DBMS Mid-1 Important Questions

Unit-1

1) What is DBMS? Explain DBMS architecture with neat diagram.

Ans:-

INTRODUCTION

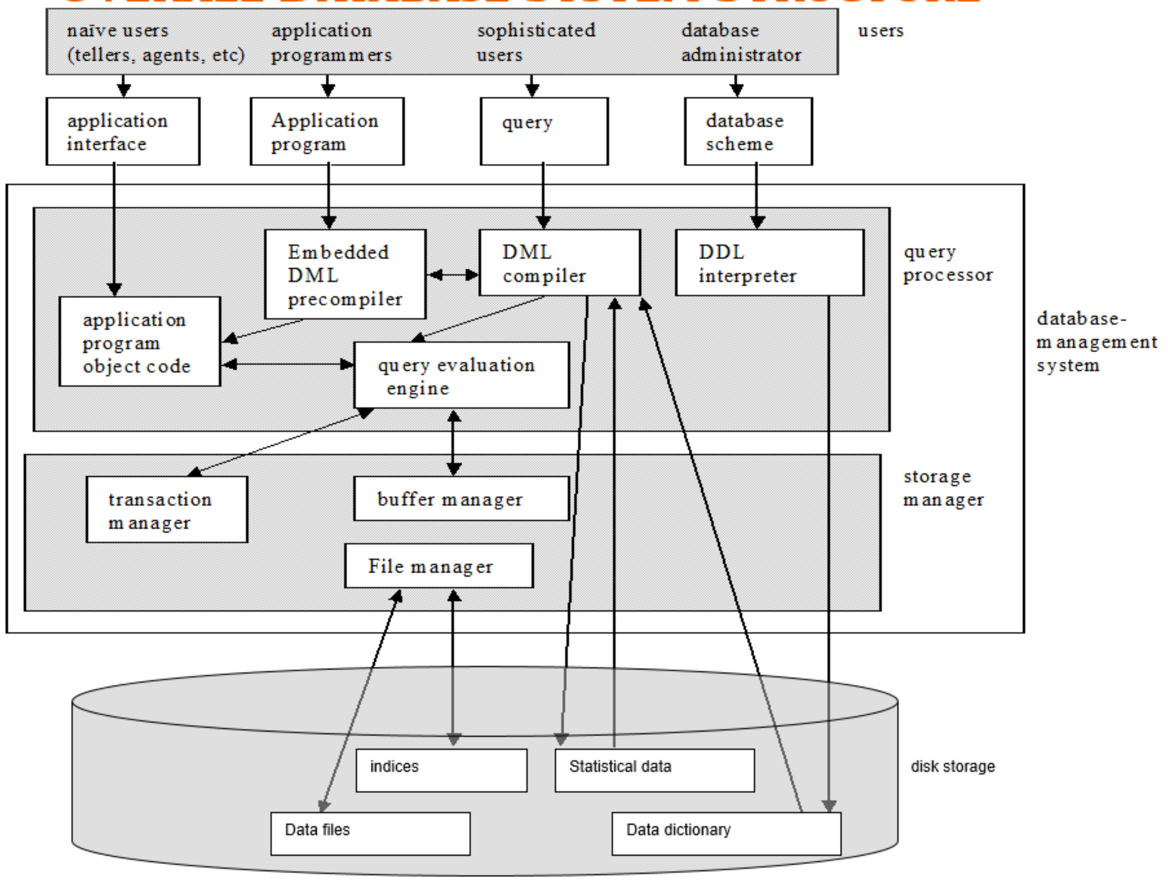
A database is a collection of related data

Eg: telephone numbers.

- A database can be of any size and varying complexity.

  E:g: library, income tax department
- A database may be generated and maintained manually or it may be computerized
- The DBMS is a general-purpose software systems that facilitates the process of defining, constructing, manipulating, retrieving and sharing database among various users and applications.
- DBMS include protecting the database and maintaining it over long period of time.
- Protection includes both system protection against hardware or software malfunctions, and security protection unauthorized access.

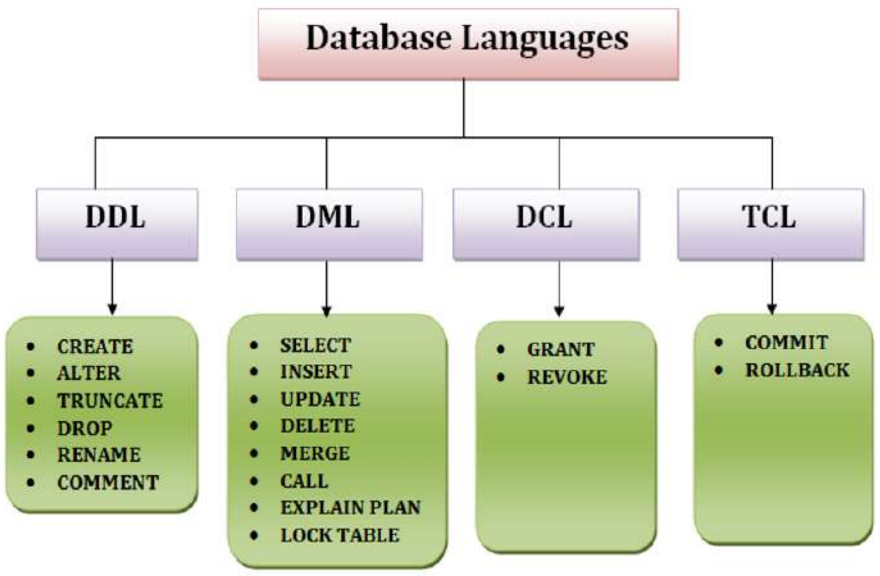
OVERALL DATABASE SYSTEM STRUCTURE



2) Explain Database language with syntax and example

Ans:-

TYPES OF DATABASE LANGUAGE



1. Data Definition Language

- **DDL** stands for **D**ata **D**efinition **L**anguage. It is used to define database structure or pattern.

- It is used to create schema, tables, indexes, constraints, etc. in the database.

- Using the DDL statements, you can create the skeleton of the database.

- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

- Here are some tasks that come under DDL:
  - **Create:** It is used to create objects in the database.
  - **Alter:** It is used to alter the structure of the database.
  - **Drop:** It is used to delete objects from the database.
  - **Truncate:** It is used to remove all records from a table.
  - **Rename:** It is used to rename an object.
  - **Comment:** It is used to comment on the data dictionary.

2. Data Manipulation Language

- **DML** stands for **D**ata **M**anipulation **L**anguage. It is used for accessing and manipulating data in a database. It handles user requests.
- Here are some tasks that come under DML:
  - **Select:** It is used to retrieve data from a database.
  - **Insert:** It is used to insert data into a table.
  - **Update:** It is used to update existing data within a table.
  - **Delete:** It is used to delete all records from a table.
  - **Merge:** It performs UPSERT operation, i.e., insert or update operations.
  - **Call:** It is used to call a structured query language or a Java subprogram.
  - **Explain Plan:** It has the parameter of explaining data.
  - **Lock Table:** It controls concurrency.

3. Data Control Language

- **DCL** stands for **D**ata **C**ontrol **L**anguage. It is used to retrieve the stored or saved data.
- The DCL execution is transactional. It also has rollback parameters.
- Here are some tasks that come under DCL:
  - **Grant:** It is used to give user access privileges to a database.
  - **Revoke:** It is used to take back permissions from the user.

4. Transaction Control Language

- TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.
- Here are some tasks that come under TCL:
  - **Commit:** It is used to save the transaction on the database.
  - **Rollback:** It is used to restore the database to original since the last Commit.

3) Explain about ER Model

Ans:-

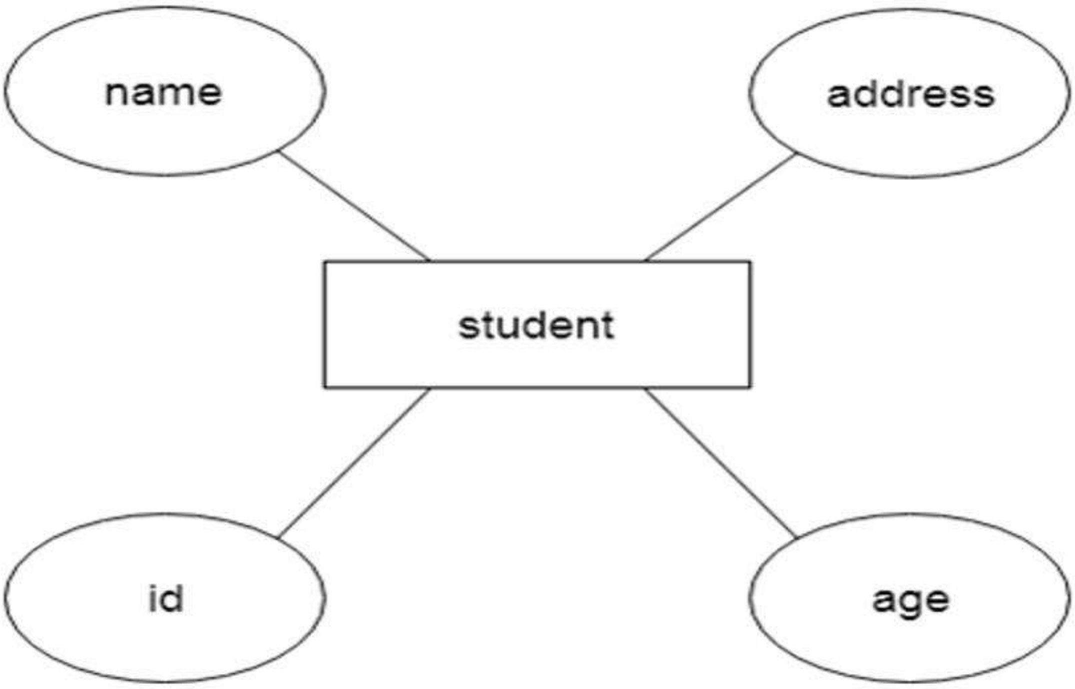
E-R Model

- 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.

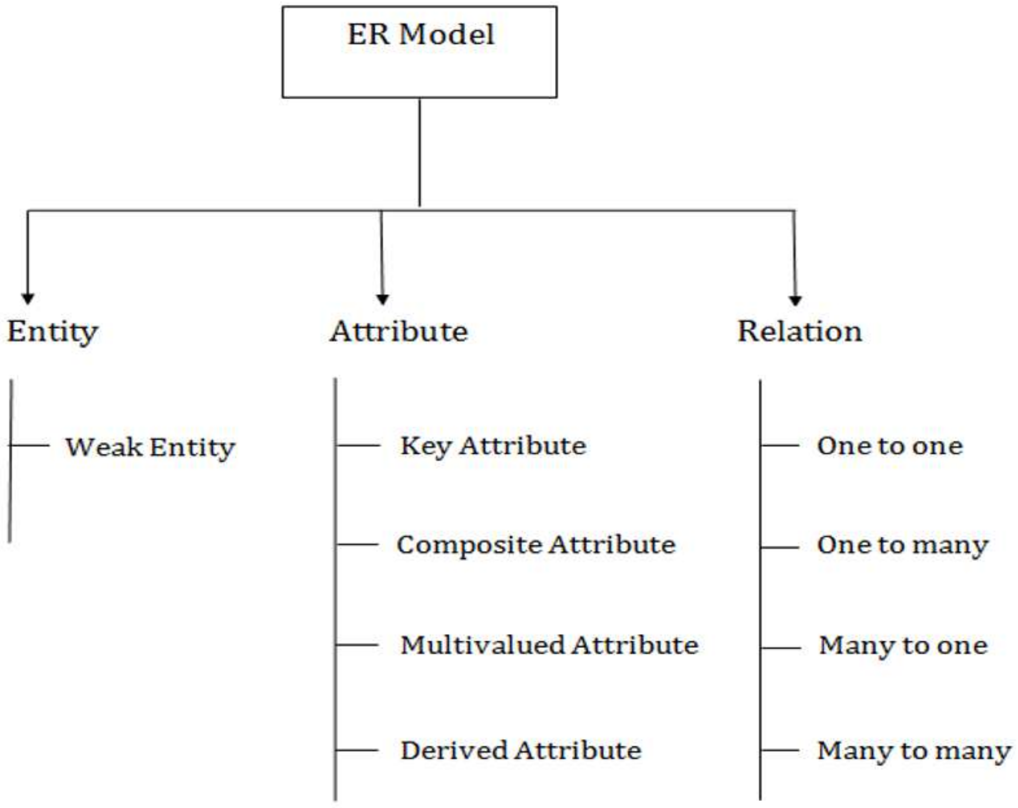
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.

