# UNIT -I

## INTRODUCTION TO DBMS:

### What is data?

- Data is nothing but facts and statistics stored or free flowing over a network, generally it's raw and unprocessed.
- Data becomes information when it is processed, turning it into something meaningful.
- What is database: The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently.
- It is also used to organize the data in the form of a table, schema, views, and reports, etc.
- Using the database, you can easily retrieve, insert, and delete the information.
- For example: The college Database organizes the data about the admin, staff, students and faculty etc.

### What is dbms?

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

- A DBMS is software that allows creation, definition and manipulation of database, allowing users to store, process and analyse data easily.
- DBMS provides us with an interface or a tool, to perform various operations like creating database, storing data in it, updating data, creating tables in the database and a lot more.
- DBMS also provides protection and security to the databases.
- It also maintains data consistency in case of multiple users.

Here are some examples of popular DBMS used these days:

- MySql
- Oracle
- SQL Server
- IBM DB2

**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 records of ins and outs
- 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 SYSTEMS**

- 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 DBMS**

- 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 Normalisation which divides the data in such a way that repetition is minimum.
- Data Consistency: On Live data, i.e. data that is being continuosly 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 is placed in the 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 graphical user interfaces, 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.

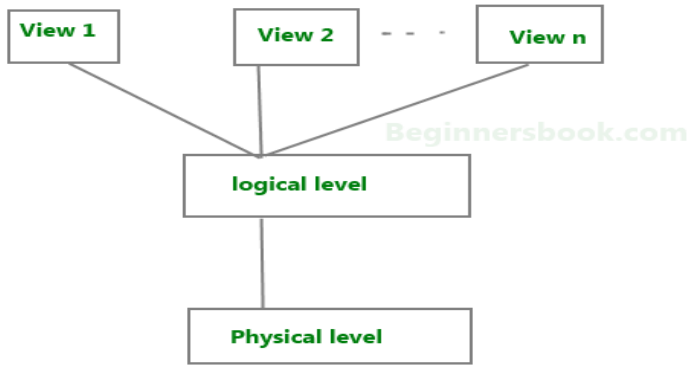
**View of Data in DBMS**

- Abstraction is one of the main features of database systems.
- Hiding irrelevant details from user and providing abstract view of data to users, helps in easy and efficient user-database interaction.
- the three level of DBMS architecture, The top level of that architecture is “view level”. The view level provides the “view of data” to the users and hides the irrelevant

details such as data relationship, database schema, constraints, security etc from the user.

**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.




Three Levels of 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.

**Instance and schema in DBMS**

- Definition of schema: Design of a database is called the schema. Schema is of three types: Physical schema, logical schema and view schema.

- The design of a database at physical level is called physical schema, how the data stored in blocks of storage is described at this level.
- Design of database at logical level is called logical schema, programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures, however the internal details such as implementation of data structure is hidden at this level (available at physical level).
- Design of database at view level is called view schema. This generally describes end user interaction with database systems.

**Definition of instance**:

The data stored in database at a particular moment of time is called instance of database. Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

**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:

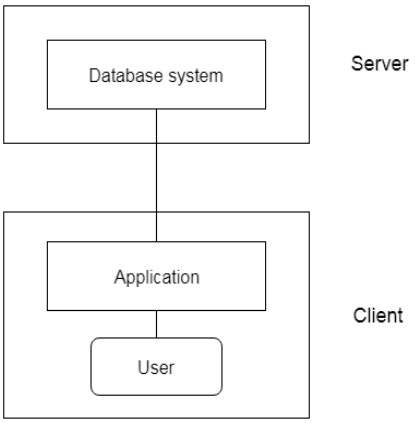
1. Single tier architecture
2. Two tier architecture
3. Three tier architecture

**1-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.

**2. Two tier architecture**

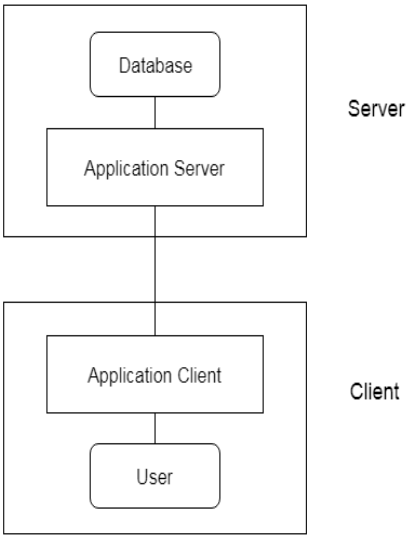
- In two-tier architecture, the Database system is present at the server machine and the DBMS application is present at the client machine, these two machines are connected with each other through a reliable network.
- Whenever client machine makes a request to access the database present at server using 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.




