SQL (STRUCTURED QUERY LANGUAGE)

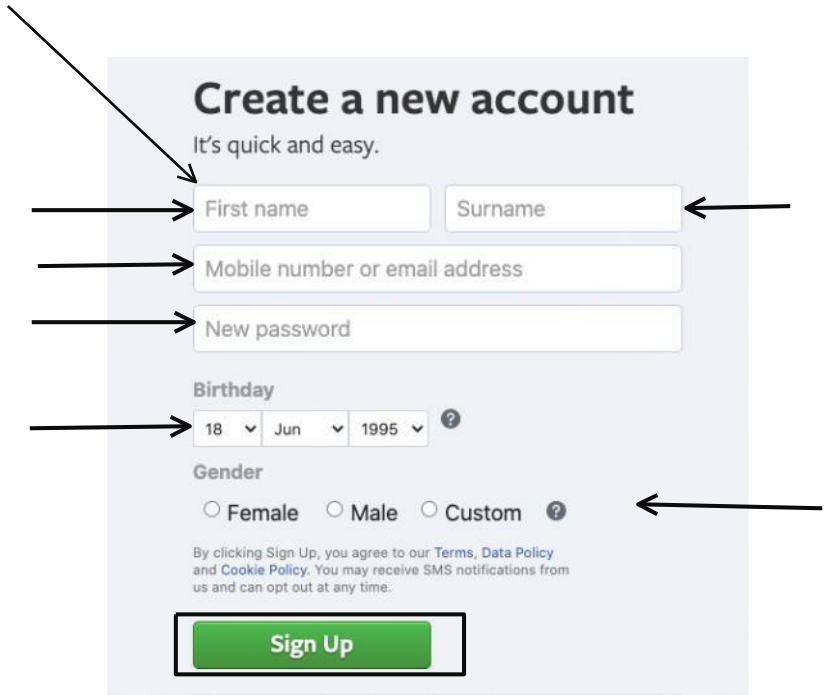
↓

DATABASE

What is DATA ?

"***Data is a raw-fact which describes the attributes of an Entity*** ".

Properties or Attributes



I am creating an account for myself :

Attributes

First name : Rohan
Surname : Singh
Phone number : 9876543210
Password : Rohan@123
Dob : 14-MAY-199X
Gender : MALE

Laptop
Brand : *Dell*
RAM : *8gb*
Touch : *no*

Laptop
Brand : *Apple*
RAM : *16gb*
Touch : *yes*

Example : Water Bottle

Entity

Height : 20cms

Color : blue

Capacity : 500ml

Attributes

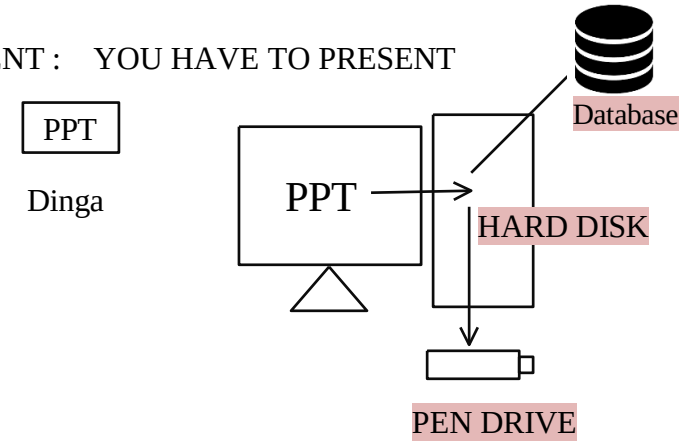
Contacts

Name : *Dinga*
Phone : 108
DOB: *14-feb-95*

DATABASE :

"***Database is a place or a medium in which we store the data in a Systematic and organized manner*** "

STUDENT : YOU HAVE TO PRESENT



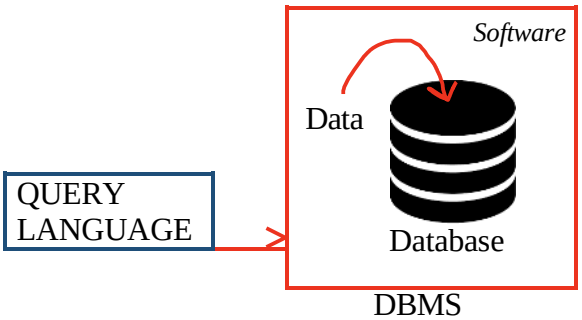
- The basic operations that can be performed on a database are
  - CREATE / INSERT
  - READ / RETRIEVE
  - UPDATE / MODIFY
  - DELETE / DROP

- These operations are referred as **"CRUD"** Operations .

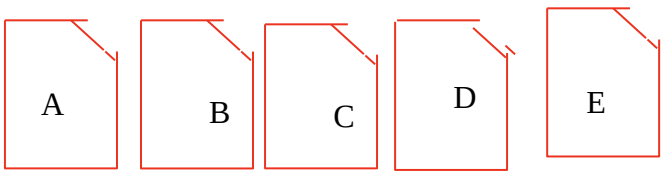
DATABASE MANAGEMENT SYSTEM (DBMS) :

"It is a software which is used to maintain and manage The database "

- **Security** and **authorization** are the two important features that DBMS provides .



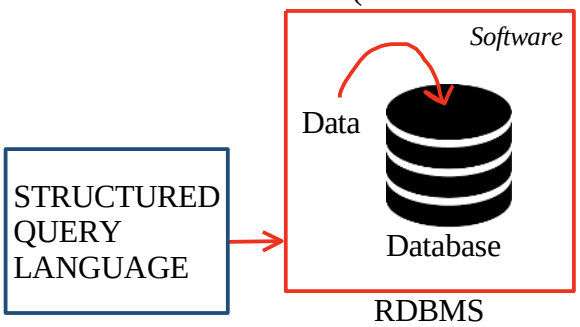
- We use query language to communicate or interact with DBMS
- DBMS stores the data in the form of *files* .



RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS):

"It is a type of DBMS software in which we store the data

In the form of Tables (rows & columns) ".



- We use SQL to communicate or interact with RDBMS
- RDBMS stores the data in the form of *Tables*.

Example :

Names
A
B
C
D
E

RELATIONAL MODEL :

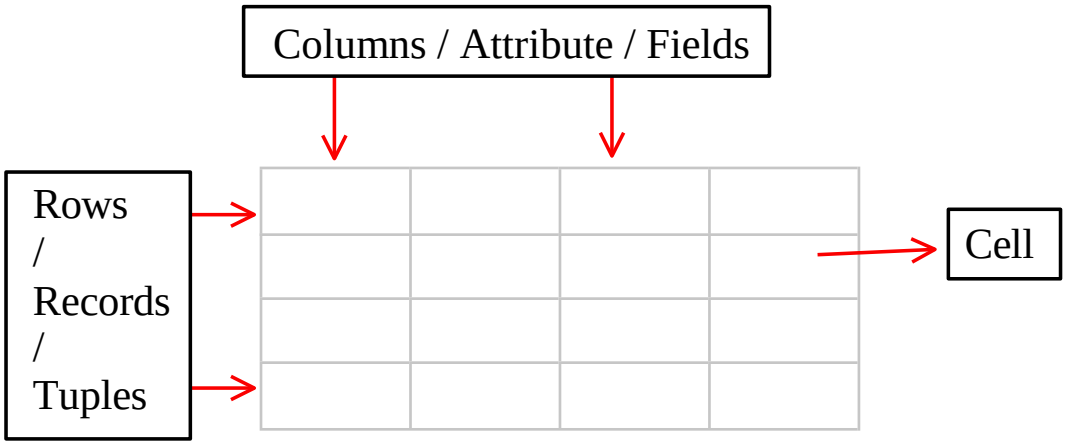
Relational Model was designed by ***E.F CODD*** .
In Relational Model we can store the data in the from of *tables* .