- In ER modelling, the database structure is portrayed as a diagram called an entity-relationship diagram.
- Ex: School Student database

EXAMPLE:



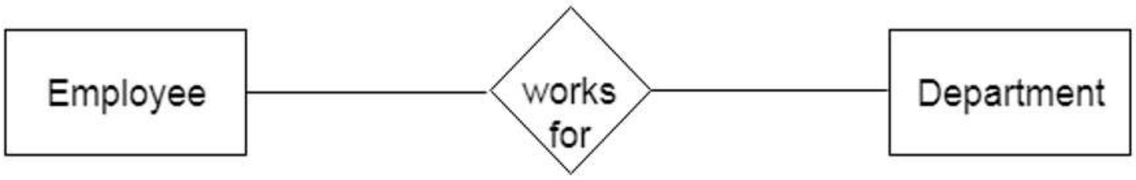
Component of ER Diagram



Entity

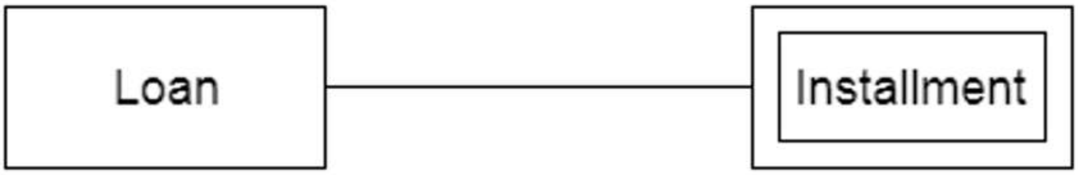
- An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

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



a. Weak Entity

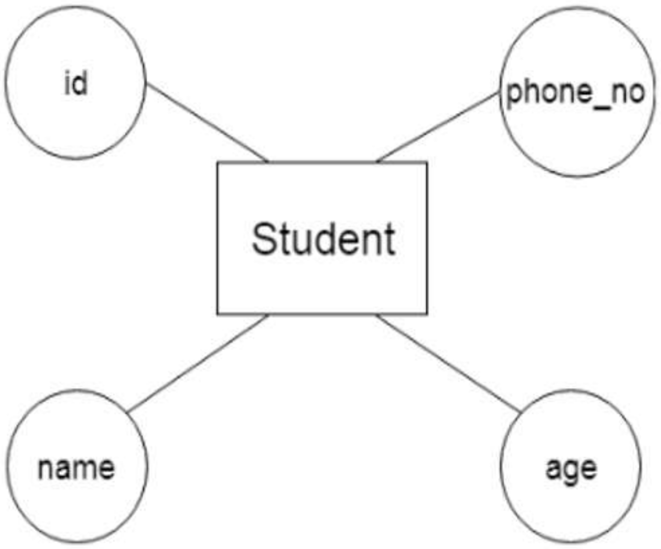
- An entity that depends on another entity called a weak entity.
- The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.



ATTRIBUTE

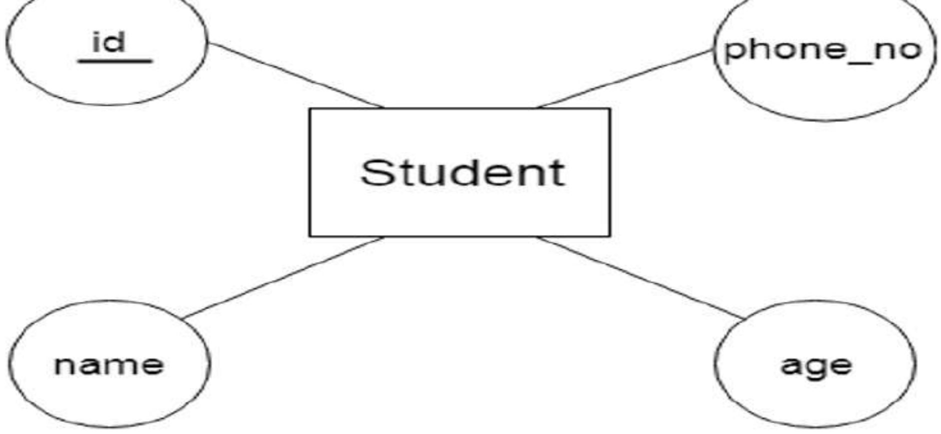
The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

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



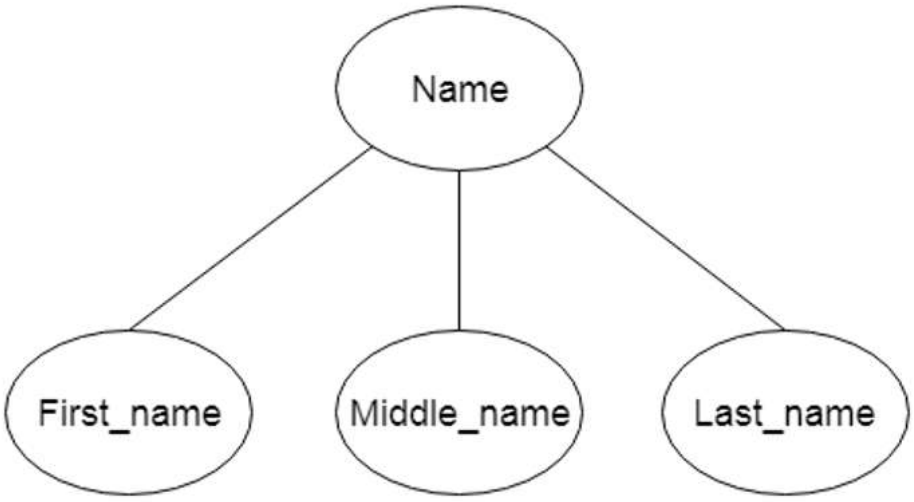
A. KEY ATTRIBUTE

- 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.



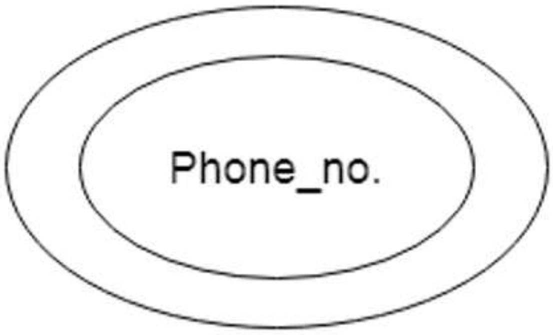
B. Composite Attribute

- 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.



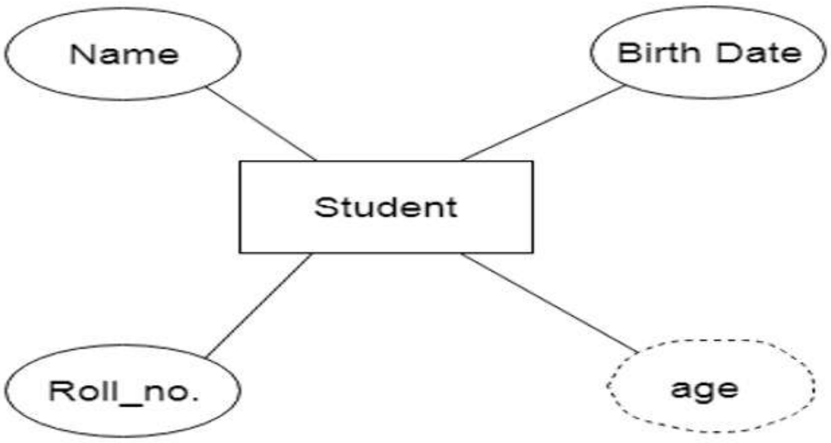
C. Multi valued Attribute

- An attribute can have more than one value. These attributes are known as a multi valued attribute. The double oval is used to represent multi valued attribute.
- **For example,** a student can have more than one phone number.



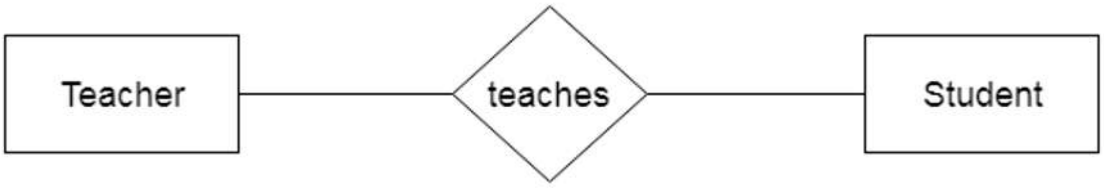
D. Derived Attribute

- 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

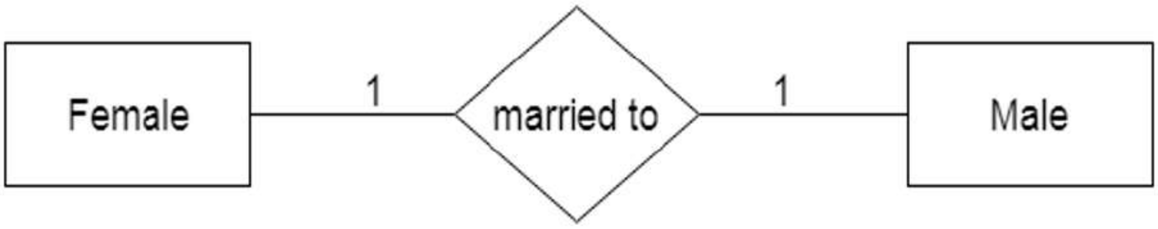
- A relationship is used to describe the relation between entities.
- Diamond or rhombus is used to represent the relationship.



1) 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.

Ex: A female can marry to one male, and a male can marry to one female.



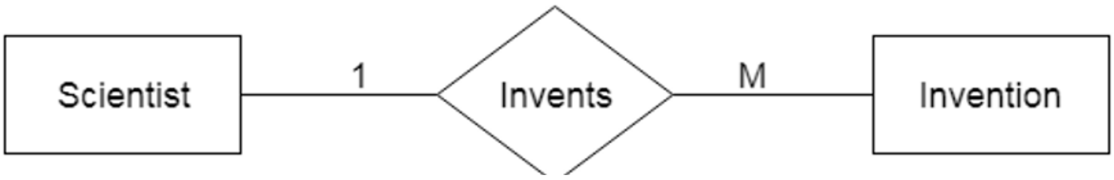
1) ONE-TO-ONE RELATIONSHIP:

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

Ex: A female can marry to one male, and a male can marry to one female.

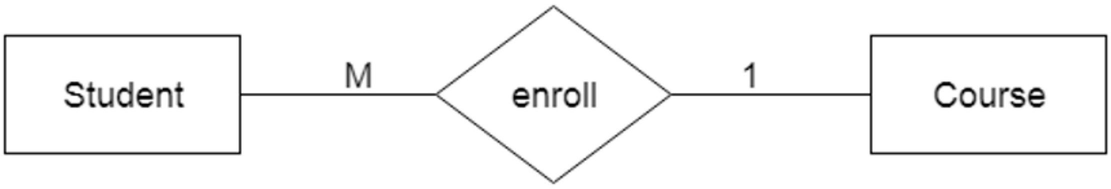
2) One-to-many relationship

- 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.