**3-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 modeling 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:

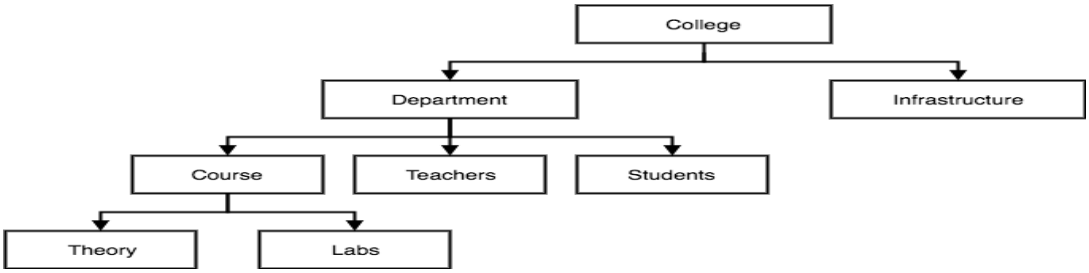
**Four Types of DBMS systems are:**

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

**Hierarchical DBMS**

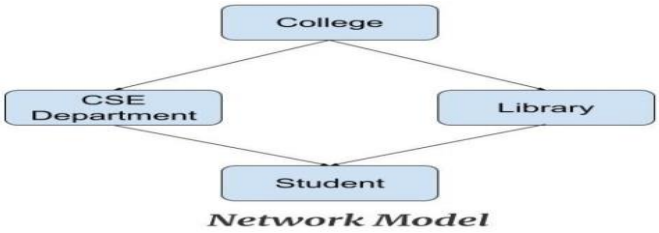
In a Hierarchical database, model data is organized in a tree-like structure. Data is Stored Hierarchically (top down or bottom up) format. Data is represented using a parent-child

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.




**Relational model**

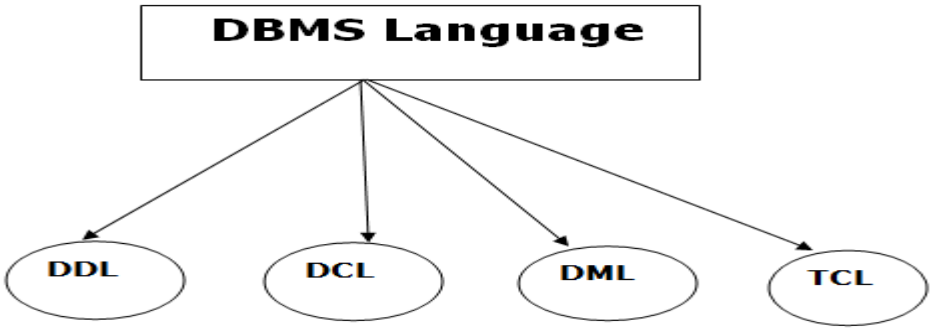
Relational DBMS is the most widely used DBMS model because it is one of the easiest. This model is based on normalizing data in the rows and columns of the tables. Relational model stored in fixed structures and manipulated using SQL.

**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.

**DBMS 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).




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

**1. DDL(Data Definition Language)** : DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.

**CREATE** – it is used to create the database or its objects (like table, index, function, views, store procedure and triggers).

There are two CREATE statements available in SQL:

- CREATE DATABASE
- CREATE TABLE

**CREATE DATABASE**

A Database is defined as a structured set of data. So, in SQL the very first step to store the data in a well structured manner is to create a database. The CREATE DATABASE statement is used to create a new database in SQL.

Syntax:

**CREATE DATABASE database_name;**

Example:

SQL> CREATE DATABASE Employee;

In order to get the list of all the databases, you can use SHOW DATABASES statement.

Example –

SQL> SHOW DATABASES;

**CREATE TABLE:**

The CREATE TABLE statement is used to create a table in SQL. We know that a table comprises of rows and columns. So while creating tables we have to provide all the information to SQL about the names of the columns, type of data to be stored in columns, size of the data etc. Let us now dive into details on how to use CREATE TABLE statement to create tables in SQL.

Syntax:

```
CREATE TABLE table_name
(
column1 data_type(size),
column2 data_type(size),
column3 data_type(size),
....
);
```

**Example Query:**

This query will create a table named Students with three columns, ROLL_NO, NAME and SUBJECT.

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
);
```

**DROP:**

DROP is used to delete a whole database or just a table.The DROP statement destroys the objects like an existing database, table, index, or view.

A DROP statement in SQL removes a component from a relational database management system (RDBMS).

Syntax:

DROP object object_name;

Examples:

DROP TABLE table_name;

table_name: Name of the table to be deleted.

DROP DATABASE database_name;

database_name: Name of the database to be deleted.

**TRUNCATE**

It is used to remove all records from a table, including all spaces

The TRUNCATE TABLE mytable statement is logically (though not physically) equivalent to the DELETE FROM mytable statement (without a WHERE clause).

Syntax:

TRUNCATE TABLE table_name;

**DROP vs TRUNCATE**

- Truncate is normally ultra-fast and its ideal for deleting data from a temporary table.
- Truncate preserves the structure of the table for future use, unlike drop table where the table is deleted with its full structure.
- Table or Database deletion using DROP statement cannot be rolled back, so it must be used wisely.

To delete the whole database

DROP DATABASE student_data;

After running the above query whole database will be deleted.

To truncate Student_details table from student_data database.

TRUNCATE TABLE Student_details;

**ALTER (ADD, DROP, MODIFY)**

ALTER TABLE is used to add, delete/drop or modify columns in the existing table. It is also used to add and drop various constraints on the existing table.

ALTER TABLE – ADD:

ADD is used to add columns into the existing table. Sometimes we may require to add additional information, in that case we do not require to create the whole database again, ADD comes to our rescue.

Syntax:

```
ALTER TABLE table_name
        ADD (Columnname_1  datatype,
        Columnname_2  datatype,
        …
        Columnname_n  datatype);
```

**ALTER TABLE – DROP**

DROP COLUMN is used to drop column in a table. Deleting the unwanted columns from the table.

Syntax:

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

**ALTER TABLE-MODIFY**

It is used to modify the existing columns in a table. Multiple columns can also be modified at once.