Any DBMS which follows Relational Model becomes RDBMS .

DBMS. → Relational Model → RDBMS

Any DBMS which follows rules of EF CODD becomes RDBMS .

TABLE : "It is a logical organization of data which consists of Columns &Rows .



Example :

Emp (Entity)
- Eid
- Ename
- Salary

Employee :

EID	ENAME	SALARY
1	SMITH	1000
2	ALLEN	1500
3	CLARK	2000

RULES OF E.F CODD :

1. The data entered into a cell must always be a *single valued data* .

   Example :

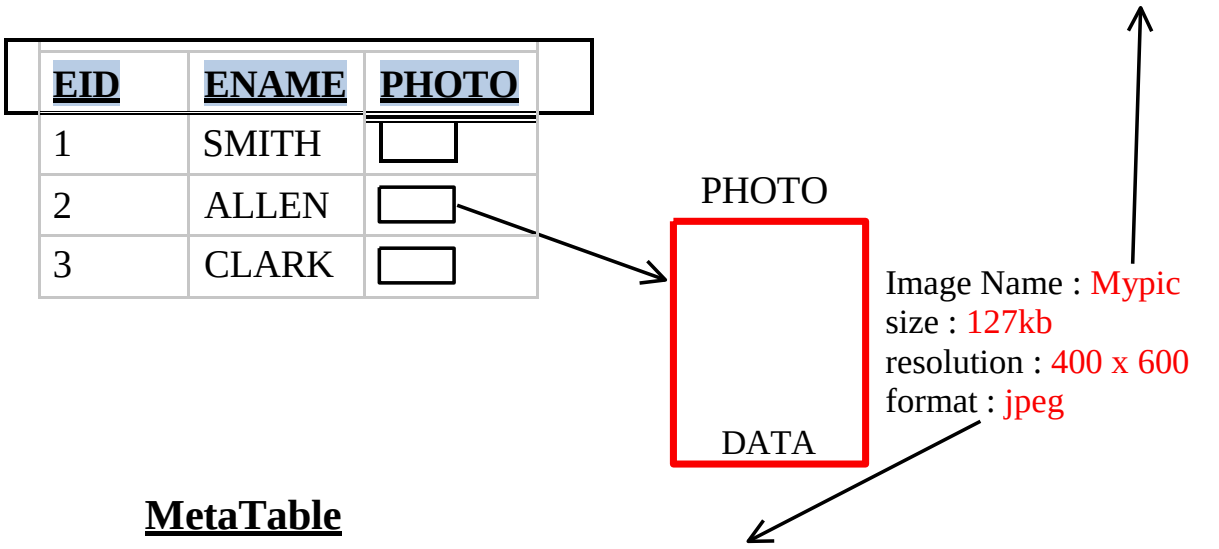
EID	ENAME	PHONE_NO
1	SMITH	101
2	ALLEN	102 , 202
3	CLARK	103

EID	ENAME	PHONE_NO	ALTERNATE NO
1	SMITH	101	
2	ALLEN	102	202
3	CLARK	103	

2. According to E.F CODD we can store the data in Multiple Tables , If needed we can establish a connection between the tables with the Help of *Key Attribute* .

3. In RDBMS we store everything in the from of tables including *Metadata* .

   Example : *Metadata* : The details about a data is knows as Metadata.



MetaTable

Image name	size	Format	Resolution
Mypic	127	jpeg	400 x 600

4. The data entered into the table can be validated in 2 steps .
    i. By assigning Datatypes .
    ii. By assigning Constraints .

   Datatypes are mandatory , whereas Constraints are Optional .

DATATYPES :

It is used to specify or determine the type of data that will be stored In a particular memory location .

Datatypes in SQL :

1. CHAR
2. VARCHAR / VARCHAR2
3. DATE
4. NUMBER
5. LARGE OBJECTS
    i. Character Large Object .
    ii. Binary Large Object .

NOTE : SQL is not a Case Sensitive Language .

1. **CHAR :** In character datatype we can store 'A-Z' , 'a-z' , '0-9' And Special Characters($, & , @ , ! …) .

- Characters must always be enclosed within single quotes ' '.
- Whenever we use char datatype we must mention size
- **Size** : it is used to specify number of characters it can store .
    - The maximum number of characters it can store is **2000ch**.
- Char follows fixed length memory allocation .

Syntax: CHAR (SIZE)

Example : CHAR (8)

R	**A**	**M**					

Used Memory ⟷ Unused Memory (memory wastage)

2. **VARCHAR :** In varchar datatype we can store 'A-Z' , 'a-z' , '0-9' And Special Characters($, & , @ , ! …) .

- Characters must always be enclosed within single quotes ' '.
- Whenever we use char datatype we must mention size
- **Size** : it is used to specify number of characters it can store .
  - The maximum number of characters it can store is **2000ch.**
- VarChar follows variable length memory allocation .

Syntax: VARCHAR (SIZE)

Example : VARCHAR (8)

R	**A**	**M**					

Used Memory ⟷ Unused Memory (free memory)

NOTE : **VARCHAR2 :** it is an updated version of varchar where in We can store up to **4000Ch.**

Syntax: VARCHAR2(SIZE)

Example :

STUDENT

USN	SNAME	ADDRESS	PAN_NO
CHAR(4)	VARCHAR(10)	VARCHAR(10)	CHAR(10)
RAM1	DINGA	BANGALORE	ABC123XYZ1
QSP2	DINGI	MYSORE	ABC123XYZ2

ASSIGNMENT :

1. DIFFERENTIATE BETWEEN CHAR & VARCHAR

ASCII : [American Standard Code For Information Interchange]

'A'	65
'Z'	90
'a'	97
'z'	122

3. **NUMBER :** It is used to store numeric values .

SYNTAX: **NUMBER** (Precision , **[** Scale **]**)

[ ] - Not Mandatory .

Precision: it is used to determine the number of digits used
To store integer value .

Scale: it is used to determine the number of digits used to store
Decimal (floating) value within the precision .

- Scale is not mandatory, and the default value of scale
  Is zero (0) .

Example :	Number (3)	+/- 999
Example :	Number (5 , 0)	+/- 99999
Example :	Number (5 , 2)	+/- 999.99
Example :	Number (7 , 3)	+/- 9999.999
Example :	Number (4 , 4)	+/- .9999
Example :	Number (5 , 4)	+/- 9.9999
Example :	Number (3 , 6)	+/- .000999
Example :	Number (5 , 8)	+/- .00099999
Example :	Number (2 , 7)	+/- .0000099

EID	**PHONE_NO**	**SALARY**
Number(3)	Number (10)	Number (7 , 2)
101	9876543210	9000.85

4. **DATE:** it is used to store dates in a particular format .