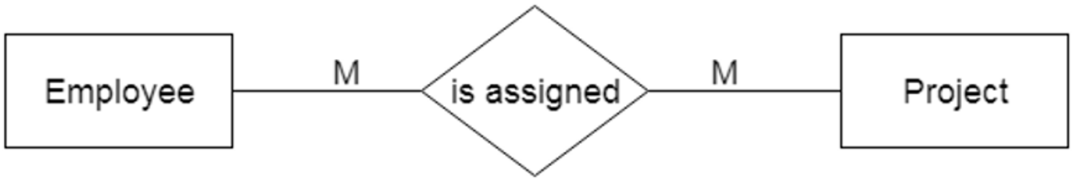
3) Many-to-one relationship

- 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.



4) 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.



4) What are the disadvantages of file system? How can you overcome these disadvantages with DBMS? Explain differences between file system and DBMS?

Ans:-

Database system Vs file systems

File system

- One way to keep the information on the computer and store it in operating system files.
- This system in order to allow the users to manipulate the information, the system must have number of application programs to manipulate the files.

File processing systems has number of major disadvantages:

Data redundancy

- Same information may be duplicated in several places
- Eg: address and telephone number of a particular customer may appear in a file that consists saving account records and in a file that consists of checking- account records

Data inconsistency

- Various copies of same data may no longer agree
- Eg: change of customer address may be reflected in savings account records but not in other records such as checking-account record .

Difficulty in accessing data

- File processing environment do not allow wanted data to de retrieved in convenient and efficient way.
- Eg: bank officer want the information of customers who lives in Hyderabad area, then the data processing department takes few days time to generate the information either manually or to write application program.
- If he want to trim the query again few days are required.

Data isolation

- As the data is scattered in different files , and the files may be in different formats, writing a new application programs to retrieve the appropriate data is difficult.

Integrity problem

- Data values stored in the data base must satisfy certain consistency constraints .
- E:g: (acc-balance>2500) this become a part of application program
- The constraints has changed or added it is difficult to change the program code.

Atomicity problems

- Failures of computer may leave database in an inconsistent state with partial updates carried out in file systems
- The fund transfer must be atomic
- Eg : transfer of funds from one account to another should either complete or not happen at all before the system fails.

Concurrent -access anomalies

- To provide supervision is difficult because data may be accessed by many different application program that have not been coordinated previously
- E:g : account 500 –A withdraw 50 and B withdraw 100 the balance should be 350 but it will shows as 450 for A and 400 for B which are not correct.

Security problems

- Every user of the database system should not access all the data
- As application programs are added to the file-processing system in an ad hoc manner, maintaining security is difficult.
- E:g: payroll personnel – bank employees, not customer accounts.

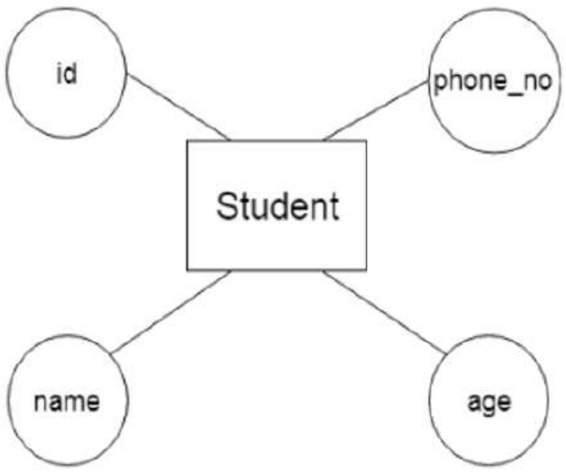
5) Explain different types of attributes

Ans:-

ATTRIBUTE

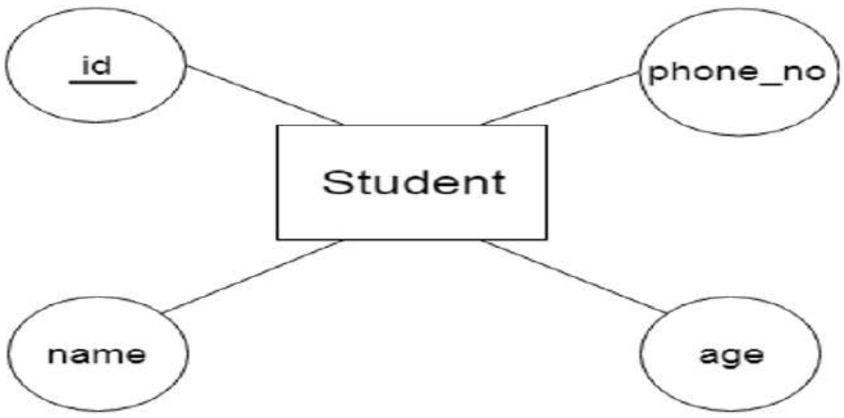
The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

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



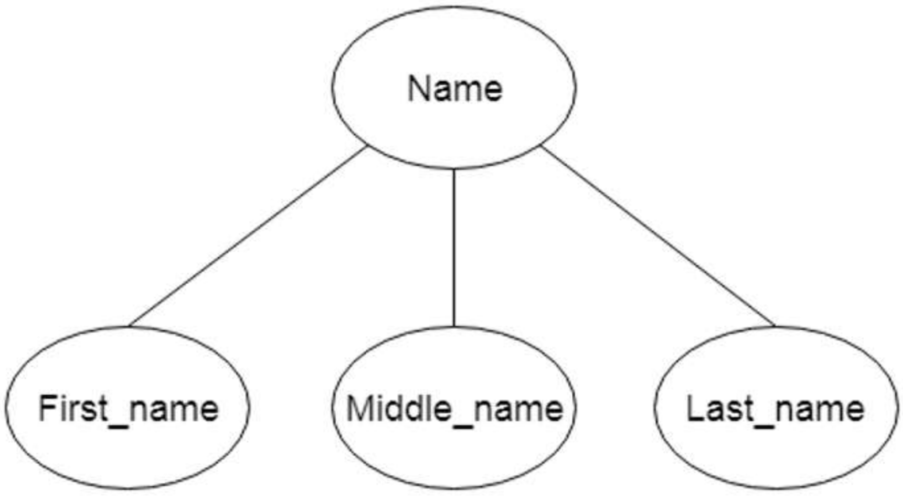
A. KEY ATTRIBUTE

- 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.



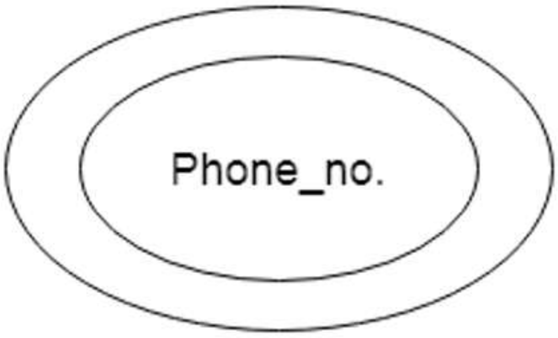
B. Composite Attribute

- 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.



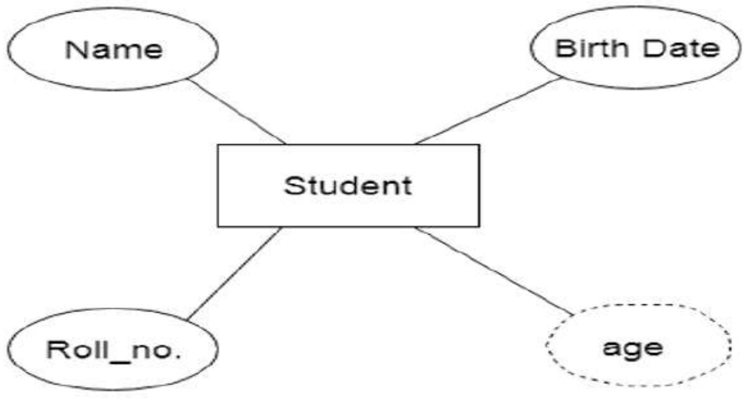
C. Multi valued Attribute

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



6) Application of DBMS

Ans:-

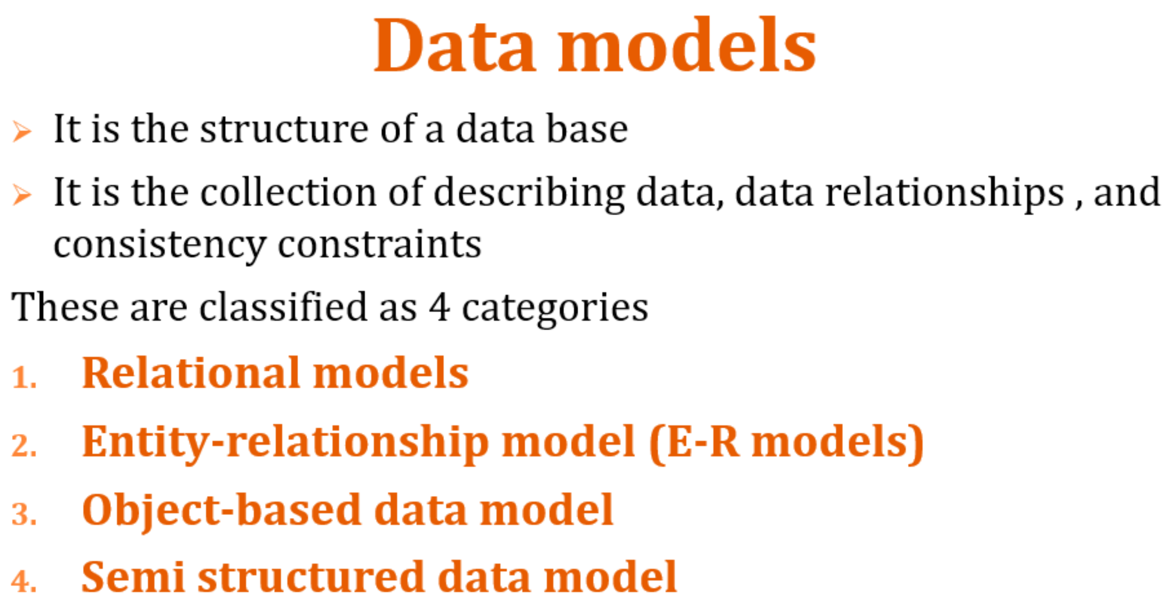
Data base system applications

Data base is widely used

- Banking
- Airlines
- Universities
- Credit card transactions
- Telecommunication
- Finance
- Sales
- Library
- Manufacturing
- Human resources