```
ALTER TABLE table_name
MODIFY column_name column_type;
```

QUERY:

**To ADD 2 columns AGE and COURSE to table Student.**

```
ALTER TABLE Student ADD (AGE number(3),COURSE varchar(40));
```

**MODIFY column COURSE in table Student**

```
ALTER TABLE Student MODIFY COURSE varchar(20);
```

**Comments**

As is any programming languages comments matter a lot in SQL also. In this set we will learn about writing comments in any SQL snippet.

Comments can be written in the following three formats:

- Single line comments.
- Multi line comments
- In line comments

**DML(Data Manipulation Language)** : The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements.

**SELECT Statement**

select statement is used to fetch data from relational database. A relational database is organized collection of data. As we know that data is stored inside tables in a database. SQL select statement or SQL select query is used to fetch data from one or more than one tables.

SELECT Syntax

**One column:**

Here column_name is the name of the column for which we need to fetch data and table_name is the name of the table in the database.

SELECT column_name FROM table_name;

**More than one columns:**

SELECT column_name_1, column_name_2, ... FROM table_name;

**To fetch the entire table or all the fields in the table:**

SELECT * FROM table_name;

Example:

SELECT EMP_NAME FROM EMPLOYEES;

To fetch the entire EMPLOYEES table:

SELECT * FROM EMPLOYEES;

Query to fetch the fields ROLL_NO, NAME, AGE from the table Student:

SELECT ROLL_NO, NAME, AGE FROM Student;

**INSERT INTO Statement**

The INSERT INTO statement of SQL is used to insert a new row in a table. There are two ways of using INSERT INTO statement for inserting rows:

Only values: First method is to specify only the value of data to be inserted without the column names.

INSERT INTO table_name VALUES (value1, value2, value3,…);

Column names and values both: In the second method we will specify both the columns which we want to fill and their corresponding values as shown below:

INSERT INTO table_name (column1, column2, column3,..) VALUES ( value1, value2, value3,..);

**Example:**

Method 1 (Inserting only values) :

INSERT INTO Student VALUES (‘5′,’HARSH’,’WEST BENGAL’,’XXXXXXXXXX’,’19’);

Method 2 (Inserting values in only specified columns):

INSERT INTO Student (ROLL_NO, NAME, Age) VALUES (‘5′,’PRATIK’,’19’);

**UPDATE Statement**

The UPDATE statement in SQL is used to update the data of an existing table in database. We can update single columns as well as multiple columns using UPDATE statement as per our requirement.

Basic Syntax:

UPDATE TableName

SET column_name1 = value, column_name2 = value....

WHERE condition;

EX1:

SQL> UPDATE EMPLOYEES

SET EMP_SALARY = 10000

WHERE EMP_AGE > 25;

EX2;

SQL> UPDATE EMPLOYEES

SET EMP_SALARY = 120000

WHERE EMP_NAME = 'Apoorv';

**DELETE Statement**

The DELETE Statement in SQL is used to delete existing records from a table. We can delete a single record or multiple records depending on the condition we specify in the WHERE clause.

Basic Syntax:

DELETE FROM table_name WHERE some_condition;

Deleting single record: Delete the rows where NAME = ‘Ram’. This will delete only the first row.

DELETE FROM Student WHERE NAME = 'Ram';

Deleting multiple records: Delete the rows from the table Student where Age is 20. This will delete 2 rows(third row and fifth row).

DELETE FROM Student WHERE Age = 20;

Delete all of the records: There are two queries to do this as shown below,

query1: "DELETE FROM Student";

query2: "DELETE * FROM Student";

**DCL(Data Control Language)** : DCL includes commands such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls of the database system.

Examples of DCL commands:

**GRANT-gives user's access privileges to database.**

**REVOKE-withdraw user's access privileges given by using the GRANT command.**

**TCL(transaction Control Language**) : TCL commands deals with the transaction within the database.

Examples of TCL commands:

**COMMIT– commits a Transaction.**

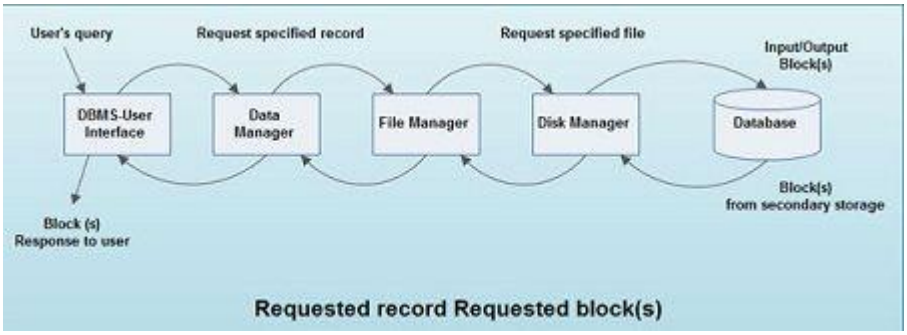
**ROLLBACK– rollbacks a transaction in case of any error occurs.**

**SAVEPOINT–sets a savepoint within a transaction.**

**SET TRANSACTION–specify characteristics for the transaction**.

**What is the Procedure for Database Access?**

- Any access to the stored data is done by the data manager. A user's request for data is-received by the data manager, which detern1ines the physical record required. The decision as 10 which physical record is needed may require some preliminary consultation of the database and/or the data dictionary prior to the access of the actual data itself.
- The data manager sends the request for a specific physical record to the file manager. The file manager decides which physical block of secondary storage devices contains the required record and sends the request for the appropriate block to the disk manager. A block is a unit of physical input/output operations between primary and secondary storage. The disk manager retrieves the block and sends it to the file manager, which sends the required record to the data manager.




**DATA BASE USERS AND ADMINISTRATORS:**

Database users are the persons who interact with the database and take the benefits of database.

They are differentiated into different types based on the way they expect to interact with the system.

Naive users: They are the unsophisticated users who interact with the system by using permanent applications that already exist. Example: Online Library Management System, ATMs (Automated Teller Machine), etc.

Application programmers: They are the computer professionals who interact with system through DML. They write application programs.

Sophisticated users: They interact with the system by writing SQL queries directly through the query processor without writing application programs.

Specialized users: They are also sophisticated users who write specialized database applications that do not fit into the traditional data processing framework. Example: Expert System, Knowledge Based System, etc.