It used *Oracle specified Format* .

'DD-MON-YY'	OR	'DD-MON-YYYY'
'22-JUN-20'		'22-JUN-2020'

SYNTAX: **DATE**

Example :

DOB	**Hiredate**	**Anniversary**
Date	Date	Date
'01-JAN-1945'	'20-JUN-20'	'15-APR-2008'

5. LARGE OBJECTS

1. Character large object (CLOB) :

It is used to store characters up to 4 GB of size .

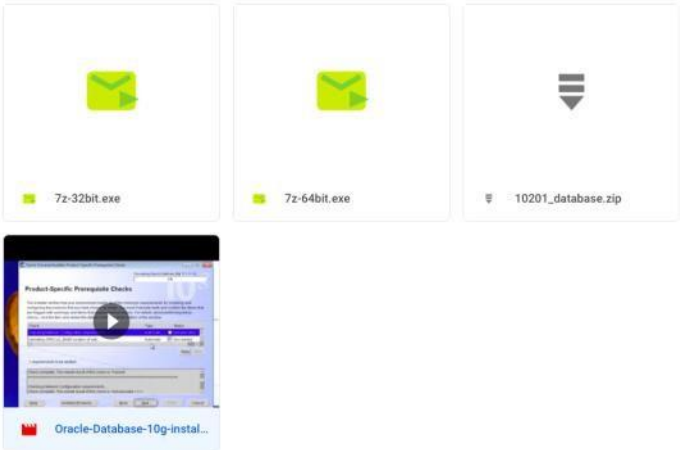
2. Binary large object (BLOB) :

It is used to store binary values of images , mp3 , mp4 Documents etc …. Up to 4GB of size .

NOTE :

FOR WINDOWS :

Software : **Oracle : Oracle 10g - Version**
Name : SQL*Plus
To download : bit.ly/roSoftWIN



If getting errors !!!

Gmail me on

:ramakrishankella117@gmail.com

Or

Send screenshot to INSTAGRAM : link2ram

CONSTRAINTS :

It is a rule given to a column for validation .

Types of Constraints :

1. UNIQUE
2. NOT NULL
3. CHECK
4. PRIMARY KEY
5. FOREIGN KEY .

1. **UNIQUE** : "*It is used to avoid duplicate values into the column* ".

2. **NOT NULL** : *"It is used to avoid Null "*.

3. **CHECK** : "*It is an extra validation with a condition*
   If the condition is satisfied then the value is accepted else Rejected ".

4. **PRIMARY KEY** : "*It is a constraint which is used to identify a record Uniquely from the table* " .

   Characteristics of Primary key :
   - We can have only 1 PK in a table
   - PK cannot accept duplicate / repeated values .
   - PK cannot accept Null
   - PK is always a combination of Unique and Not Null Constraint.

5. **FOREIGN KEY** : "*It is used to establish a connection between the The tables* "

   Characteristics of Foreign key :
   - We can have only Multiple FK in a table
   - FK can accept duplicate / repeated values .
   - FK can accept Null
   - FK is not a combination of Unique and Not Null Constraint.
   - For an Attribute (column) to become a FK ,it is mandatory That it must be a PK in its own table .

Example :

EMP

Primary key				
		Check (Salary >		**Check**
		0)		**(length(phone) = 10)**
Not Null	**Not Null**	**Not Null**	**Not Null**	**Not Null**

Unique				Unique
EID	**NAME**	**SALARY**	**DOJ**	**PHONE**
Number(2)	Varchar(10)	Number(7,2)	Date	Number(10)
1	**A**	**10000**	**'20-JUN-20'**	**9876543210**
2	**B**	**20000**	**'20-JUN-19'**	**9876543222**
3	**C**	**35000**	**'01-JAN-18'**	**9876543333**
4	**D**	**50000**	**'01-OCT-19'**	**9876511111**

Example for Foreign Key :

Emp

EID	NAME	SALARY	DNO FK	CID FK
1	A	10000	20	2
2	B	20000	10	3
3	C	35000	20	1
4	D	50000	10	2

Child Table

Customer

CID	CNAME	CNO
1	X	1001
2	Y	2002
3	Z	3003

Parent Table

Dept

DNO	DNAME	LOC
10	D1	L1
20	D2	L2

Parent Table

ASSIGNMENT :

1. Differentiate between Primary key and Foreign key .

PRIMARY KEY	FOREIGN KEY
It is used to identify a records Uniquely from the table.	It is used to establish a connection Between the tables
It cannot accept Null	It can accept Null
It cannot accept duplicate values	It can accept duplicate values
It is always a combination of Not Null and Unique constraint	It is not a combination of Not Null and Unique constraint
We can have only 1 PK in a table	We can have Multiple FK in a table

NOTE : NULL

Null Is a *keyword* which is used to represent Nothing / Empty Cell.

Characteristics of Null :

- Null doesn't represent 0 or Space .
- Any operations performed on a Null will result in Null itself

- Null doesn't Occupy any Memory .

- Null doesn't Occupy any Memory .
- We cannot Equate Null .

OVERVIEW OF SQL *STATEMENTS* :

1. DATA DEFINITION LANGUAGE (DDL)
2. DATA MANIPULATION LANGUAGE (DML)
3. TRANSCATION CONTROL LANGUAGE (TCL)
4. DATA CONTROL LANGUAGE (DCL)
5. DATA QUERY LANGUAGE (DQL)

STATEMENTS ARE CLASSIFIED INTO 5 DIFFERENT TYPES

- DATA DEFINITION LANGUAGE (DDL)
- DATA MANIPULATION LANGUAGE (DML)
- TRANSACTION CONTROL LANGUAGE (TCL)
- DATA CONTROL LANGUAGE (DCL)
- DATA QUERY LANGUAGE (DQL)

1. DATA DEFINITION LANGUAGE (DDL):

" *DDL is used to construct an object in the database and deals with the Structure of the Object* "

It has 5 statements :

1. CREATE
2. RENAME
3. ALTER
4. TRUNCATE
5. DROP

{Dr.CAT}

1. **CREATE :** " IT IS USED TO BUILD / CONSTRUCT AN OBJECT "

Object / Entity can be a Table or a View (Virtual Table) .

How to Create a Table :

- Name of the table
  - Tables cannot have same names .
- Number of Columns .
- Names of the columns .
- Assign datatypes for the Columns.
- Assign Constraints [NOT MANDATORY] .