7) Define Data Model

Ans:-



Data models

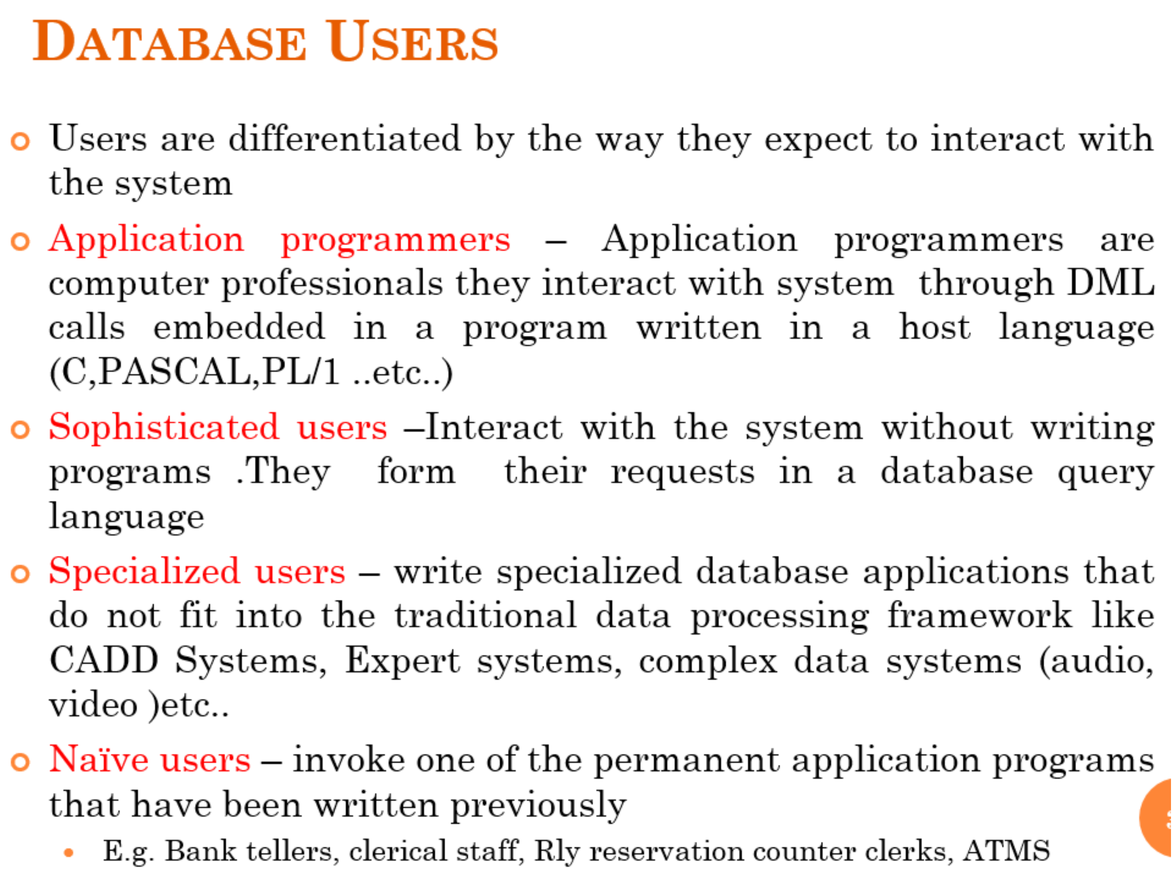
- It is the structure of a data base
- It is the collection of describing data, data relationships , and consistency constraints

These are classified as 4 categories

1. **Relational models**
2. **Entity-relationship model (E-R models)**
3. **Object-based data model**
4. **Semi structured data model**

8) Different types of Database users

Ans:-



DATABASE USERS

- Users are differentiated by the way they expect to interact with the system
- Application programmers – Application programmers are computer professionals they interact with system through DML calls embedded in a program written in a host language (C,PASCAL,PL/1 ..etc..)
- Sophisticated users –Interact with the system without writing programs .They form their requests in a database query language
- Specialized users – write specialized database applications that do not fit into the traditional data processing framework like CADD Systems, Expert systems, complex data systems (audio, video)etc..
- Naïve users – invoke one of the permanent application programs that have been written previously
  - E.g. Bank tellers, clerical staff, Rly reservation counter clerks, ATMS

Unit-2

1) DDL,DML Statements with Syntax and Examples

Ans:-

DDL allows you to create SQL statements to make operations with database data structures (schemas, tables etc.).

These are SQL DDL commands list and examples:

CREATE

CREATE statement is used to create a new database, table, index or stored procedure.

Create database example:

```
CREATE DATABASE explainjava;
```

Create table example:

```
CREATE TABLE user (
  id INT(16) PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(255) NOT NULL
);
```

DROP

DROP statement allows you to remove database, table, index or stored procedure.

Drop database example:

```
DROP DATABASE explainjava;
```

Drop table example:

```
DROP TABLE user;
```

ALTER

ALTER is used to modify existing database data structures (database, table).

Alter table example:

```
ALTER TABLE user ADD COLUMN lastname VARCHAR(255) NOT NULL;
```

RENAME

RENAME command is used to rename SQL table.

Rename table example:

```
RENAME TABLE user TO student;
```

TRUNCATE

TRUNCATE operation is used to delete all table records.

Logically it's the same as DELETE command.

Differences between DELETE and TRUNCATE commands are:

- TRUNCATE is really faster
- TRUNCATE cannot be rolled back
- TRUNCATE command does not invoke ON DELETE triggers

Example:

```
TRUNCATE student;
```

DML is a Data Manipulation Language, it's used to build SQL queries to manipulate (select, insert, update, delete etc.) data in the database.

This is DML commands list with examples:

SELECT

SELECT query is used to retrieve a data from SQL tables.

Example:

```
SELECT * FROM student;
```

INSERT

INSERT command is used to add new rows into the database table.

Example:

```
INSERT INTO student (name, lastname) VALUES ('Dmytro', 'Shvechikov');
```

UPDATE

UPDATE statement modifies records into the table.

Example:

```
UPDATE student SET name = 'Dima' WHERE lastname = 'Shvechikov';
```

DELETE

DELETE query removes entries from the table.

Example:

```
DELETE FROM student WHERE name = 'Dima';
```

2) Joins-Syntax, Types and Examples

Ans:-

Joins

- **Join in DBMS** is a binary operation which allows you to combine **join** product and selection in one single statement.
- The goal of creating a **join** condition is that it helps you to combine the data from two or more **DBMS** tables.
- The tables in **DBMS** are associated using the primary key and foreign keys.

Joins Types

- **INNER JOIN**
- **NATURAL JOIN**
- **LEFT OUTER JOIN**
- **RIGHT OUTER JOIN**
- **FULL OUTER JOIN**
- **SELF JOIN**

Inner Join

A **Inner Join** is a join operation that joins two tables by their common column with duplicate attributes. It returns rows when there is a match in both tables

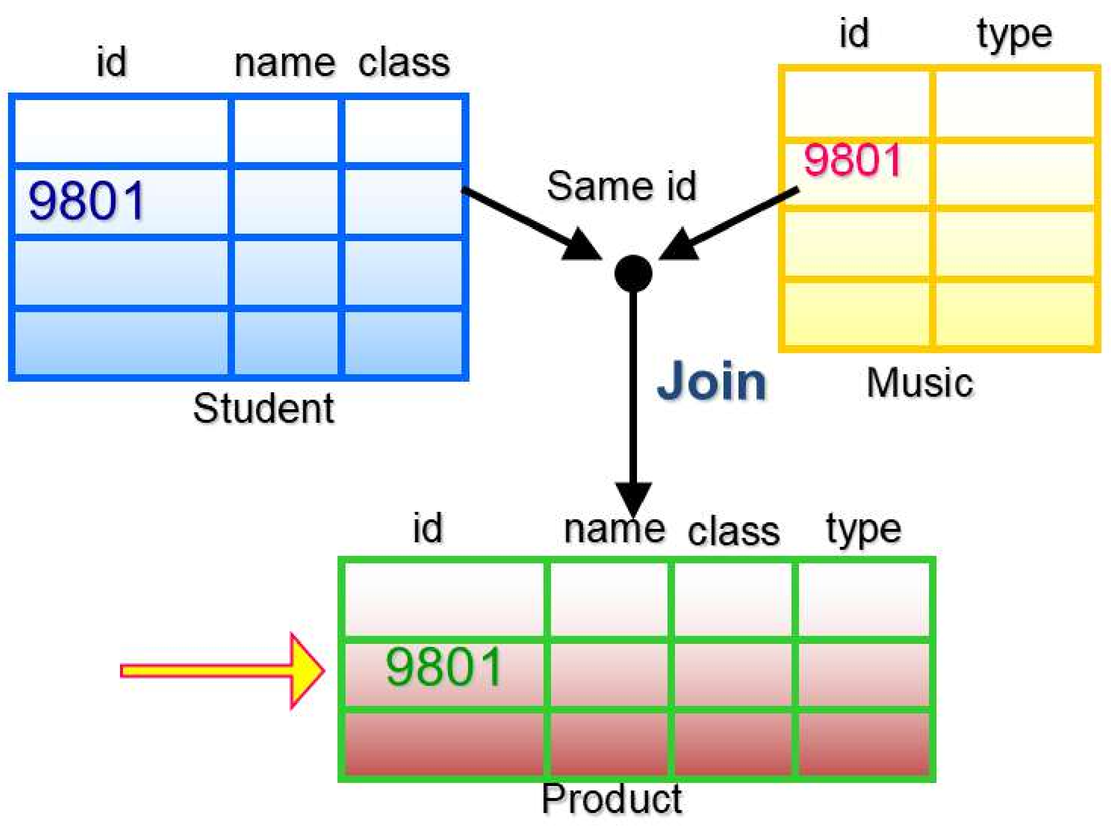
```
Select * from emp e, dept d where e.deptno=d.deptno;
(Or)
Select * from emp e join dept d on e.deptno=d.deptno;
```

Natural Join

A **Natural Join** is a join operation that joins two tables by their common column. Similar to inner join but it removes the duplicate columns in the result.

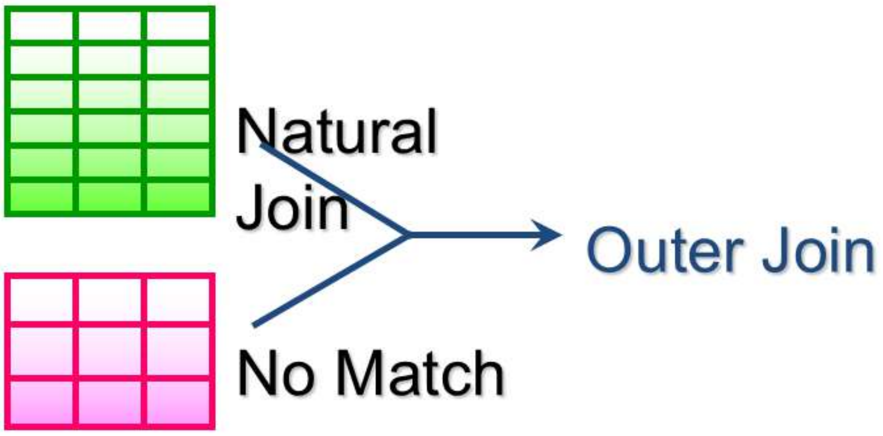
```
Select * from emp natural join dept;
```

Natural Join



Outer Join (Left, Right, Full)

An **Outer Join** is a join operation that includes rows that have a match, plus rows that do not have a match in the other table.



Left Outer Join

A **Left Outer Join** is returns all rows from the left table, even if there are no matches in the right table.

```
Select * from dept d left join emp e on
d.deptno=e.deptno;
                    (or)
Select * from dept d, emp e where
d.deptno=e.deptno(+);
```

Right Outer Join

A **Right Outer Join** is returns all rows from the right table, even if there are no matches in the left table.

```
Select * from emp e right join dept d on
e.deptno=d.deptno;
                    (or)
Select * from emp e, dept d where
e.deptno(+)=d.deptno;
```

Full Outer Join

A **Full Outer Join** is returns rows when there is a match in one of the tables

```
Select * from emp e full join dept d on
e.deptno=d.deptno;
```

Self Join

A **Self Join** is used to join a table to itself as if the table were two tables, temporarily renaming at least one table in the SQL statement.