**Database Administrators**

The life cycle of database starts from designing, implementing to administration of it. A database for any kind of requirement needs to be designed perfectly so that it should work without any issues. Once all the design is complete, it needs to be installed. Once this step is complete, users start using the database. The database grows as the data grows in the database. When the database becomes huge, its performance comes down. Also accessing the data from the database becomes challenge. There will be unused memory in database, making the memory inevitably huge. These administration and maintenance of database is taken care

by database Administrator – DBA.

A DBA has many responsibilities. A good performing database is in the hands of DBA. Database Administrators coordinate all the activities of the database system. They have all the permissions.

**Tasks of DBA**

- Creatingtheschema
- Specifying integrity constraints
- Storage structure and access method definition
- Granting permission to other users.
- Monitoring performance
- Routine Maintenance

**Transaction Management?**

- A Database Transaction is a logical unit of processing in a DBMS which entails one or more database access operation. In a nutshell, database transactions represent real-world events of any enterprise.
- All types of database access operation which are held between the beginning and end transaction statements are considered as a single logical transaction in DBMS. During the transaction the database is inconsistent. Only once the database is committed the state is changed from one consistent state to another.

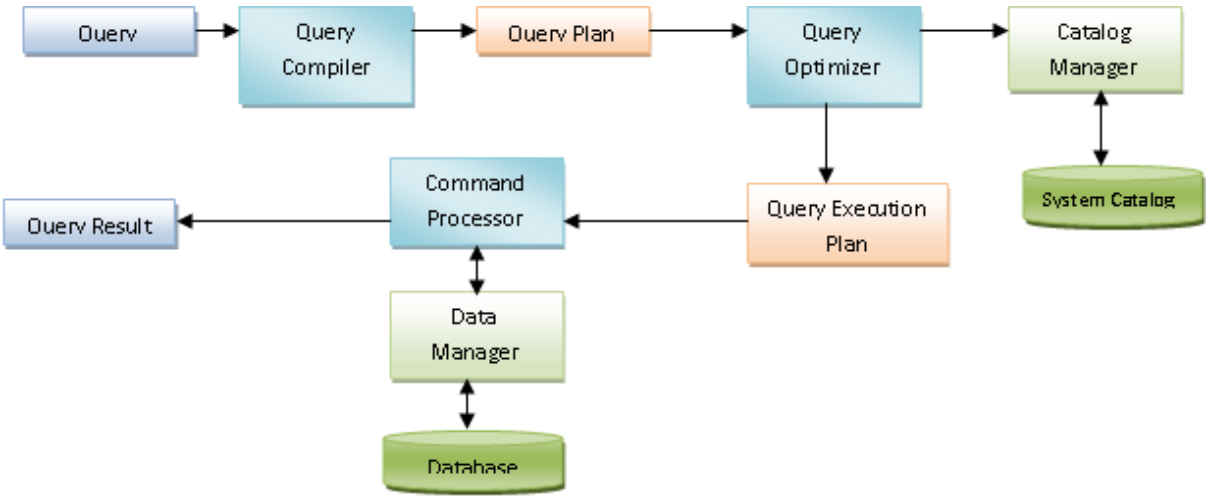
**What are ACID Properties?**

ACID Properties are used for maintaining the integrity of database during transaction processing. ACID in DBMS stands for Atomicity, Consistency, Isolation, and Durability.

- **Atomicity**: A transaction is a single unit of operation. You either execute it entirely or do not execute it at all. There cannot be partial execution.
- **Consistency**: Once the transaction is executed, it should move from one consistent state to another.

## Query Processing in DBMS

A query processor is one of the major components of a relational database or an electronic database in which data is stored in tables of rows and columns. It complements the storage engine, which writes and reads data to and from storage media.




## Parsing and Translation

As query processing includes certain activities for data retrieval. Initially, the given user queries get translated in high-level database languages such as SQL. It gets translated into expressions that can be further used at the physical level of the file system. After this, the actual evaluation of the queries and a variety of query -optimizing transformations and takes place.

## Query Evaluation Plan

- In order to fully evaluate a query, the system needs to construct a query evaluation plan.
- The annotations in the evaluation plan may refer to the algorithms to be used for the particular index or the specific operations.
- Such relational algebra with annotations is referred to as **Evaluation Primitives**. The evaluation primitives carry the instructions needed for the evaluation of the operation.

- Thus, a query evaluation plan defines a sequence of primitive operations used for evaluating a query. The query evaluation plan is also referred to as **the query execution plan**.
- A **query execution engine** is responsible for generating the output of the given query. It takes the query execution plan, executes it, and finally makes the output for the user query.

**Optimization**

- The cost of the query evaluation can vary for different types of queries. Although the system is responsible for constructing the evaluation plan, the user does need not to write their query efficiently.
- Usually, a database system generates an efficient query evaluation plan, which minimizes its cost. This type of task performed by the database system and is known as Query Optimization.
- For optimizing a query, the query optimizer should have an estimated cost analysis of each operation. It is because the overall operation cost depends on the memory allocations to several operations, execution costs, and so on.

**What is Relational Model?**

**Relational Model (RM)** represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.

The table name and column names are helpful to interpret the meaning of values in each row. The data are represented as a set of relations. In the relational model, data are stored as tables. However, the physical storage of the data is independent of the way the data are logically organized.

**Relational Model Concepts**

1. **Attribute:** Each column in a Table. Attributes are the properties which define a relation. e.g., Student_Rollno, NAME,etc.

2. **Tables** – In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.
3. **Tuple** – It is nothing but a single row of a table, which contains a single record.
4. **Relation Schema:** A relation schema represents the name of the relation with its attributes.
5. **Degree:** The total number of attributes which in the relation is called the degree of the relation.
6. **Cardinality:** Total number of rows present in the Table.
7. **Column:** The column represents the set of values for a specific attribute.
8. **Relation instance** – Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.
9. **Relation key** - Every row has one, two or multiple attributes, which is called relation key.
10. **Attribute domain** – Every attribute has some pre-defined value and scope which is known as attribute domain

**Keys in DBMS**

**KEYS in DBMS** is an attribute or set of attributes which helps you to identify a row(tuple) in a relation(table). They allow you to find the relation between two tables. Keys help you uniquely identify a row in a table by a combination of one or more columns in that table. Key is also helpful for finding unique record or row from the table. Database key is also helpful for finding unique record or row from the table.

**Why we need a Key?**

Here are some reasons for using sql key in the DBMS system.

- 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.

**Types of Keys in Database Management System**

There are mainly seven different types of Keys in DBMS and each key has its different functionality:

- **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.

Primary key example:

```
CREATE TABLE Persons (
   ID int NOT NULL,
   LastName varchar(255) NOT NULL,
   FirstName varchar(255),
   Age int,
   PRIMARY KEY (ID)
);
```

**Create Primary Key (ALTER TABLE statement)**

Syntax

The syntax to create a primary key using the ALTER TABLE statement in SQL is:

```
ALTER TABLE table_name
  ADD CONSTRAINT constraint_name
    PRIMARY KEY (column1, column2, ... column_n);
```

FOREIGN KEY on CREATE TABLE

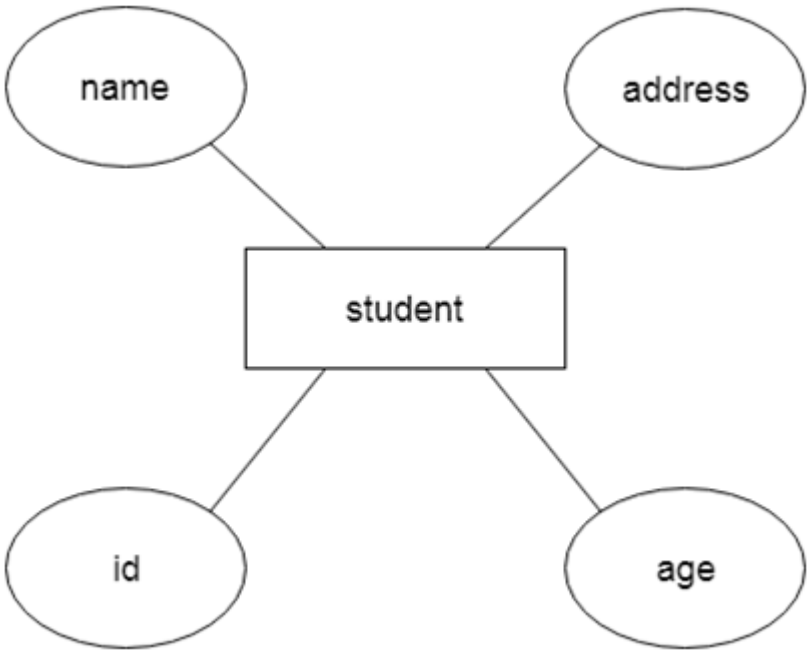
The following SQL creates a FOREIGN KEY on the "PersonID" column when the "Orders" table is created:

```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

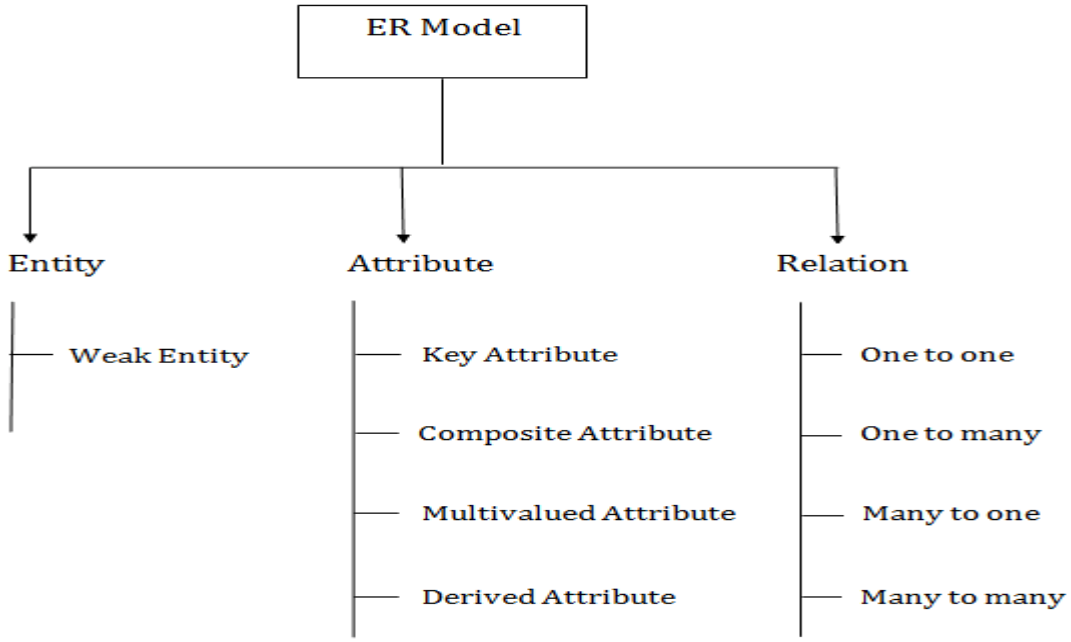
**ER model**

- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

**For example,** Suppose we design a school database. In this database, the student will be an entity with attributes like address, name, id, age, etc. The address can be another entity with attributes like city, street name, pin code, etc and there will be a relationship between them.



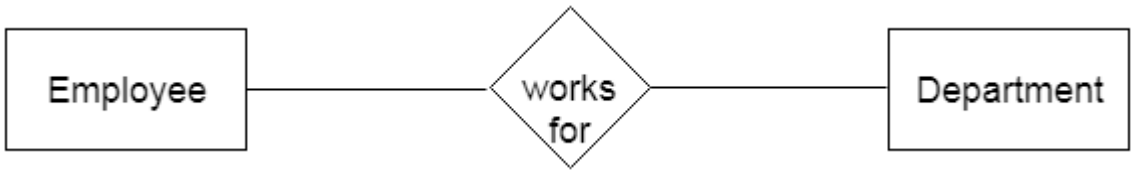

Component of ER Diagram




1. Entity:

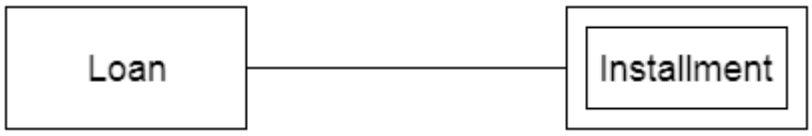
An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

Consider an organization as an example- manager, product, employee, department etc. can be taken as an entity.




**a. Weak Entity**

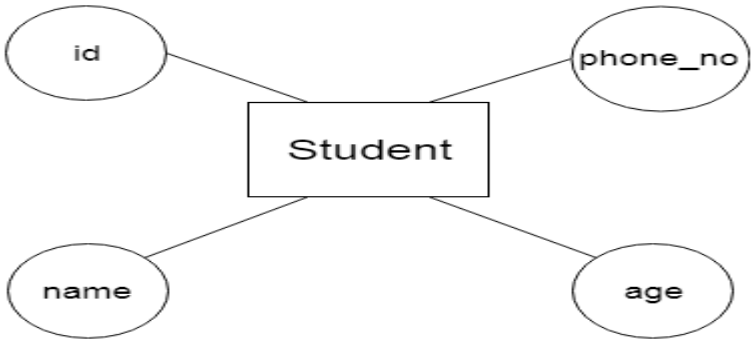
An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.




2. Attribute

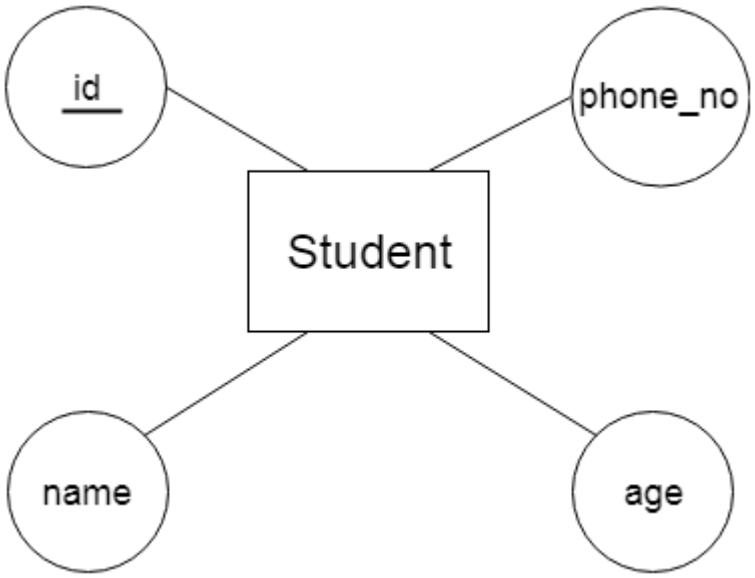
The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