Example 1:

Table_Name : **CUSTOMER**
Number of Columns : **4**

Customer

Column_Name	CID	CNAME	CNO	ADDRESS
Datatypes	Number(2)	Varchar(10)	Number (10)	Varchar(15)
Null / Not Null	Not Null	Not Null	Not Null	Null
Unique	Unique		Unique	
Check			Check (length(CNO) = 10)	
Primary Key	Primary Key			
Check			Check (length(CNO) = 10)	
Primary Key	Primary Key			
Foreign Key				

Not Mandatory

Syntax to create a table :

```
CREATE TABLE Table_Name (
        Column_Name1  datatype  constraint_type ,
        Column_Name2  datatype  constraint_type ,
        Column_Name3  datatype  constraint_type ,
        .
        .
        Column_NameN  datatype  constraint_type
    );
```

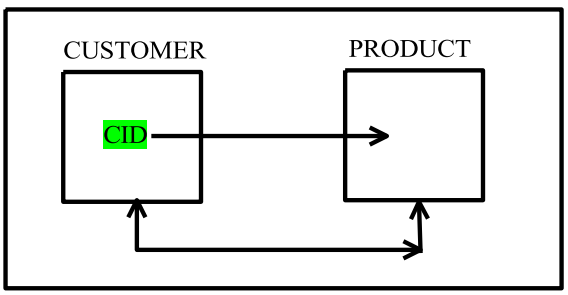
Example :

```
CREATE TABLE CUSTOMER
    (
        CID  Number(2)  primary key ,
        CNAME Varchar(10) ,
        CNO Number(10) not null check( length( CNO ) = 10 ) ,
        ADDRESS Varchar(15)
    );
```

NOTE :

To Describe the table:

```
Syntax:    DESC Table_Name ;
```



Example 2:

Table_Name : **PRODUCT**
Number of Columns : 4

Product

Column_Name	**PID**	**PNAME**	**PRICE**	**CID**
Datatypes	Number(2)	Varchar(10)	Number (7,2)	Number(2)
Null / Not Null	Not Null	Not Null	Not Null	Null
Unique	Unique			
Check			Check (Price > 0)	
Primary Key	Primary Key			
Foreign Key				**Foreign Key**

Syntax to create a table :

```
CREATE TABLE Table_Name (
        Column_Name1 datatype constraint_type ,
        Column_Name2 datatype constraint_type ,
        Column_Name3 datatype constraint_type ,
        .
        .
        Column_NameN datatype ,
        Constraint Foreign key references Parent_Table_Name(Column_Name)
    );
```

Example :

```
CREATE TABLE PRODUCT
    (
        PID Number(2) primary key ,
        PNAME Varchar(10) ,
        PRICE Number(7,2) check( Price > 0) ,
        CID Number(2) ,
        Constraint CID_FK Foreign Key(CID) references CUSTOMER( CID )
    );
```

2. **RENAME :** "IT IS USED TO CHANGE THE NAME OF THE OBJECT "

Syntax: RENAME Table_Name TO New_Name ;

Example :

```
RENAME Customer TO Cust ;
```

3. **ALTER :**" IT IS USED TO MODIFY THE STRUCTURE OF THE TABLE "

- **TO ADD A COLUMN :**

Syntax: ALTER TABLE Table_Name
ADD Column_Name Datatype Constraint_type ;

Example :
```
ALTER TABLE Cust
    ADD MAIL_ID Varchar(15) ;
```

- **TO DROP A COLUMN :**

Syntax: ALTER TABLE Table_Name
DROP COLUMN Column_Name ;

Example : ALTER TABLE Cust
DROP COLUMN MAIL_ID ;

- **TO RENAME A COLUMN :**

Syntax: ALTER TABLE Table_Name
RENAME COLUMN Column_Name TO new_Column_Name ;

Example : ALTER TABLE Cust
RENAME COLUMN CNO TO PHONE_NO ;

- **TO MODIFY THE DATATYPE :**

Syntax: ALTER TABLE Table_Name
MODIFY COLUMN_NAME New_Datatype;

Example : ALTER TABLE Cust
MODIFY CNAME CHAR(10) ;

- **TO MODIFY NOT NULL CONSTRAINTS :**

Syntax: ALTER TABLE Table_Name
MODIFY COLUMN_NAME Existing_datatype [NULL]/NOT NULL;

Example : ALTER TABLE Cust
MODIFY ADDRESS Varchar(15) Not Null ;

4. **TRUNCATE :** " IT IS USED TO REMOVE ALL THE RECORDS FROM THE TABLE PREMANENTLY "

Syntax: TRUNCATE TABLE Table_Name ;

Cust

Cid	Cname	Phone_no	Address
1	A	1234567890	BANGALORE
2	B	1234567899	MYSORE
3	C	1234567880	MANGALORE

Example : TRUNCATE TABLE Cust ;

Cust

Cid	Cname	Phone_no	Address

5. **DROP :**" IT IS USED TO REMOVE THE TABLE FROM THE DATABASE "

Syntax: DROP TABLE Table_Name ;

Example :

DATABASE

Cust

Cid	Cname	Phone_no	Address
1	A	1234567890	BANGALORE
2	B	1234567899	MYSORE
3	C	1234567880	MANGALORE

Bin Folder

Bin$123abc

DROP

Address

TO RECOVER THE TABLE :

Syntax: FLASHBACK TABLE Table_Name
TO BEFORE DROP ;

Example :

DATABASE

Address : BIN$123ABCXYZ

Bin Folder

Cust

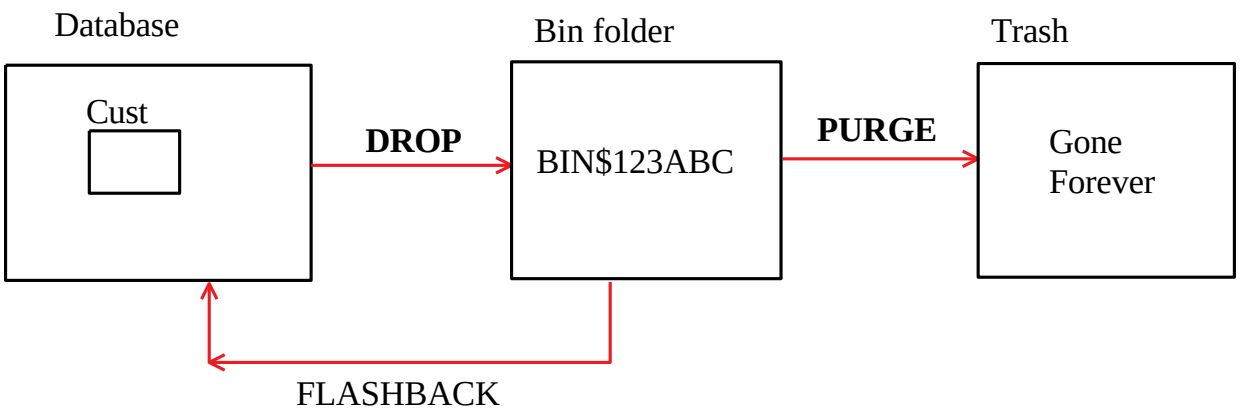
FLASHBACK

FLASHBACK TABLE Cust
TO BEFORE DROP ;

TO DELETE THE TABLE FROM BIN FOLDER :

Syntax: PURGE TABLE Table_Name ;

Example : PURGE TABLE Cust ;



FLASHBACK

NOTE : DDL STATEMENTS ARE AUTO-COMMIT STATEMENTS

DATA MANIPULATION LANGUAGE (DML)

It is used to Manipulate the Object by performing insertion , updating and deletion .

1. INSERT
2. UPDATE
3. DELETE

1. **INSERT :** It is used to insert / create records in the table .