We wanted to get a list of employees and their immediate managers:

```
Select e1.ename as Employee, e2.ename as Boss
from emp e1, emp e2 where e1.mgr=e2.empno;
```

3) Operators-Set

Ans:-

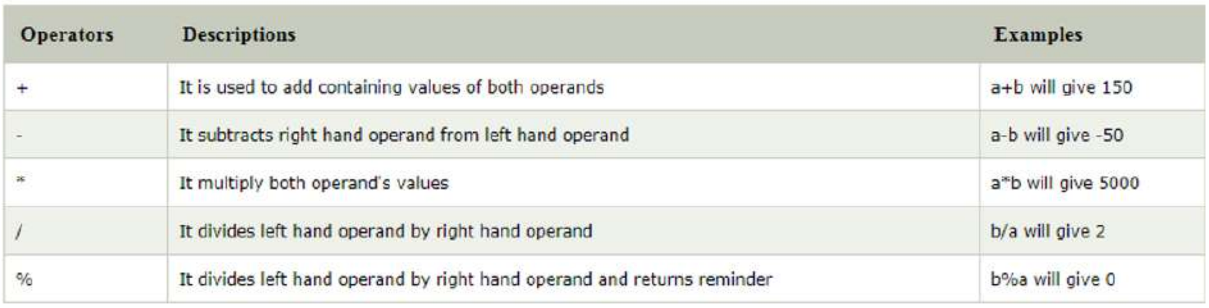
SQL Operators

- SQL statements generally contain some reserved words or characters that are used to perform operations such as comparison and arithmetical operations etc. These reserved words or characters are known as operators.

- Generally there are three types of operators in SQL:
  1. SQL Arithmetic Operators
  2. SQL Comparison Operators
  3. SQL Logical Operators

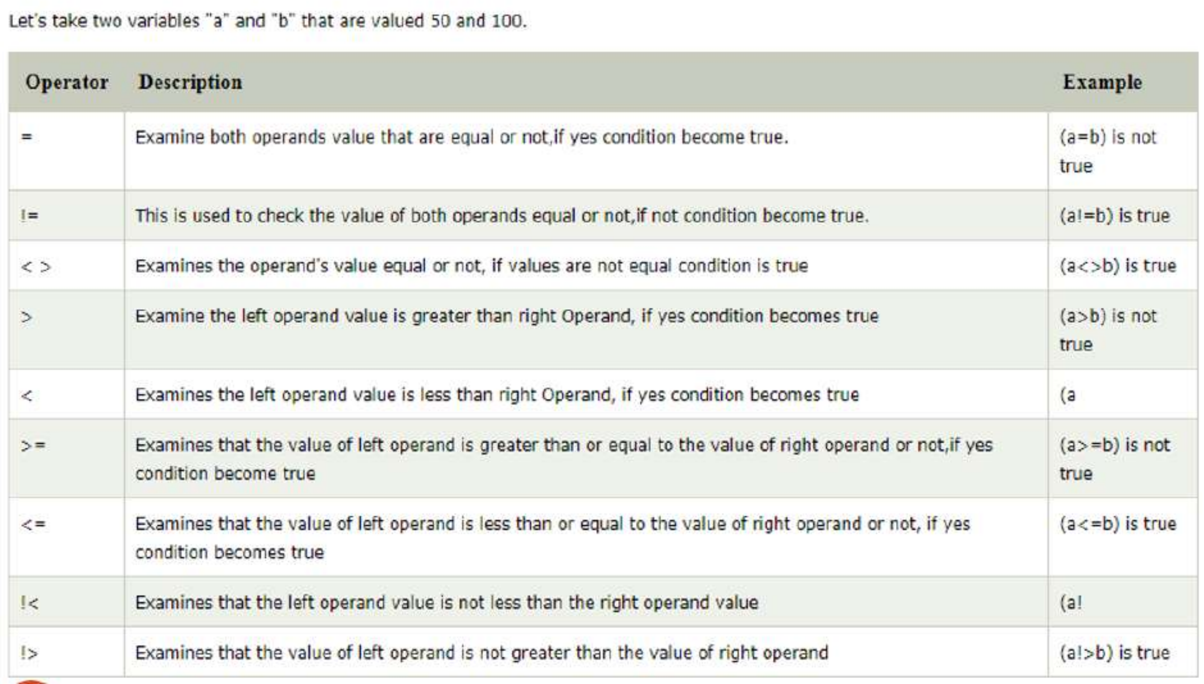
SQL Arithmetic Operators

- Let's assume two variables "a" and "b". Here "a" is valued 50 and "b" valued 100.



Operators	Descriptions	Examples
+	It is used to add containing values of both operands	a+b will give 150
-	It subtracts right hand operand from left hand operand	a-b will give -50
*	It multiply both operand's values	a*b will give 5000
/	It divides left hand operand by right hand operand	b/a will give 2
%	It divides left hand operand by right hand operand and returns reminder	b%a will give 0

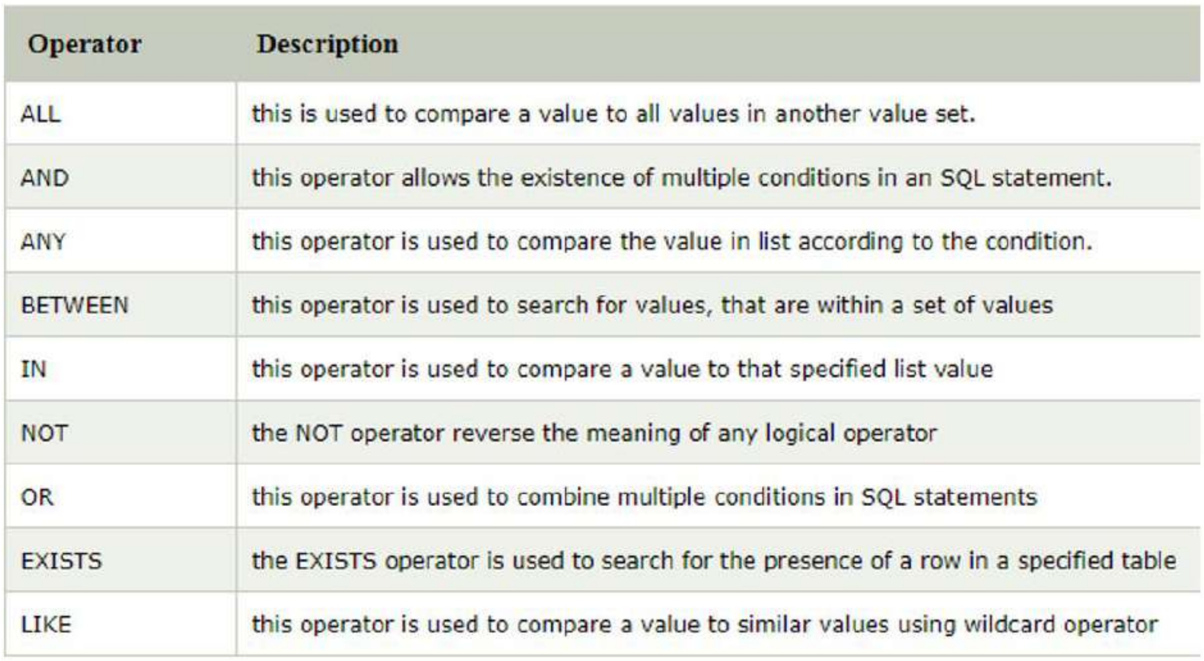
SQL Comparison Operators



Let's take two variables "a" and "b" that are valued 50 and 100.

Operator	Description	Example
=	Examine both operands value that are equal or not,if yes condition become true.	(a=b) is not true
!=	This is used to check the value of both operands equal or not,if not condition become true.	(a!=b) is true
< >	Examines the operand's value equal or not, if values are not equal condition is true	(a<>b) is true
>	Examine the left operand value is greater than right Operand, if yes condition becomes true	(a>b) is not true
<	Examines the left operand value is less than right Operand, if yes condition becomes true	(a
>=	Examines that the value of left operand is greater than or equal to the value of right operand or not,if yes condition become true	(a>=b) is not true
<=	Examines that the value of left operand is less than or equal to the value of right operand or not, if yes condition becomes true	(a<=b) is true
!<	Examines that the left operand value is not less than the right operand value	(a!
!>	Examines that the value of left operand is not greater than the value of right operand	(a!>b) is true

SQL Logical Operators



Operator	Description
ALL	this is used to compare a value to all values in another value set.
AND	this operator allows the existence of multiple conditions in an SQL statement.
ANY	this operator is used to compare the value in list according to the condition.
BETWEEN	this operator is used to search for values, that are within a set of values
IN	this operator is used to compare a value to that specified list value
NOT	the NOT operator reverse the meaning of any logical operator
OR	this operator is used to combine multiple conditions in SQL statements
EXISTS	the EXISTS operator is used to search for the presence of a row in a specified table
LIKE	this operator is used to compare a value to similar values using wildcard operator

SQL Set Operation

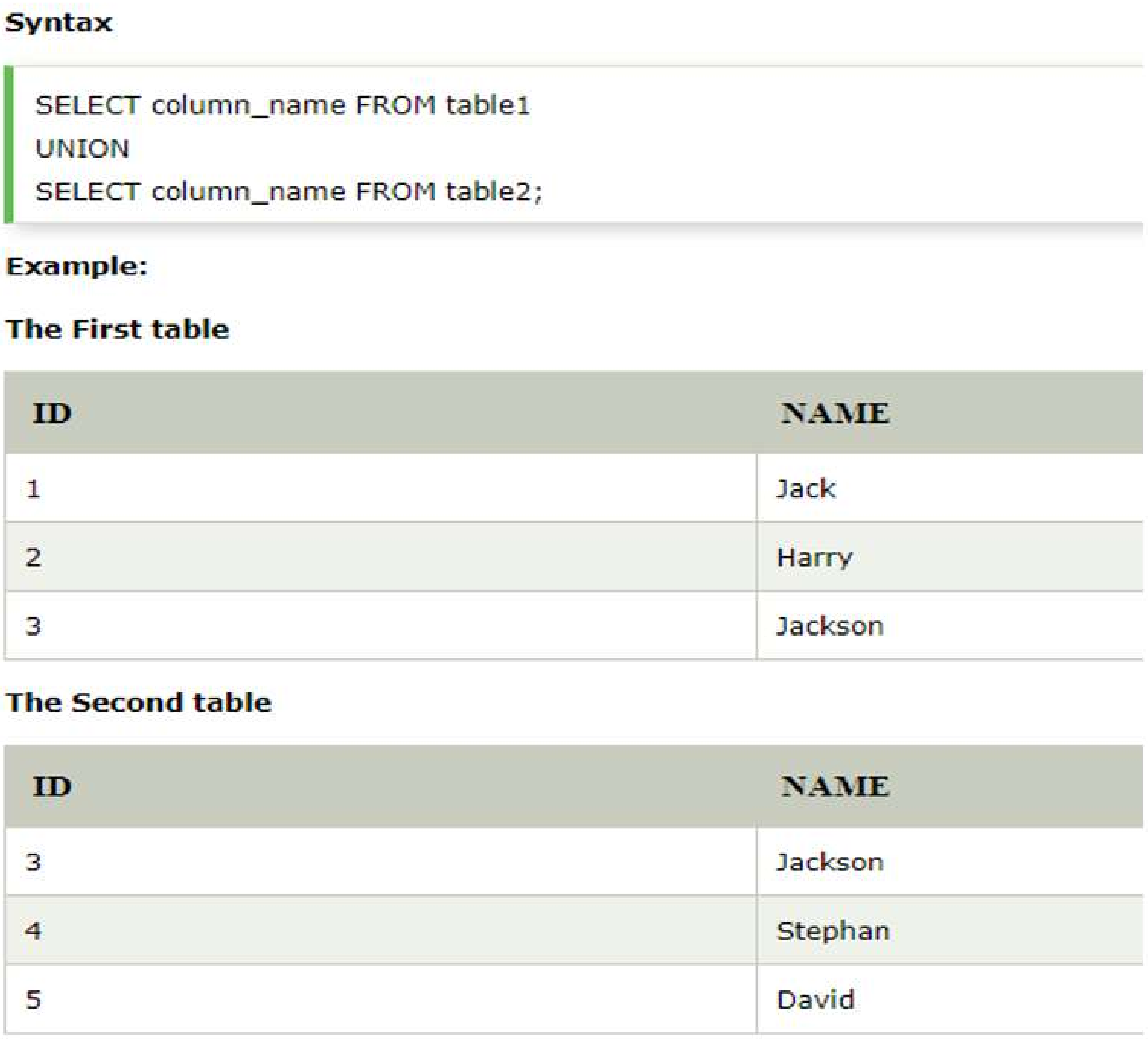
- The SQL Set operation is used to combine the two or more SQL SELECT statements.
- Types of Set Operation
    1. Union
    2. UnionAll
    3. Intersect
    4. Minus

Union
1
Union All
2
Set Operation
3
Intersect
4
Minus

1. Union

- The SQL Union operation is used to combine the result of two or more SQL SELECT queries.
- In the union operation, all the number of data type and columns must be same in both the tables on which UNION operation is being applied.
- The union operation eliminates the duplicate rows from its resultset.

