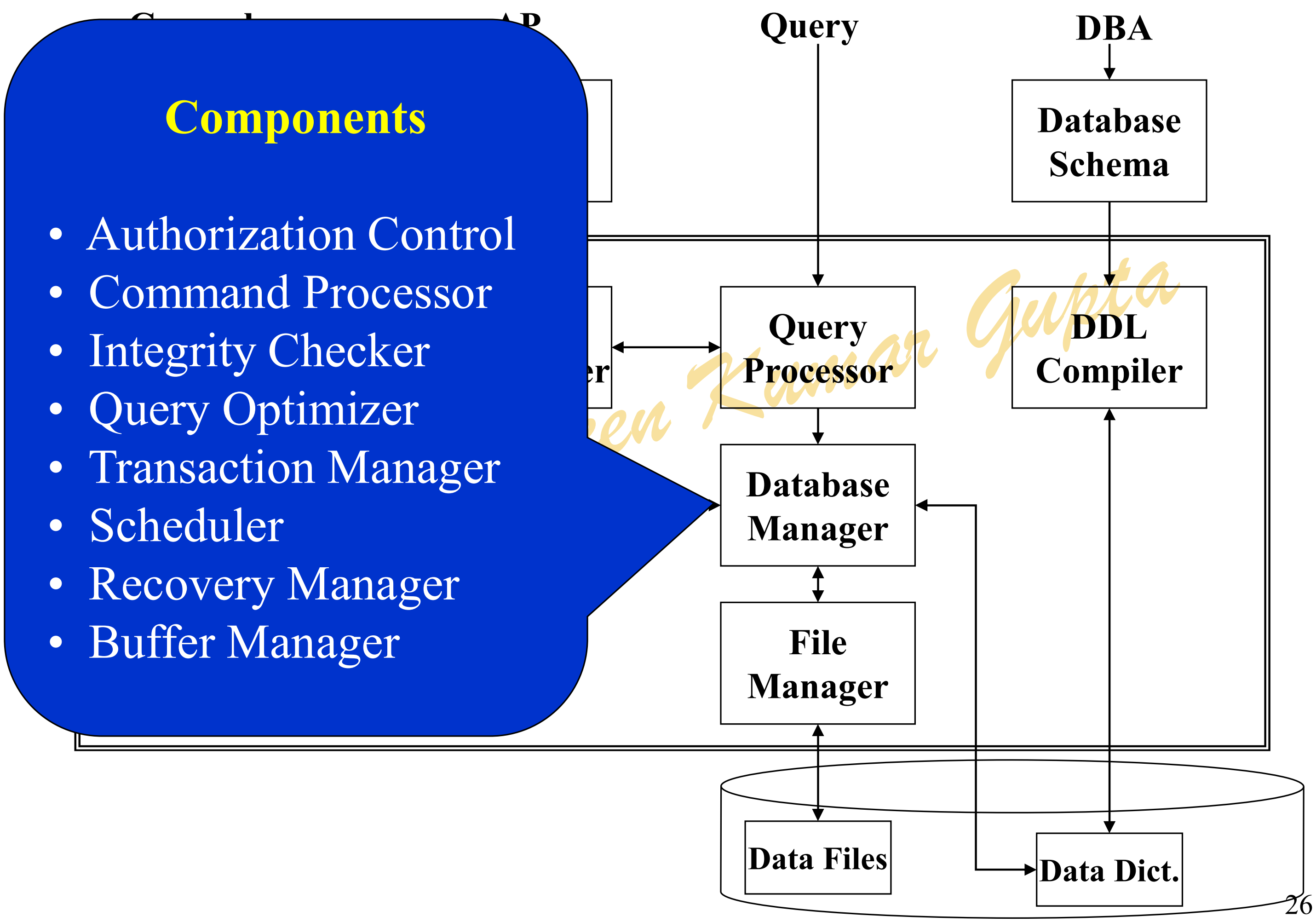

Responsible for receiving query language statements and changing to a form the DBMS can understand. It has two parts : (i) parser (ii) query optimizer

It is the interface b/w low-level data, application programs and queries. It enforces constraints to maintain the consistency and integrity of the data as well as its security. It synchronizes the concurrent access. It also perform backup and recovery operations.