Syntax: INSERT INTO Table_Name VALUES(v1 , v2 , v3 ……) ;

CUSTOMER

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)

INSERT INTO CUSTOMER VALUES(1 , 'DINGA' , 9876543210 ,'BANGALORE') ;

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	DINGA	9876543210	BANGALORE

INSERT INTO CUSTOMER VALUES(2 , 'DINGI' , 9876543211 ,'MANGALORE') ;

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	DINGA	9876543210	BANGALORE
2	DINGI	9876543211	MANGALORE

PRODUCT

PID	PNAME	PRICE	CID
NUMBER(2)	VARCHAR(10)	NUMBER(6,2)	NUMBER(3)

INSERT INTO PRODUCT VALUES(11 , 'iPhone' , 10000 , 2);

PID	PNAME	PRICE	CID
NUMBER(2)	VARCHAR(10)	NUMBER(6,2)	NUMBER(3)
11	iPhone	10000	2

INSERT INTO PRODUCT VALUES(22 , 'Mac Book' , 20000 , 1);

PID	PNAME	PRICE	CID
NUMBER(2)	VARCHAR(10)	NUMBER(6,2)	NUMBER(3)
11	iPhone	10000	2
22	Mac Book	20000	1

2. **UPDATE :** It is used to modify an existing value .

```
Syntax:  UPDATE Table_Name
                SET  Col_Name = Value , Col_Name = Value ,,,,,
                 [WHERE stmt ] ;
```

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	ABHI	1234567890	BANGALORE
2	ABDUL	9876543210	MANGALORE

- WAQT update the phone number of Abdul to 7778889994

```
UPDATE CUSTOMER
SET CNO = 7778889994
WHERE CNAME ='ABDUL' ;
```

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	ABHI	1234567890	BANGALORE
2	ABDUL	7778889994	MANGALORE

- WAQT change the address of the customer to Mysore whose cid is 1 .

```
UPDATE CUSTOMER
SET ADDRESS = 'MYSORE'
WHERE CID = 1 ;
```

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	ABHI	1234567890	MYSORE
2	ABDUL	7778889994	MANGALORE

3. **DELETE :** It is used to remove a particular record from the table .

```
Syntax: DELETE FROM Table_Name
            [ WHERE stmt ];
```

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	ABHI	1234567890	BANGALORE

2	ABDUL	1234567891	MANGALORE

- WAQT remove abdul from the list of customers .

```
DELETE FROM CUSTOMER
WHERE CNAME ='ABDUL' ;
```

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	ABHI	1234567890	BANGALORE

ASSIGNMENT ON DML STATEMENTS :

1. WAQT update the salary of employee to double their salary if He is working as a manager .

2. WAQT change the name of SMITH to SMIITH .

3. WAQT modify the job of KING to 'PRESIDENT' .

4. WAQT to change name of ALLEN to ALLEN MORGAN .

5. WAQT hike the salary of the employee to 10% . If employees earn less than 2000 as a salesman .

6. WAQ TO delete the employees who don’t earn commission .

7. WAQ to remove all the employees hired before 1987 in dept 20

1. TRANSACTION CONTROL LANGUAGE (TCL)

We have 3 Statements :

1. COMMIT
2. ROLLBACK
3. SAVEPOINT

1. COMMIT : "This statement is used to SAVE the transactions into the DB ".

Syntax: **COMMIT ;**

Example :

SYNTAX: **ROLLBACK TO** Savepoint_Name ;

2. DATA CONTROL LANGUAGE :

"This statement is used to control the flow of data between the users ".

We have 2 statements :

1. GRANT
2. REVOKE

DATA QUERY LANGUAGE (DQL) :

" DQL is used to retrieve the data from the database " .

It had 4 statements :

1. SELECT
2. PROJECTION
3. SELECTION
4. JOIN

1. **SELECT :** "It is used to retrieve the *data* from the table and display it.

2. **PROJECTION :** "It is a process of retrieving the data by *selecting only the columns* is known as Projection " .
   - In projection all the records / values present in a particular column are by default selected .

3. **SELECTION :** "It is a process of retrieving the data by *selecting both the columns and rows* is known as Selection " .

4. **JOIN :**"It is a process of retrieving the data from *Multiple tables* simultaneously is known as Join " .

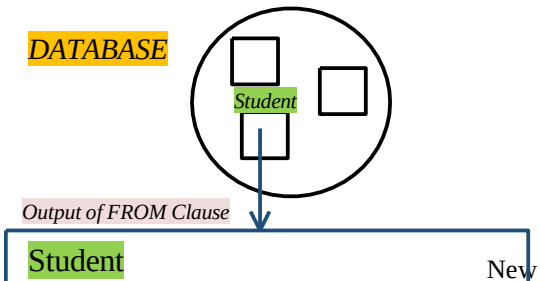
PROJECTION

- "It is a process of retrieving the data by *selecting only the columns* is known as Projection " .
- In projection all the records / values present in a particular column are by default selected .

SYNTAX :
SELECT * / [DISTINCT] Column_Name / Expression [ALIAS]
FROM Table_Name ;

ORDER OF EXECUTION
1. FROM Clause
2. SELECT Clause

Example : Write a query to display names of all the students .



SELECT SNAME
FROM STUDENT ;

SID	SNAME	BRANCH	PER
1	A	ECE	60
2	B	CSE	75
3	C	ME	50
4	D	ECE	80
5	C	CSE	75
6	E	CIVIL	95

Output of SELECT Clause

SNAME
A
B
C
D
C
E

NOTE :

- FROM Clause starts the execution .
- For FROM Clause we can pass Table_Name as an argument .
- The job of FROM Clause is to go to the Database and search for the table and put the table under execution .
- SELECT Clause will execute after the execution of FROM Clause
- For SELECT Clause we pass 3 arguments
  - *
  - Column_Name
  - Expression
- The job of SELECT Clause is to go the table under execution and select the columns mentioned .
- SELECT Clause is responsible for preparing the result table .
- Asterisk(*) : it means to select all the columns from the table .
- Semicolon : it means end of the query .

- WAQTD student id and student names for all the students.

```
SELECT SID , SNAME
FROM STUDENT ;
```

- WAQTD name and branch of all the students .

```
SELECT  SNAME , BRANCH
FROM STUDENT ;
```

- WAQTD NAME , BRANCH AND PERCENTAGE FOR ALL THE STUDENTS .

```
SELECT SNAME , BRANCH , PER
FROM STUDENT ;
```

- WAQTD details of all the students from students table .

SELECT *
FROM STUDENT ;

- WAQTD sname , sid , per , branch of all the students .