Example:



Syntax

```
SELECT column_name FROM table1
UNION
SELECT column_name FROM table2;
```

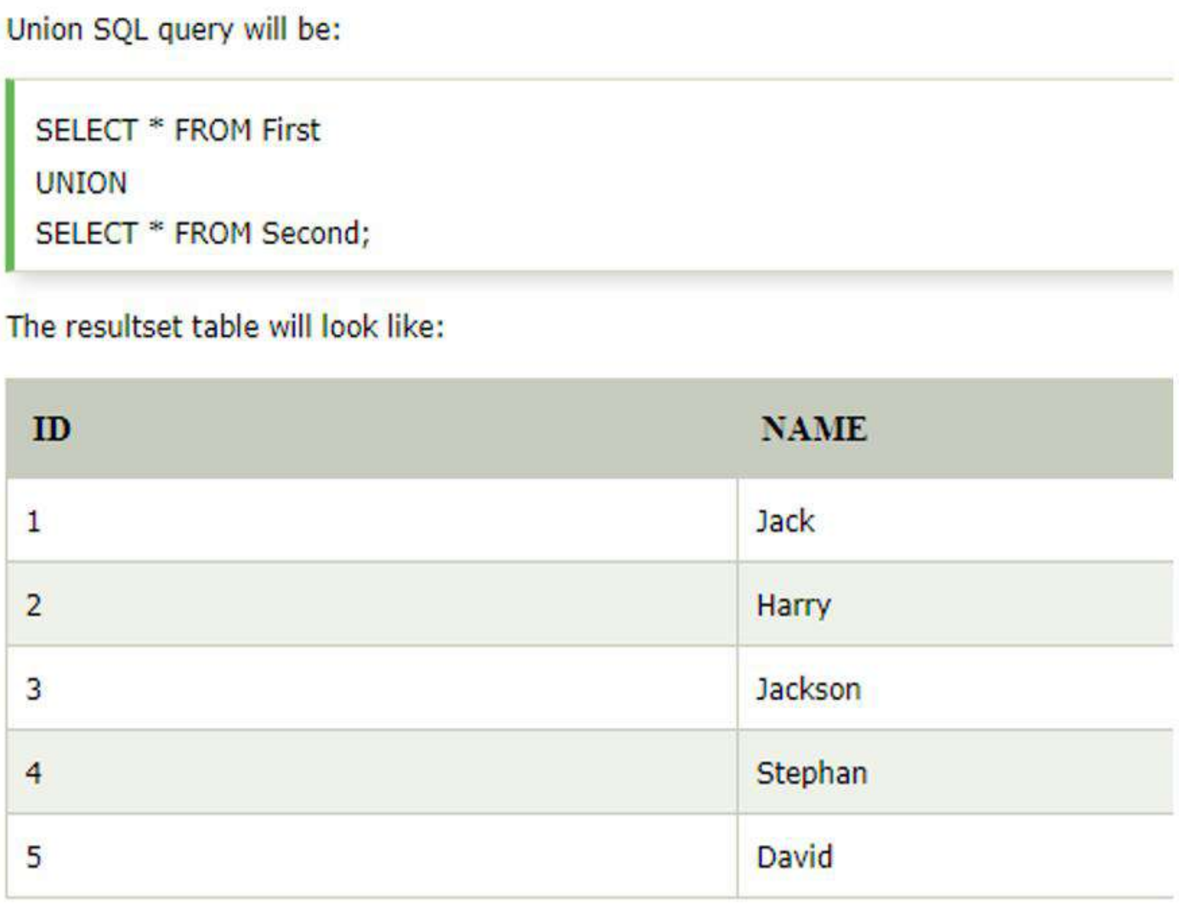
Example:

The First table

ID	NAME
1	Jack
2	Harry
3	Jackson

The Second table

ID	NAME
3	Jackson
4	Stephan
5	David



Union SQL query will be:

```
SELECT * FROM First
UNION
SELECT * FROM Second;
```

The resultset table will look like:

ID	NAME
1	Jack
2	Harry
3	Jackson
4	Stephan
5	David

2. Union All

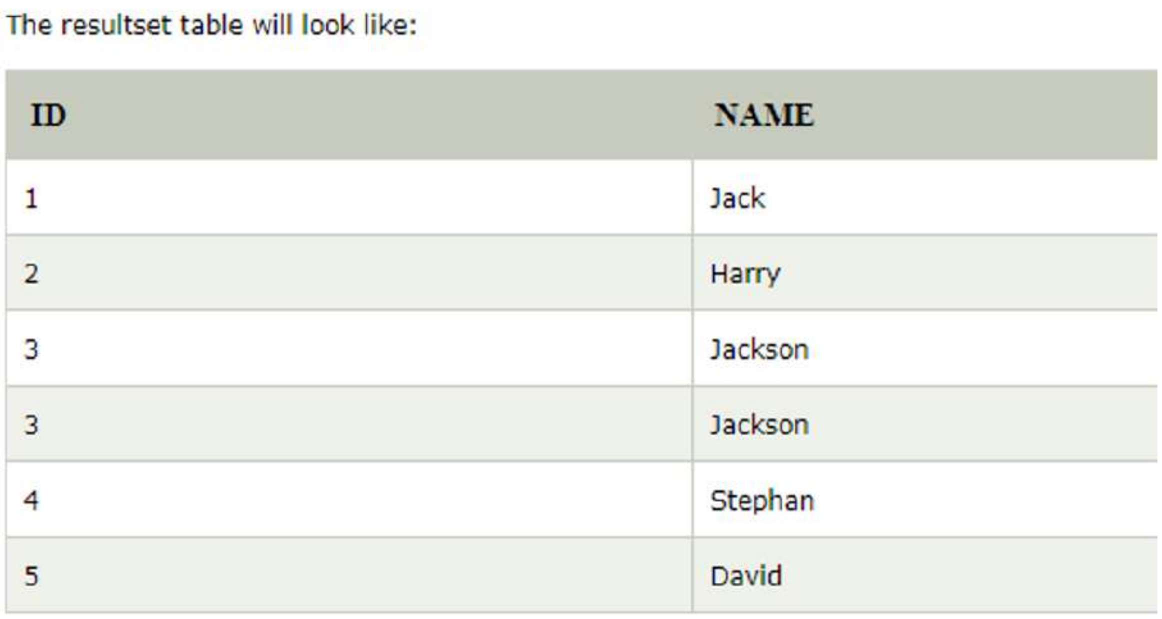
- Union All operation is equal to the Union operation. It returns the set without removing duplication and sorting the data.
- **Syntax:**

```
SELECT column_name FROM table1
UNION ALL
SELECT column_name FROM table2;
```

Example: Using the above First and Second table.

Union All query will be like:

```
SELECT * FROM First
UNION ALL
SELECT * FROM Second;
```



The resultset table will look like:

ID	NAME
1	Jack
2	Harry
3	Jackson
3	Jackson
4	Stephan
5	David

3. Intersect

- It is used to combine two SELECT statements. The Intersect operation returns the common rows from both the SELECT statements.
- In the Intersect operation, the number of datatype and columns must be the same.
- It has no duplicates and it arranges the data in ascending order by default.
- **Syntax**

```
SELECT column_name FROM table1
INTERSECT
SELECT column_name FROM table2;
```

- **Example:**
- **Using the above First and Second table.**
- Intersect query will be:

```
SELECT * FROM First
INTERSECT
SELECT * FROM Second;
```

The resultset table will look like:

ID	NAME
3	Jackson

4. Minus

- It combines the result of two SELECT statements. Minus operator is used to display the rows which are present in the first query but absent in the second query.
- It has no duplicates and data arranged in ascending order by default.
- **Syntax:**

```
SELECT column_name FROM table1
MINUS
SELECT column_name FROM table2;
```

Example

- **Using the above First and Second table.**
- Minus query will be:

```
SELECT * FROM First
MINUS
SELECT * FROM Second;
```

The resultset table will look like:

ID	NAME
1	Jack
2	Harry

4) Sub Queries

Ans:-

SQL Sub Query

- A Sub query is a query within another SQL query and embedded within the WHERE clause.
- **Important Rule:**
  - A sub query can be placed in a number of SQL clauses like WHERE clause, FROM clause, HAVING clause.
  - You can use Sub query with SELECT, UPDATE, INSERT, DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN, etc.
  - A sub query is a query within another query. The outer query is known as the main query, and the inner query is known as a sub query.
  - Sub queries are on the right side of the comparison operator.
  - A sub query is enclosed in parentheses.
  - In the Sub query, ORDER BY command cannot be used. But GROUP BY command can be used to perform the same function as ORDER BY command.

1. Sub queries with the Select Statement

- SQL subqueries are most frequently used with the Select statement.
- **Syntax**

```
SELECT column_name
FROM table_name
WHERE column_name expression operator
( SELECT column_name  from table_name WHERE ... );
```

2. Subqueries with the INSERT Statement

- SQL subquery can also be used with the Insert statement. In the insert statement, data returned from the subquery is used to insert into another table.
- In the subquery, the selected data can be modified with any of the character, date functions.
- **Syntax:**

```
INSERT INTO table_name (column1, column2, column3....)
SELECT *
FROM table_name
WHERE VALUE OPERATOR
```

3. Subqueries with the UPDATE Statement

- The subquery of SQL can be used in conjunction with the Update statement.
- When a subquery is used with the Update statement, then either single or multiple columns in a table can be updated.
- **Syntax**

```
UPDATE table
SET column_name = new_value
WHERE VALUE OPERATOR
  (SELECT COLUMN_NAME
  FROM TABLE_NAME
  WHERE condition);
```