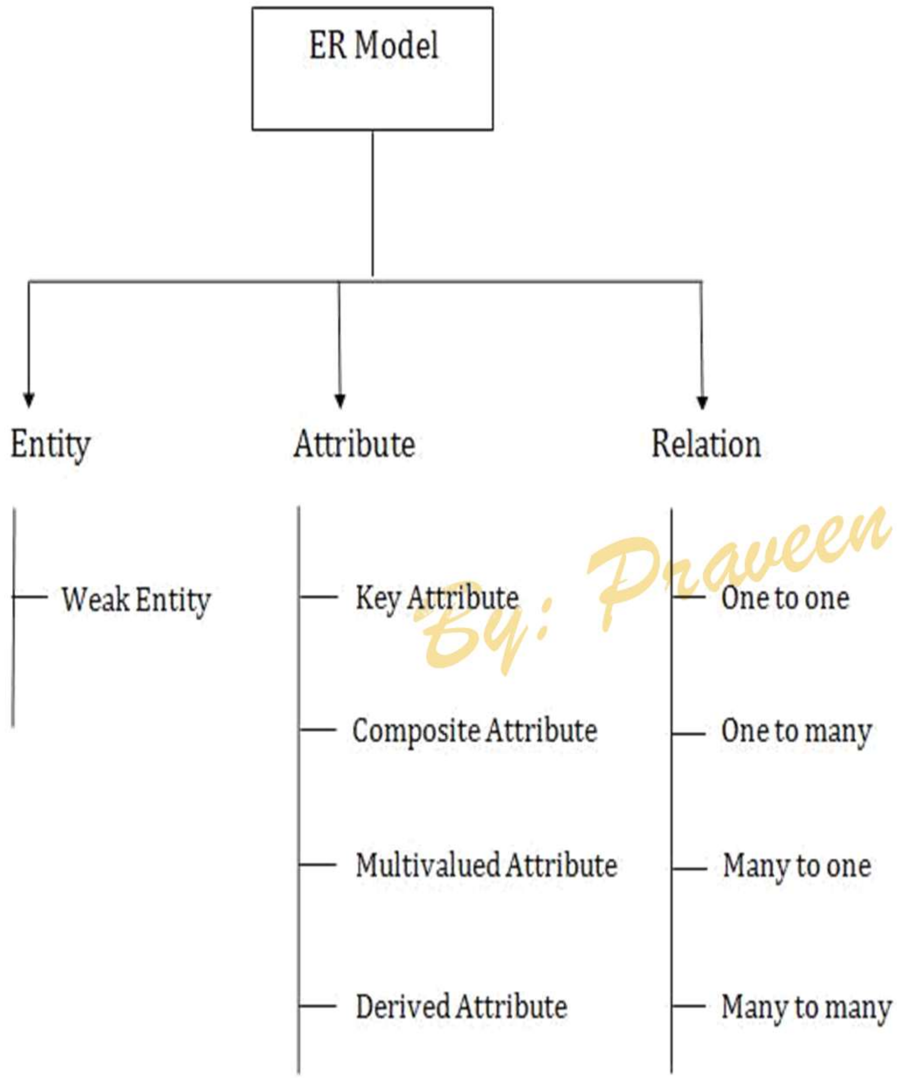


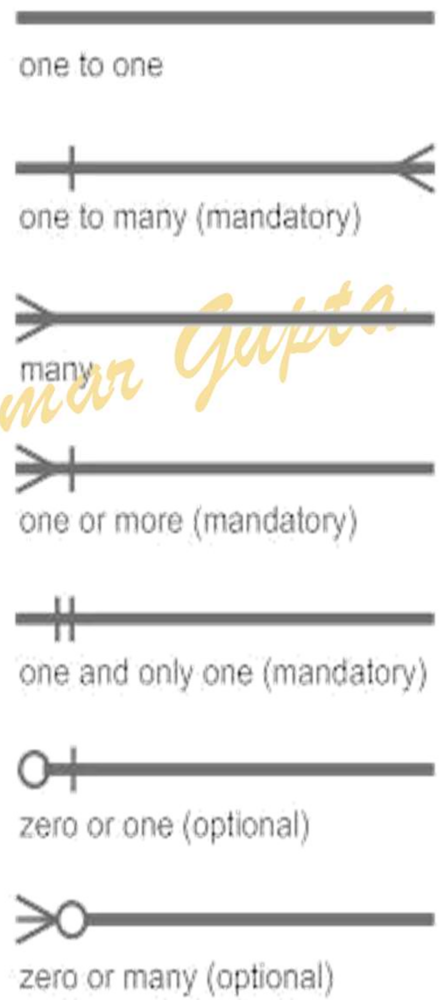
## Components

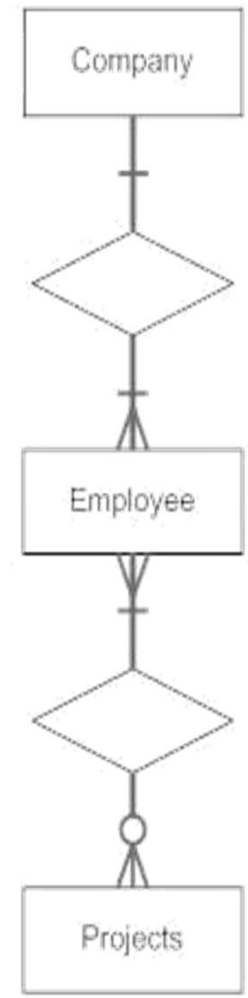
- Authorization Control
- Command Processor
- Integrity Checker
- Query Optimizer
- Transaction Manager
- Scheduler
- Recovery Manager
- Buffer Manager



- Authorization Control: Checks that the user has necessary authorization to carry out the required function.
- Command Processor: Converts commands to a logical sequence of steps.
- Integrity Checker: Checks the requested operation satisfies all necessary integrity constraints such as key constraints.
- Query Optimizer: Examines the query language statements and tries to choose the best and most efficient way to executing the query. Factors – CPU time, disk time, network time, sorting methods and scanning methods.
- Transaction Manager: The transaction manager maintains tables of authorization concurrency.
- Scheduler: It controls the relative order in which transaction operations are executed.
- Recovery Manager: Ensures that the database remains in a consistent state in the presence of failures. Responsible for transaction commit and abort.
- Buffer Manager: Responsible for the transfer of data between main memory and secondary storage.


ER Model
Entity
Attribute
Relation
Weak Entity
Key Attribute
Composite Attribute
Multivalued Attribute
Derived Attribute
One to one
One to many
Many to one
Many to many