SELECT SNAME , SID , PER , BRANCH
FROM STUDENT ;

EMP Table :

EMPNO	ENAME	JOB	HIREDATE	MGR	SAL	COMM	DEPTNO
7369	SMITH	CLERK	17-DEC-80	7902	800		20
7499	ALLEN	SALESMAN	20-FEB-81	7698	1600	300	30
7521	WARD	SALESMAN	22-FEB-81	7698	1250	500	30
7566	JONES	MANAGER	02-APR-81	7839	2975		20
7654	MARTIN	SALESMAN	28-SEP-81	7698	1250	1400	30
7698	BLAKE	MANAGER	01-MAY-81	7839	2850		30
7782	CLARK	MANAGER	09-JUN-81	7839	2450		10
7788	SCOTT	ANALYST	19-APR-87	7566	3000		20
7839	KING	PRESIDENT	17-NOV-81		5000		10
7844	TURNER	SALESMAN	08-SEP-81	7698	1500	0	30
7876	ADAMS	CLERK	23-MAY-87	7788	1100		20
7900	JAMES	CLERK	03-DEC-81	7698	950		30
7902	FORD	ANALYST	03-DEC-81	7566	3000		20
7934	MILLER	CLERK	23-JAN-82	7782	1300		10

- **WAQTD name salary and commission given to all the employees .**

Select ename , sal , comm
From emp ;

- **WAQTD name of the employee along with their date of joining .**

Select ename , hiredate
From emp ;

DEPT :

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

- **WAQTD dname and location for all the depts .**

Select dname , loc
From dept ;

QUESTIONS ON EMP AND DEPT TABLE:

1.WRITE A QUERY TO DISPLAY ALL THE DETAILS FROM THE

EMPLOYEE TABLE.
2. WAQTD NAMES OF ALL THE EMPLOYEES.
3. WAQTD NAME AND SALARY GIVEN TO ALL THE EMPLOYEES.
4. WAQTD NAME AND COMMISSION GIVEN TO ALL THE EMPLOYEES.
5. WAQTD EMPLOYEE ID AND DEPARTMENT NUMBER OF ALL THE EMPLOYEES
IN EMP TABLE.
6. WAQTD ENAME AND HIREDATE OF ALL THE EMPLOYEES .
7. WAQTD NAME AND DESIGNATION OF ALL THE EMPLPOYEES .
8. WAQTD NAME , JOB AND SALARY GIVEN ALL THE EMPLOYEES.
9. WAQTD DNAMES PRESENT IN DEPARTMENT TABLE.
10. WAQTD DNAME AND LOCATION PRESENT IN DEPT TABLE.

Assignments have to Mailed TO : ramakrishnakella117@gmail.com

DISTINCT Clause

" It is used to remove the duplicate or repeated values from the Result table " .

- Distinct clause has to be used As the first argument to select clause .
- We can use multiple columns As an argument to distinct clause, it will remove the combination of columns in which the records are duplicated .

Example :

Student

SID	SNAME	BRANCH	PER
1	A	ECE	60
2	B	CSE	75
3	C	ME	50
4	D	ECE	80
5	C	CSE	75
6	E	CIVIL	95

- SELECT SNAME
  FROM STUDENT ;

SNAME
A
B
C
D
C
E

- SELECT DISTINCT SNAME
  FROM STUDENT ;

SNAME
A
B
C
D
~~C~~
E

→

SNAME
A
B
C
D
E

- SELECT DISTINCT BRANCH
  FROM STUDENT ;

BRANCH
ECE
CSE
ME
ECE
CSE
CIVIL

→

BRANCH
ECE
CSE
ME
CIVIL

- SELECT DISTINCT PER
  FROM STUDENT ;

PER
60
75
50
80
75
95

→

PER
60
75
50
80
95

- SELECT DISTINCT BRANCH , PER

FROM STUDENT ;

BRANCH	**PER**
ECE	60
CSE	75
ME	50
ECE	80
~~CSE~~	~~75~~
CIVIL	95

BRANCH	**PER**
ECE	60
CSE	75
ME	50
ECE	80
CIVIL	95

EXPRESSION

"A statement which gives result is known as Expression ".

Expression is a combination Operand and Operator .

Operand : These are the values that we pass .

Operator : These are the Symbols which perform some Operation on
The Operand .

Example : 5 * 10

EMP

EID	ENAME	SAL
1	A	100
2	B	200
2	C	100

1. WAQTD name and salary given to the employees .

   SELECT ENAME , SAL
   FROM EMP ;

2. WAQTD name and annual salary of the employees .

   SELECT ENAME , SAL * 12
3. FROM EMP ;

ENAME	SAL*12
A	1200
B	2400
C	1200

4. WAQTD all the details of the employee along with annual salary

   Select eid , ename , sal , sal*12
   From emp ;

   Select emp.* , sal*12
   From emp ;

5. WAQTD name and salary with a hike of 20% .

   Select ename , Sal + Sal*20/100
   From emp ;

Formulae to calculate percentage :

Sal + Sal * a / 100	Sal * 1.a

6. WAQTD name and salary of an employee with a deduction Of 10% .

   Select ename , sal - sal * 10 /100
   From emp ;

ALIAS

"It is an alternate name given to a Column or an Expression In the result table " .

- We can assign alias name with or without using 'As' keyword .
- Alias names have to be a single string which is separated by An underscore or enclosed within double quotes .

Example :	ANNUAL_SALARY
	"ANNUAL SALARY"

- WAQTD annual salary for all the employees .

  Select sal*12
  From emp ;

SAL*12
1200
2400
1200

  Select sal*12 Annual_Salary
  From emp ;

Annual Salary
1200
2400
1200

  Select sal + sal * 10 / 100 Hike
  From emp ;

- WAQTD name and salary with a deduction 32% .

  Select Ename , sal-sal*32/100 as deduction
  From emp ;

ASSIGNMENT ON EXPRESSION & ALIAS

1. WAQTD NAME OF THE EMPLOYEE ALONG WITH THEIR ANNUAL SALARY.
2. WAQTD ENAME AND JOB FOR ALL THE EMPLOYEE WITH THEIR HALF TERM SALARY.

3. WAQTD ALL THE DETAILS OF THE EMPLOYEES ALONG WITH AN ANNUALBONUS OF 2000.
4. WAQTD NAME SALARY AND SALARY WITH A HIKE OF 10%.
5. WAQTD NAME AND SALARY WITH DEDUCTION OF 25%.
6. WAQTD NAME AND SALARY WITH MONTHLY HIKE OF 50.
7. WAQTD NAME AND ANNUAL SALARY WITH DEDUCTION OF 10%.
8. WAQTD TOTAL SALARY GIVEN TO EACH EMPLOYEE (SAL+COMM).
9. WAQTD DETAILS OF ALL THE EMPLOYEES ALONG WITH ANNUAL SALARY.
10. WAQTD NAME AND DESIGNATION ALONG WITH 100 PENALTY IN SALARY.

SELECTION :

"It is a process of retrieving the data by *selecting both the columns and rows* is known as Selection " .