**For example,** id, age, contact number, name, etc. can be attributes of a student.



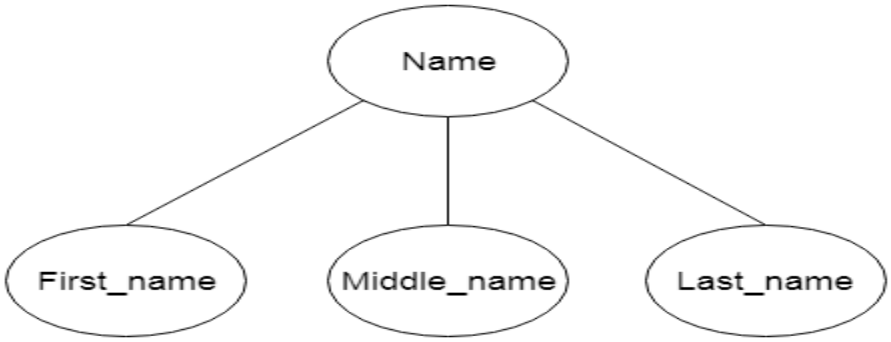

**a. Key Attribute**

The key attribute is used to represent the main characteristics of an entity. It represents a primary key. The key attribute is represented by an ellipse with the text underlined.




**b. Composite Attribute**

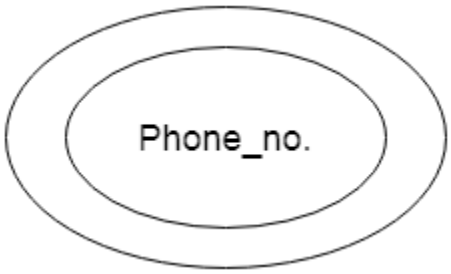
An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse, and those ellipses are connected with an ellipse.




**c. Multivalued Attribute**

An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute.

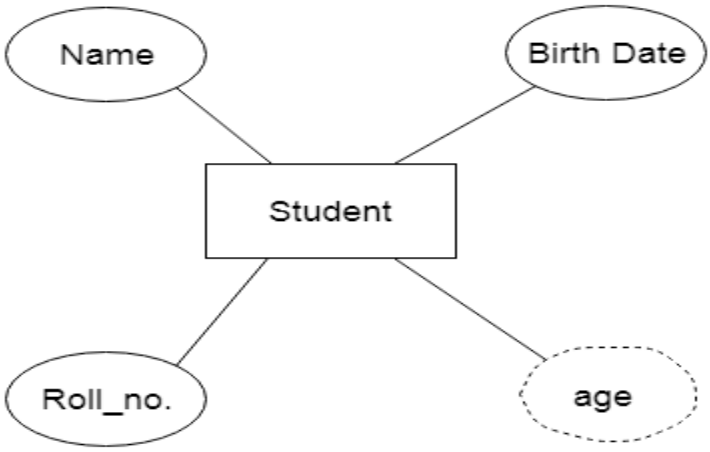
**For example,** a student can have more than one phone number.




**d. Derived Attribute**

An attribute that can be derived from other attribute is known as a derived attribute. It can be represented by a dashed ellipse.

**For example,** A person's age changes over time and can be derived from another attribute like Date of birth.




3. Relationship

A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship


Types of relationship are as follows:

**a. One-to-One Relationship**

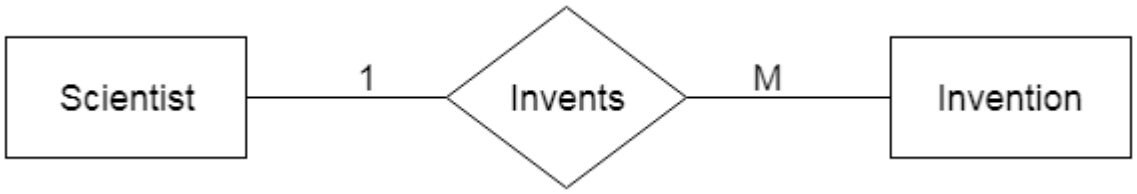
When only one instance of an entity is associated with the relationship, then it is known as one to one relationship.

**For example,** A female can marry to one male, and a male can marry to one female.


**b. One-to-many relationship**

When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then this is known as a one-to-many relationship.

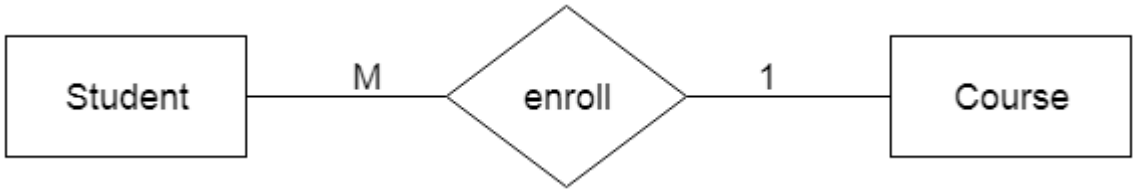
**For example,** Scientist can invent many inventions, but the invention is done by the only specific scientist.