4. Subqueries with the DELETE Statement

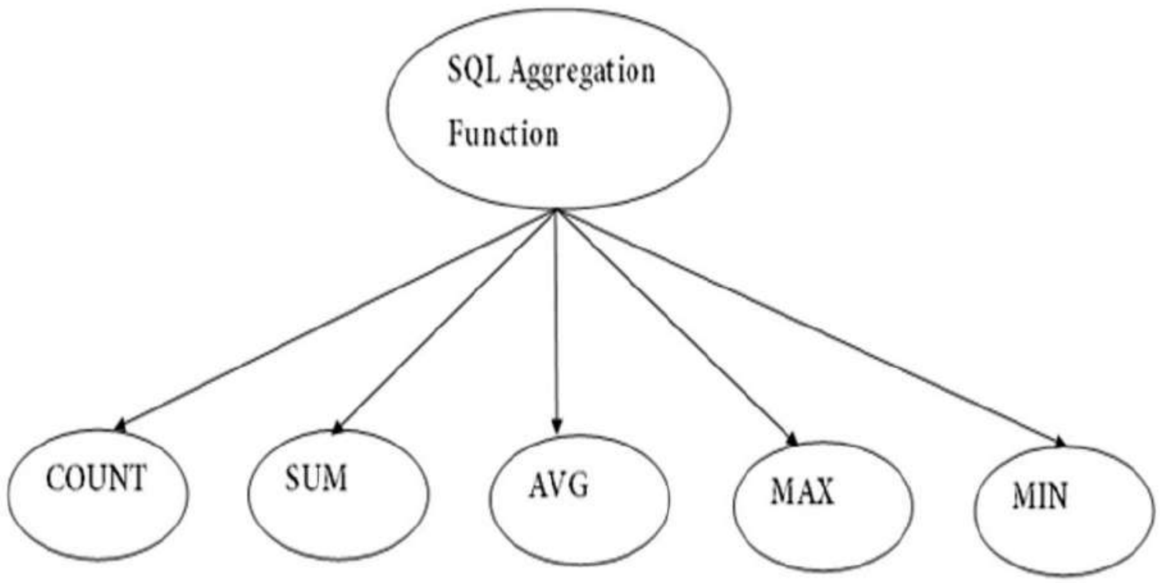
- The subquery of SQL can be used in conjunction with the Delete statement just like any other statements mentioned above.
- **Syntax**

```
DELETE FROM TABLE_NAME
WHERE VALUE OPERATOR
  (SELECT COLUMN_NAME
  FROM TABLE_NAME
  WHERE condition);
```

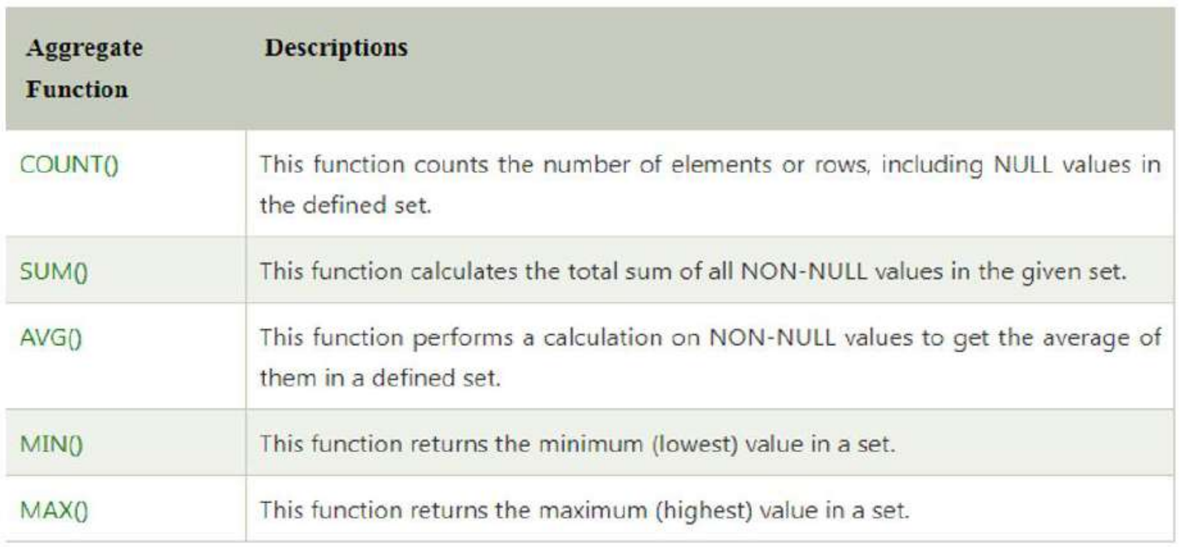
5) Aggregate Operators

Ans:-

SQL Aggregation Function



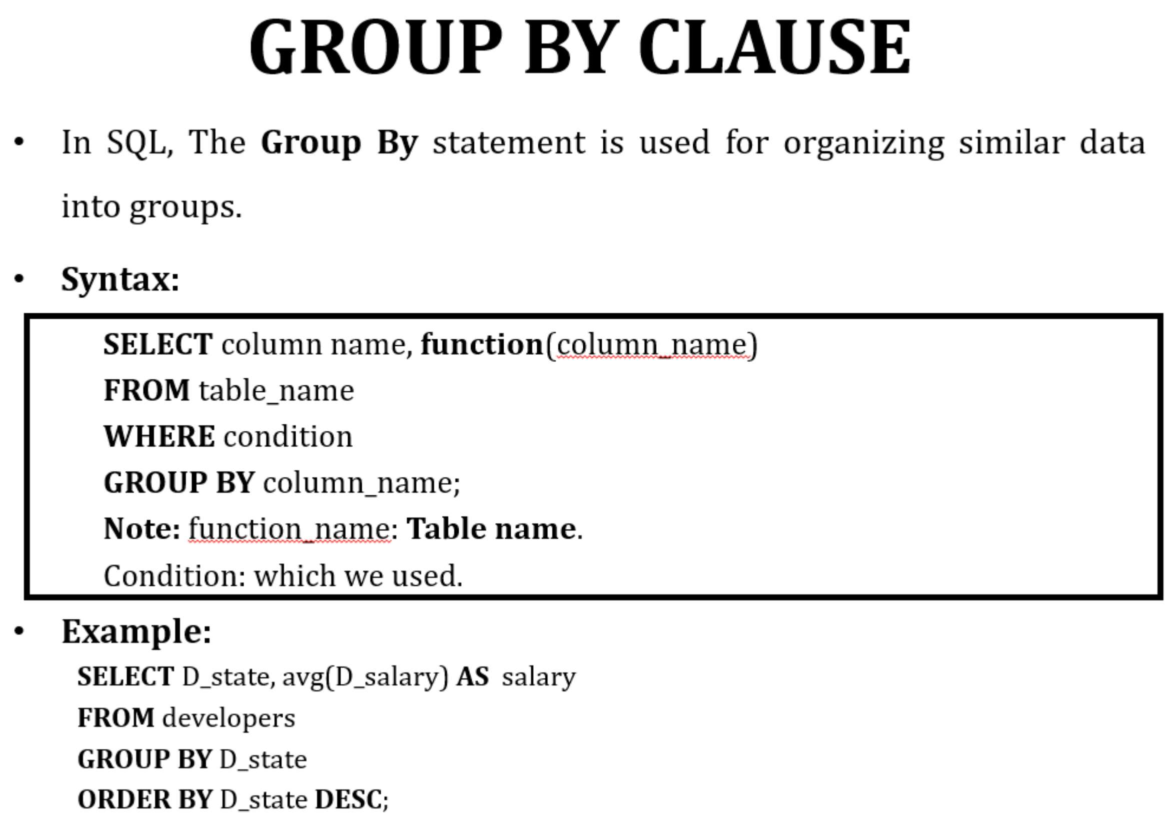
SQL Aggregation Function



Aggregate Function	Descriptions
COUNT()	This function counts the number of elements or rows, including NULL values in the defined set.
SUM()	This function calculates the total sum of all NON-NULL values in the given set.
AVG()	This function performs a calculation on NON-NULL values to get the average of them in a defined set.
MIN()	This function returns the minimum (lowest) value in a set.
MAX()	This function returns the maximum (highest) value in a set.

6) Group By, Having Clause

Ans:-



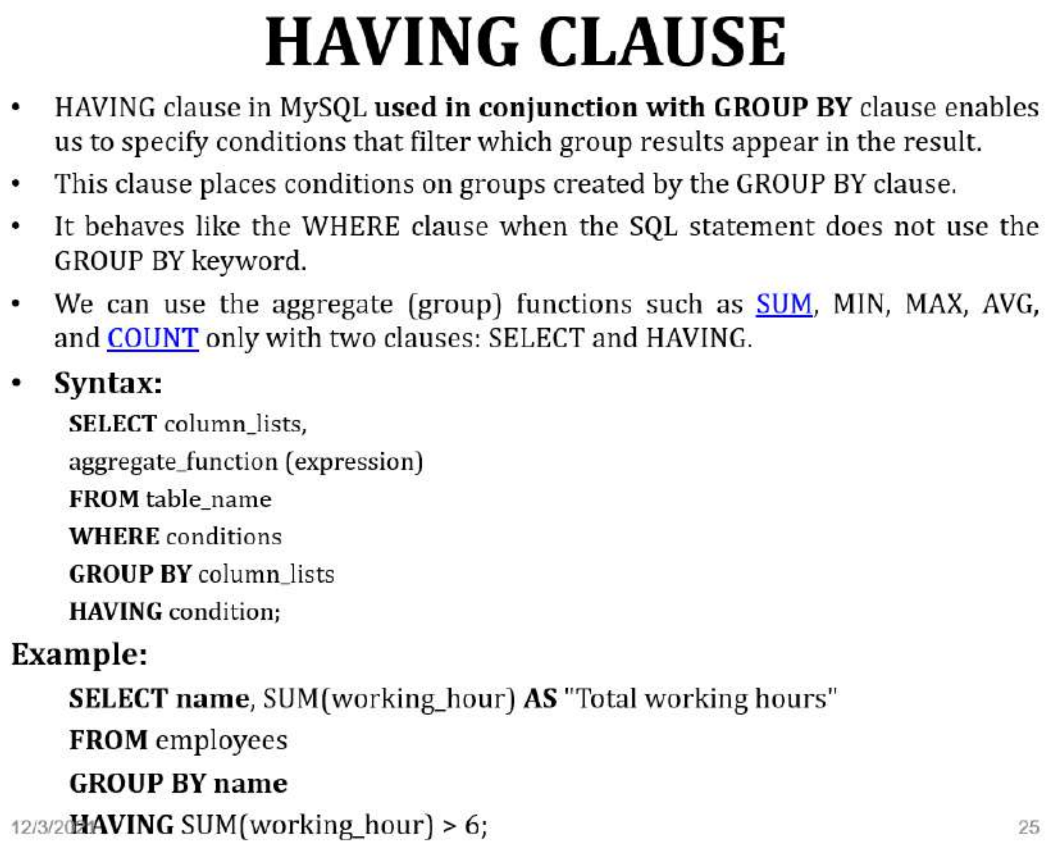
GROUP BY CLAUSE

- In SQL, The **Group By** statement is used for organizing similar data into groups.
- **Syntax:**

```
SELECT column name, function(column_name)
FROM table_name
WHERE condition
GROUP BY column_name;
Note: function_name: Table name.
Condition: which we used.
```

- **Example:**

```
SELECT D_state, avg(D_salary) AS  salary
FROM developers
GROUP BY D_state
ORDER BY D_state DESC;
```



HAVING CLAUSE

- HAVING clause in MySQL **used in conjunction with GROUP BY** clause enables us to specify conditions that filter which group results appear in the result.
- This clause places conditions on groups created by the GROUP BY clause.
- It behaves like the WHERE clause when the SQL statement does not use the GROUP BY keyword.
- We can use the aggregate (group) functions such as SUM, MIN, MAX, AVG, and COUNT only with two clauses: SELECT and HAVING.
- **Syntax:**