SYNTAX :
SELECT * / [DISTINCT] Column_Name / Expression [ALIAS]
FROM Table_Name
WHERE <*Filter_Condition*> ;

ORDER OF EXECUTION

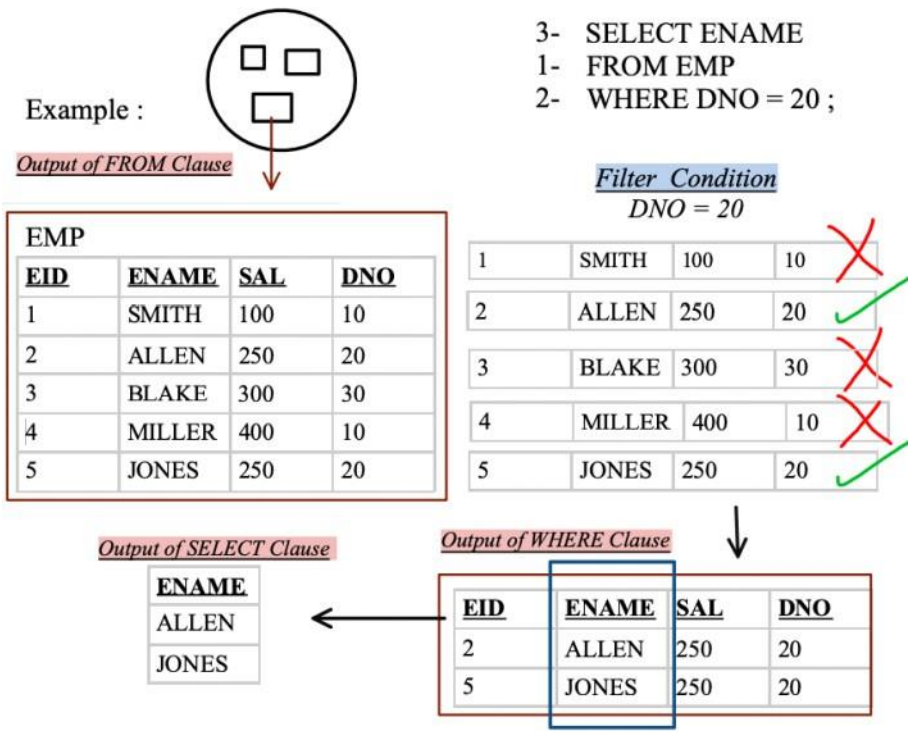
1. FROM
2. WHERE
3. SELECT

WHERE Clause

"Where clause is used to filter the records ".

Example :

- WAQTD names of the employees working in dept 20 .



- WAQTD names of the employees getting salary More than 300 .

```
SELECT ENAME
FROM EMP
WHERE SAL > 300 ;
```

- WAQTD names and salary of the employees working in dept 10.

```
SELECT ENAME , SAL
FROM EMP
WHERE DEPTNO = 10  ;
```

- WAQTD all the details of the employees whose salary is Less than 1000 rupees .

```
SELECT *
FROM EMP
WHERE SAL < 1000 ;
```

EMP *:*

EMPNO	ENAME	JOB	HIREDATE	MGR	SAL	COMM	DEPTNO
7369	SMITH	CLERK	17-DEC-80	7902	800		20
7499	ALLEN	SALESMAN	20-FEB-81	7698	1600	300	30
7521	WARD	SALESMAN	22-FEB-81	7698	1250	500	30
7566	JONES	MANAGER	02-APR-81	7839	2975		20
7654	MARTIN	SALESMAN	28-SEP-81	7698	1250	1400	30
7698	BLAKE	MANAGER	01-MAY-81	7839	2850		30
7782	CLARK	MANAGER	09-JUN-81	7839	2450		10
7788	SCOTT	ANALYST	19-APR-87	7566	3000		20
7839	KING	PRESIDENT	17-NOV-81		5000		10
7844	TURNER	SALESMAN	08-SEP-81	7698	1500	0	30
7876	ADAMS	CLERK	23-MAY-87	7788	1100		20
7900	JAMES	CLERK	03-DEC-81	7698	950		30
7902	FORD	ANALYST	03-DEC-81	7566	3000		20
7934	MILLER	CLERK	23-JAN-82	7782	1300		10

- WAQTD name and hiredate of an employee hired on '09-JUN-1981'

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE  DATE = '09-JUN-1981' ;
```

- WAQTD details of the employee whose name is 'Miller'

```
SELECT *
FROM EMP
WHERE ENAME ='MILLER'  ;
```

- WAQTD details of the employee hired after '01-JAN-1982'

```
SELECT *
FROM EMP
WHERE HIREDATE > '01-JAN-1982' >  ;
```

- WAQTD name sal and hiredate of the employees who were Hired before 1985 .

```
SELECT ENAME , SAL , HIREDATE
FROM EMP
WHERE HIREDATE < '01-JAN-1985' ;
```

- WAQTD name sal and hiredate of the employees who were Hired after 1985 .

```
SELECT ENAME , SAL , HIREDATE
FROM EMP
WHERE HIREDATE  > '31-DEC-1985' ;
```

- WAQTD name of the employees who was hired on Valentine's day 2020 .

```
SELECT ENAME
```

FROM EMP
WHERE HIREDATE = '14-FEB-2020' ;

ASSIGNMENT ON WHERE Clause .

1. WAQTD THE ANNUAL SALARY OF THE EMPLOYEE WHOS NAME IS SMITH

2. WAQTD NAME OF THE EMPLOYEES WORKING AS CLERK

3. WAQTD SALARY OF THE EMPLOYEES WHO ARE WORKING AS SALESMAN

4. WAQTD DETAILS OF THE EMP WHO EARNS MORE THAN 2000

5. WAQTD DETAILS OF THE EMP WHOS NAME IS JONES

6. WAQTD DETAILS OF THE EMP WHO WAS HIRED AFTER 01-JAN-81

7. WAQTD NAME AND SAL ALONG WITH HIS ANNUAL SALARY IF THE ANNUAL SALARY IS MORE THAN 12000

8. WAQTD EMPNO OF THE EMPLOYEES WHO ARE WORKING IN DEPT 30

9. WAQTD ENAME AND HIREDATE IF THEY ARE HIRED BEFORE 1981

10. WAQTD DETAILS OF THE EMPLOYEES WORKING AS MANAGER

11. WAQTD NAME AND SALARY GIVEN TO AN EMPLOYEE IF EMPLOYEE EARNS A COMMISSION OF RUPEES 1400

12. WAQTD DETAILS OF EMPLOYEES HAVING COMMISSION MORE THAN SALARY

13. WAQTD EMPNO OF EMPLOYEES HIRED BEFORE THE YEAR 87

14. WAQTD DETAILS OF EMPLOYEES WORKING AS AN N ANALYST

15. WAQTD DETAILS OF EMPS EARNING MORE THAN 2000 RUPEES PER MONTH

COMMANDS ON SQL*Plus :

1. CLEAR SCREEN [**CL SCR**] : *To clear the screen*
2. SET LINES 100 PAGES 100 : *To set the dimensions of the output page .*

3. EXIT / QUIT : *To Close the Software* .
4. When account is Locked !!!
    - Log in as SYSTEM
    - Password TIGER
    - ALTER USER SCOTT ACCOUNT UNLOCK ;
    - ALTER USER SCOTT IDENTIFIED BY TIGER ;
5. SELECT * FROM TAB ;
    - **EMP**
    - **DEPT**
    - SALGRADE
    - BONUS

OPERATORS IN SQL

1. ARITHEMATIC OPERATORS :- (+ , - , * , /)
2. CONCATENATION OPERATOR :- (||)
3. COMPARISION OPERATORS :- (= , != or <>)
4. RELATIONAL OPERATOR :- (> , < , >= , <=)
5. LOGICAL OP : (AND , OR , NOT)
6. SPECIAL OPERATOR :-
    1. IN
    2. NOT IN
    3. BETWEEN
    4. NOT BETWEEN
    5. IS
    6. IS NOT
    7. LIKE
    8. NOT LIKE
7. SUBQUERY OPERATORS:-
    1. ALL
    2. ANY
    3. EXISTS
    4. NOT EXISTS

CONCATENATION Operator :

" It is used to join the strings ".

Symbol : ||

Example :
```
SELECT ENAME
FROM EMP
WHERE JOB ='MANAGER' ;
```

Ename
ALLEN
MARTIN
SMITH