**c. Many-to-one relationship**

When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as a many-to-one relationship.

**For example,** Student enrolls for only one course, but a course can have many students.



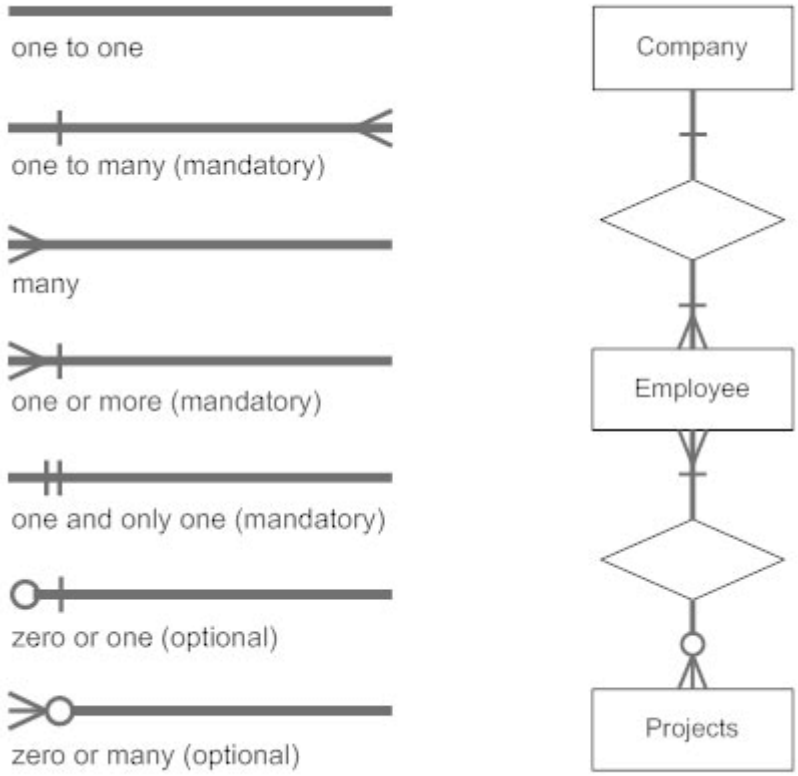
**d. Many-to-many relationship**

When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship.

**For example,** Employee can assign by many projects and project can have many employees.


Notation of ER diagram

Database can be represented using the notations. In ER diagram, many notations are used to express the cardinality. These notations are as follows:

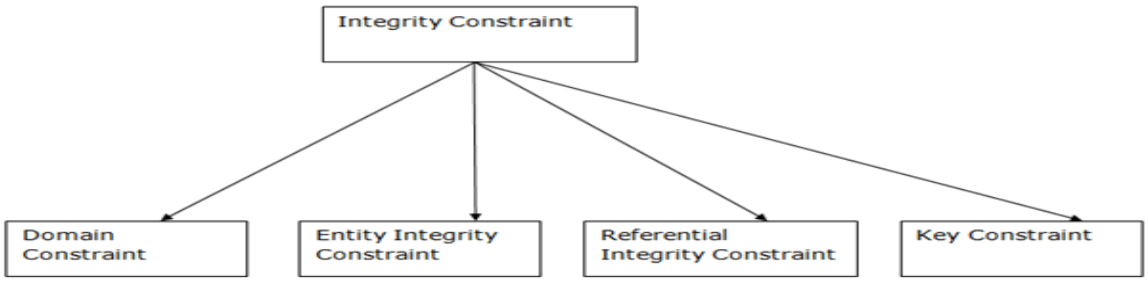



Integrity Constraints

- 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.

Types of Integrity Constraint




**1. 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

**2. 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.

**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

3. 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 |

**4. 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.

| 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

**ER Design Issues**

- ER design issues need to be discussed for better ER- design
- users often mislead the concept of the elements and the design process of the ER diagram. Thus, it leads to a complex structure of the ER diagram and certain issues that does not meet the characteristics of the real-world enterprise model.
- Here, we will discuss the basic design issues of an ER database schema in the following points:

1) Use of Entity Set vs Attributes

The use of an entity set or attribute depends on the structure of the real-world enterprise that is being modelled and the semantics associated with its attributes. It leads to a mistake when the user use the primary key of an entity set as an attribute of another entity set. Instead, he should use the relationship to do so. Also, the primary key attributes are implicit in the relationship set, but we designate it in the relationship sets.

2) Use of Entity Set vs. Relationship Sets

It is difficult to examine if an object can be best expressed by an entity set or relationship set.

3) Use of Binary vs n-ary Relationship Sets

Generally, the relationships described in the databases are binary relationships. However, non-binary relationships can be represented by several binary relationships.

4) Placing Relationship Attributes

The cardinality ratios can become an affective measure in the placement of the relationship attributes. So, it is better to associate the attributes of one-to-one or one-to-many relationship sets with any participating entity sets, instead of any relationship set.

**Conceptual design**

Conceptual design is the first stage in the database design process. The goal at this stage is to design a database that is independent of database software and physical details. The output of this process is a conceptual data model that describes the main data entities, attributes, relationships, and constraints of a given problem domain.

Keep in mind the following minimal data rule:

"All that is needed is there, and all that is there is needed".

1. Data analysis and requirements

2. Entity relationship modeling and normalization

3. Data model verification

4. Distributed database design