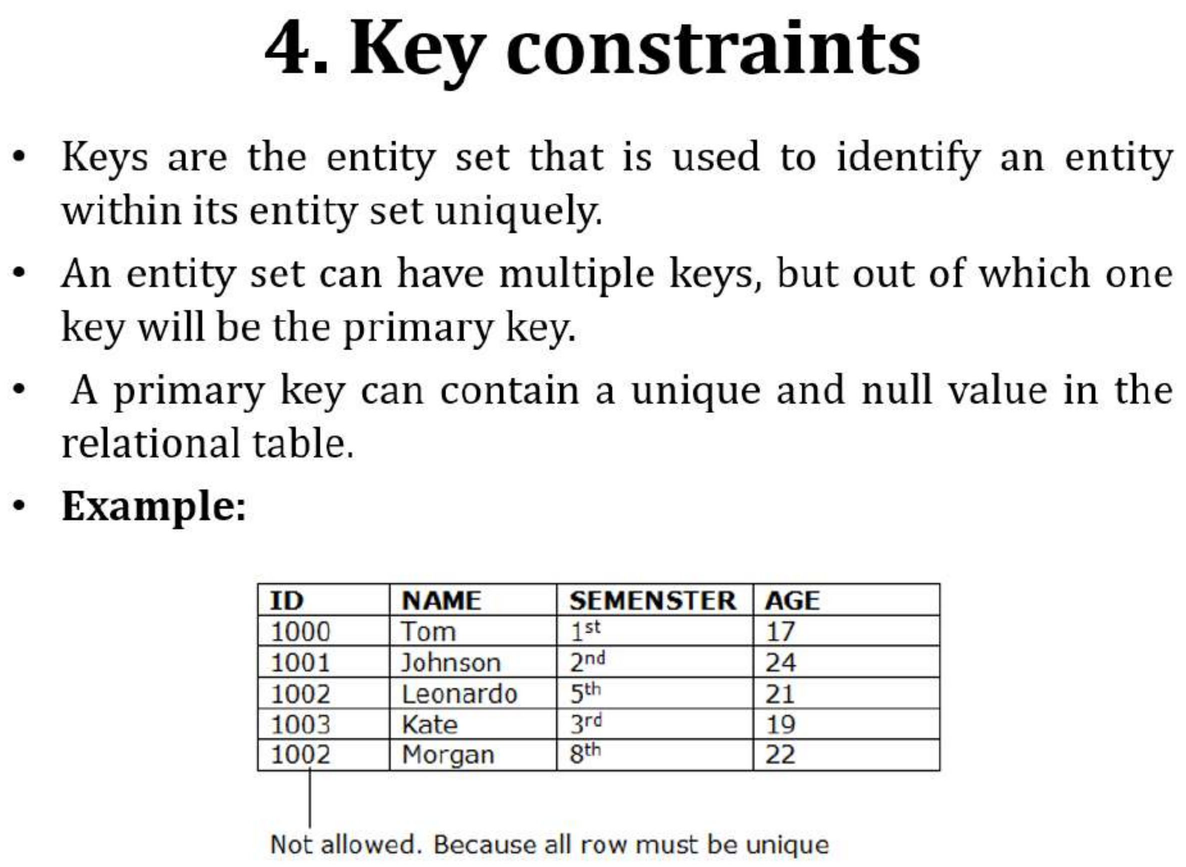
```
SELECT column_lists,
aggregate_function (expression)
FROM table_name
WHERE conditions
GROUP BY column_lists
HAVING condition;
```

Example:

```
SELECT name, SUM(working_hour) AS "Total working hours"
FROM employees
GROUP BY name
HAVING SUM(working_hour) > 6;
```

7) Key Constraints

Ans:-



4. Key constraints

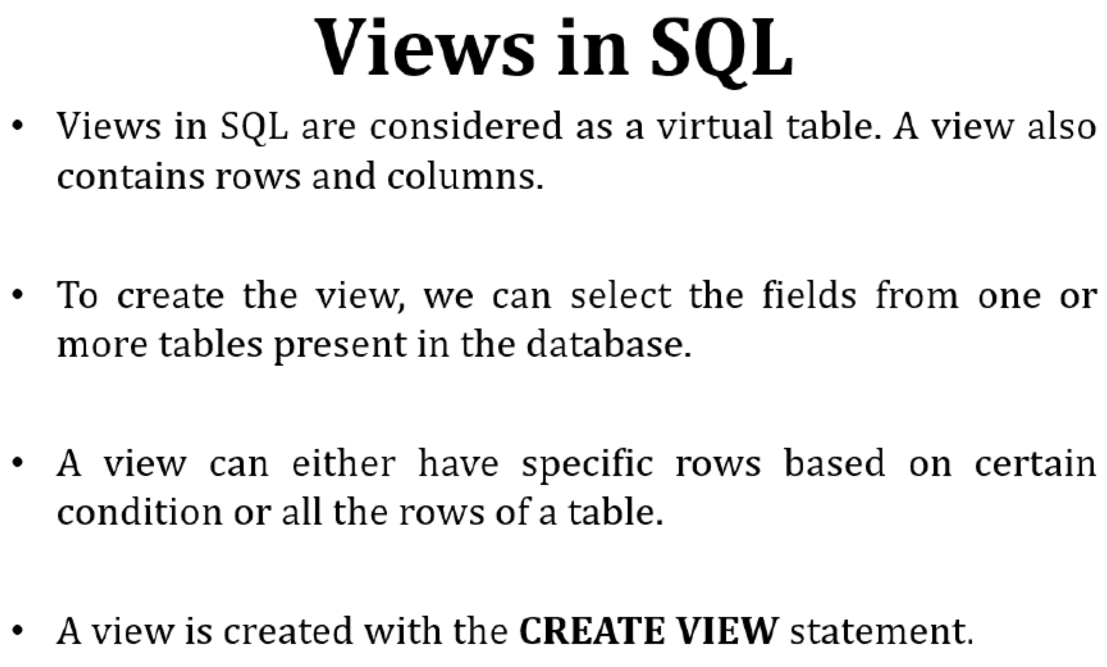
- Keys are the entity set that is used to identify an entity within its entity set uniquely.
- An entity set can have multiple keys, but out of which one key will be the primary key.
- A primary key can contain a unique and null value in the relational table.
- **Example:**

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

8) What is a View, Syntax and Examples

Ans:-



Views in SQL

- Views in SQL are considered as a virtual table. A view also contains rows and columns.
- To create the view, we can select the fields from one or more tables present in the database.
- A view can either have specific rows based on certain condition or all the rows of a table.
- A view is created with the **CREATE VIEW** statement.

1. Creating view

- A view can be created using the **CREATE VIEW** statement.
- We can create a view from a single table or multiple tables.
- **Syntax:**

```
CREATE VIEW view name AS
SELECT column1, column2.....
FROM table name
WHERE condition;
```

Note:

view name: Name for the View

table name: Name of the table

condition: Condition to select rows

a. Creating View from a single table

- In this example, we create a View named Details View from the table Student_Detail.
- **Query:**

```
CREATE VIEW Details View AS
SELECT NAME, ADDRESS
FROM Student Details
WHERE STU_ID < 4;
```

To see the data in the View:

```
SELECT * FROM Details View;
```

b. Creating View from multiple tables

- View from multiple tables can be created by simply include multiple tables in the SELECT statement.
- In the given example, a view is created named Marks View from two tables Student_Detail and Student Marks.
- **Query:**

```
CREATE VIEW Marks View AS
SELECT Student_Detail.NAME, Student_Detail. ADDRESS,
Student_Marks.MARKS
FROM Student_Detail, Student Mark
WHERE Student_Detail.NAME = Student_Marks.NAME;
```

To see the data in the View:

```
SELECT * FROM Marks View;
```

2.Updating View

- There are certain conditions needed to be satisfied to update a view.
- If any one of these conditions is **not** met, then we will not be allowed to update the view.
- We can use the **CREATE OR REPLACE VIEW** statement to add or remove fields from a view.
- **Syntax:**

```
CREATE OR REPLACE VIEW view_name AS
SELECT column1,coulmn2,..
FROM table_name
WHERE condition;
```

Example:

```
CREATE OR REPLACE VIEW MarksView AS
SELECT StudentDetails.NAME, StudentDetails.ADDRESS,
    StudentMarks.MARKS,  StudentMarks.AGE
FROM StudentDetails, StudentMarks
WHERE StudentDetails.NAME = StudentMarks.NAME;
```

- **If we fetch all the data from MarksView now as:**

  SELECT * FROM MarksView;

a. **Inserting a row in a view**:
   We can insert a row in a View in a same way as we do in a table. We can use the INSERT INTO statement of SQL to insert a row in a View.

   Syntax:

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

   Note: view_name: Name of the View

b. **Deleting a row from a View**:
   Deleting rows from a view is also as simple as deleting rows from a table. We can use the DELETE statement of SQL to delete rows from a view.

- **Syntax**: DELETE FROM view_name WHERE condition;
- **Note: view_name**: Name of view from where we want to delete rows
  condition: Condition to select rows
- **Example**:
  DELETE FROM DetailsView

  WHERE NAME="Suresh";

If we fetch all the data from DetailsView now as,

- SELECT * FROM DetailsView;

3. Deleting View

- SQL allows us to delete an existing View. We can delete or drop a View using the DROP statement.
- **Syntax**

  DROP VIEW view name;
- **Note:**
- **view name**: Name of the View which we want to delete.
- **Example:**
- DROP VIEW Marks View;

Unit-3

1) Relational Database Structure Terminology

Ans:-

Terminology Used in Basic Structure of Relational Model:

- **Tables** – In relational data model, relations are saved in the format of Tables. This format stores the relation among entities.
- A table has rows and columns, where rows represents records and columns represent the attributes.

- **Tuple** – A single row of a table, which contains a single record for that relation is called a tuple.

- **Relation instance** – A finite set of tuples in the relational database system represents relation instance. Relation instances do not have duplicate tuples.

- **Relation schema** – A relation schema describes the relation name (table name), attributes, and their names.

- **Relation key** – Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely.

- **Attribute domain** – Every attribute has some pre-defined value scope, known as attribute domain.

2) Database Schema and Instance

Ans:-

Database Schemas

- **Definition of schema**: Design of a database is called the schema. Schema is of three types:
  1. Physical schema
  2. logical schema
  3. 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.

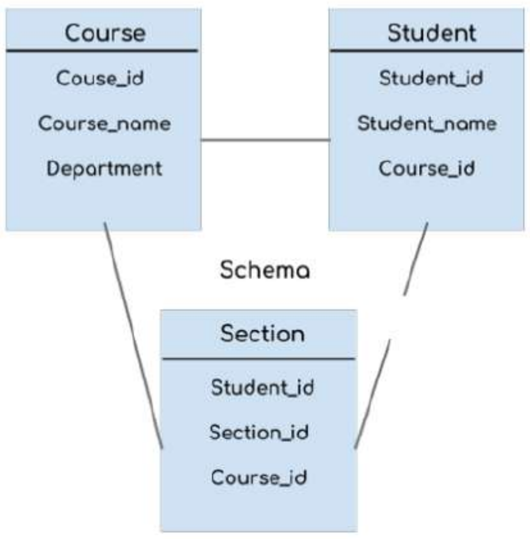
- Design of database at logical level is called **logical schema**, programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures, however the internal details such as implementation of data structure is 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.

Example

➢**For example:** In the following diagram, we have a schema that shows the relationship between three tables: Course, Student and Section.

➢The diagram only shows the design of the database, it doesn't show the data present in those tables. Schema is only a structural view(design) of a database as shown in the diagram .



Database Instances

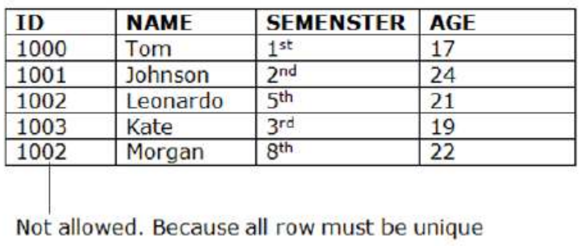
- **Definition of instance**: The data stored in database at a particular moment of time is called instance of database.
- Database schema defines the variable declarations in tables that belong to a particular database;
- The value of these variables at a moment of time is called the instance of that database.
- **For example:** lets say we have a single table student in the database, today the table has 100 records, so today the instance of the database has 100 records. Lets say we are going to add another 100 records in this table by tomorrow so the instance of database tomorrow will have 200 records in table. In short, at a particular moment the data stored in database is called the instance, that changes over time when we add or delete data from the database.

3) Key Constraints

Ans:-

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.
- **Example:**



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

All the best now do the rest…