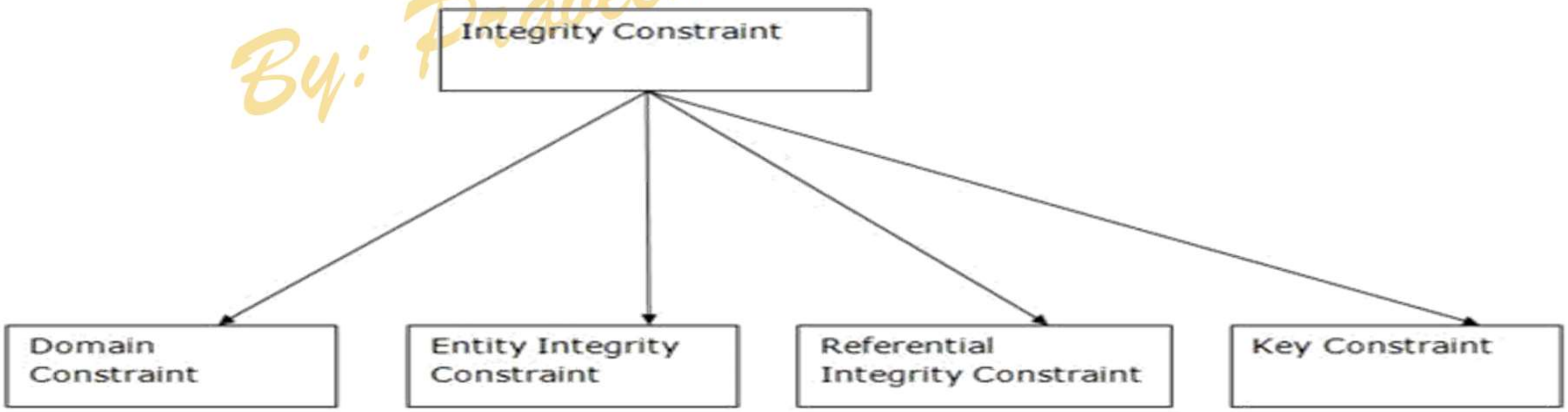

one to one
one to many (mandatory)
many
one or more (mandatory)
one and only one (mandatory)
zero or one (optional)
zero or many (optional)


Company
Employee
Projects

Integrity Constraints

- Integrity constraints are a set of rules. It is used to maintain the quality of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to guard against accidental damage to the database.





## Domain constraints

- Domain constraints can be defined as the definition of a valid set of values for an attribute.
- The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

**Example:**

| ID | NAME | SEMENSTER | AGE |
|---|---|---|---|
| 1000 | Tom | 1st | 17 |
| 1001 | Johnson | 2nd | 24 |
| 1002 | Leonardo | 5th | 21 |
| 1003 | Kate | 3rd | 19 |
| 1004 | Morgan | 8th | A |

Not allowed. Because AGE is an integer attribute

**EMPLOYEE**

| EMP_ID | EMP_NAME | SALARY |
|---|---|---|
| 123 | Jack | 30000 |
| 142 | Harry | 60000 |
| 164 | John | 20000 |
| | Jackson | 27000 |

Not allowed as primary key can't contain a NULL value

## Entity integrity constraints

- The entity integrity constraint states that primary key value can't be null.
- This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
- A table can contain a null value other than the primary key field.

### Referential Integrity Constraints

- A referential integrity constraint is specified between two tables.
- In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.

(Table 1)

| EMP_NAME | NAME | AGE | D_No |
|---|---|---|---|
| 1 | Jack | 20 | 11 |
| 2 | Harry | 40 | 24 |
| 3 | John | 27 | 18 |
| 4 | Devil | 38 | 13 |

Foreign key

Not allowed as D_No 18 is not defined as a Primary key of table 2 and In table 1, D_No is a foreign key defined

Relationships

(Table 2)

Primary Key

| D_No | D_Location |
|---|---|
| 11 | Mumbai |
| 24 | Delhi |
| 13 | Noida |

| ID | NAME | SEMENSTER | AGE |
|---|---|---|---|
| 1000 | Tom | 1st | 17 |
| 1001 | Johnson | 2nd | 24 |
| 1002 | Leonardo | 5th | 21 |
| 1003 | Kate | 3rd | 19 |
| 1002 | Morgan | 8th | 22 |

Not allowed. Because all row must be unique

### Key constraints

- Keys are the entity set that is used to identify an entity within its entity set uniquely.
- An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.

**Key and Types of Keys:**

- Keys help you to identify any row of data in a table.
- In a real-world application, a table could contain thousands of records. Moreover, the records could be duplicated.
- Keys ensure that you can uniquely identify a table record despite these challenges.
- Allows you to establish a relationship between and identify the relation between tables.
- Help you to enforce identity and integrity in the relationship.

**Super Key -** A super key is a group of single or multiple keys which identifies rows in a table.

**Primary Key -** is a column or group of columns in a table that uniquely identify every row in that[[[[ table.

**Candidate Key -** is a set of attributes that uniquely identify tuples in a table.

Candidate Key is a super key with no repeated attributes.

**Alternate Key -** is a column or group of columns in a table that uniquely identify every row in that table.

**Foreign Key -** is a column that creates a relationship between two tables. The purpose of Foreign keys is to maintain data integrity and allow navigation between two different instances of an entity.

**Compound Key -** has two or more attributes that allow you to uniquely recognize a specific record. It is possible that each column may not be unique by itself within the database.

**Composite Key -** An artificial key which aims to uniquely identify each record is called a surrogate key. These kind of key are unique because they are created when you don't have any natural primary key.

**Surrogate Key -** An artificial key which aims to uniquely identify each record is called a surrogate key. These kind of key are unique because they are created when you don't have any natural primary key.