```
SELECT 'Hi ' || ename
FROM EMP
WHERE JOB ='MANAGER' ;
```

Ename
Hi ALLEN
Hi MARTIN
Hi SMITH

- WAQTD name and deptno of the employees hired After '01-JAN-87' .

  SELECT ENAME , DEPTNO
  FROM EMP
  WHERE HIREDATE > '01-JAN-1987' ;

- WAQTD name and hiredate of the employees hired before 31-JUL-88

  SELECT ENAME , HIREDATE
  FROM EMP
  WHERE HIREDATE < '31-JUL-88' ;

LOGICAL OPERATORS

1. AND
2. OR
3. NOT

We use logical operators to write multiple conditions .

1. WAQTD name and deptno along with job for the employee working in dept 10 .

   SELECT ENAME , DEPTNO , JOB
   FROM EMP
   WHERE DEPTNO = 10 ;

2. WAQTD name and deptno along with job for the employee working as manager in dept 10 .

   SELECT ENAME , DEPTNO , JOB
   FROM EMP
   WHERE JOB ='MANAGER' AND DEPTNO = 10 ;

3. WAQTD name , deptno , salary of the employee working in dept 20 and earning less than 3000 .

   SELECT ENAME, DEPTNO , SAL
   FROM EMP
   WHERE DEPTNO = 20 AND SAL < 3000 ;

4. WAQTD name and salary of the employee if emp earns More than 1250 but less than 3000 .

   SELECT ENAME , SAL
   FROM EMP
   WHERE SAL > 1250 AND SAL < 3000 ;

5. WAQTD name and deptno of the employees if the works in dept 10 or 20 .

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE DEPTNO = 10 OR DEPTNO = 20 ;
```

6. WAQTD name and sal and deptno of the employees If emp gets more than 1250 but less than 4000 and works in dept 20 .

```
SELECT ENAME , SAL , DEPTNO
FROM EMP
WHERE SAL > 1250 AND SAL < 4000 AND DEPTNO
=20 ;
```

7. WAQTD name , job , deptno of the employees working as a manager in dept 10 or 30 .

```
SELECT ENAME , JOB , DEPTNO
FROM EMP
WHERE JOB ='MANAGER' AND ( DEPTNO = 10 OR
DEPTNO = 20 ) ;
```

8. WAQTD name , deptno , job of the employees working in dept 10 or 20 or 30 as a clerk .

```
SELECT ENAME , JOB , DEPTNO
FROM EMP
WHERE JOB ='CLERK' AND ( DEPTNO = 10 OR
DEPTNO = 20 AND DEPTNO = 30 ) ;
```

9. WAQTD name , job and deptno of the employees working as clerk or manager in dept 10 .

```
SELECT ENAME , JOB , DEPTNO
FROM EMP
WHERE ( JOB = 'CLERK' OR JOB ='MANAGER' )
AND DEPTNO = 10 ;
```

10. WAQTD name , job , deptno , sal of the employees working as clerk or salesman in dept 10 or 30 and earning more than 1800 .

```
SELECT ENAME , JOB , SAL
FROM EMP
WHERE ( JOB ='CLERK' OR JOB ='SALESMAN')
AND ( DEPTNO = 10 OR DEPTNO = 30 ) AND SAL >
1800 ;
```

<u>ASSIGNMENT ON LOGICAL OPERATORS :</u>

1. WAQTD DETAILS OF THE EMPLOYEES WORKING AS CLERK AND EARNING LESS THAN 1500
2. WAQTD NAME AND HIREDATE OF THE EMPLOYEES WORKING AS MANAGER IN DEPT 30

3. WAQTD DETAILS OF THE EMP ALONG WITH ANNUAL SALARY IF THEY ARE WORKING IN DEPT 30 AS SALESMAN AND THEIR ANNUAL SALARY HAS TO BE GREATER THAN 14000.
4. WAQTD ALL THE DETAILS OF THE EMP WORKING IN DEPT 30 OR AS ANALYST
5. WAQTD NAMES OF THE EMPMLOYEES WHOS SALARY IS LESS THAN 1100 AND THEIR DESIGNATION IS CLERK
6. WAQTD NAME AND SAL , ANNUAL SAL AND DEPTNO IF DEPTNO IS 20 EARNING MORE THAN 1100 AND ANNUAL SALARY EXCEEDS 12000
7. WAQTD EMPNO AND NAMES OF THE EMPLOYEES WORKING AS MANAGER IN DEPT 20
8. WAQTD DETAILS OF EMPLOYEES WORKING IN DEPT 20 OR 30 .
9. WAQTD DETAILS OF EMPLOYEES WORKING AS ANALYST IN DEPT 10 .
10. WAQTD DETAILS OF EMPLOYEE WORKING AS PRESIDENT WITH SALARY OF RUPEES 4000
11. WAQTD NAMES AND DEPTNO , JOB OF EMPS WORKING AS CLERK IN DEPT 10 OR 20
12. WAQTD DETAILS OF EMPLOYEES WORKING AS CLERK OR MANAGER IN DEPT 10 .
13. WAQTD NAMES OF EMPLOYEES WORKING IN DEPT 10 , 20 , 30 , 40 .
14. WAQTD DETAILS OF EMPLOYEES WITH EMPNO 7902,7839.
15. WAQTD DETAILS OF EMPLOYEES WORKING AS MANAGER OR SALESMAN OR CLERK
16. WAQTD NAMES OF EMPLOYEES HIRED AFTER 81 AND BEFORE 87
17. WAQTD DETAILS OF EMPLOYEES EARNING MORE THAN 1250 BUT LESS THAN 3000
18. WAQTD NAMES OF EMPLOYEES HIRED AFTER 81 INTO DEPT 10 OR 30
19. WAQTD NAMES OF EMPLOYEES ALONG WITH ANNUAL SALARY FOR THE EMPLOYEES WORKING AS MANAGER OR CLERK INTO DEPT 10 OR 30
20. WAQTD ALL THE DETAILS ALONG WITH ANNUAL SALARY IF SAL IS BETWEEN 1000 AND 4000 ANNUAL SALARY MORE THAN 15000

SPECIAL OPERATORS :

1. IN
2. NOT IN
3. BETWEEN
4. NOT BETWEEN
5. IS
6. IS NOT
7. LIKE
8. NOT LIKE

1. **<u>IN :</u>** *It is a multi-valued operator which can accept multiple values At the RHS* **.**

Syntax: Column_Name / Exp **IN** (v1 , v2 , . . Vn)

<u>Example :</u>

- WAQTD name and deptno of the employees working in dept 10 or 30 .

  SELECT ENAME , DEPTNO
  FROM EMP
  WHERE **DEPTNO = 10 OR DEPTNO = 30** ;

  SELECT ENAME , DEPTNO
  FROM EMP
  WHERE **DEPTNO IN (10 , 30)** ;

- WAQTD name and job of the employee working as a clerk or manager Or salesman .

  SELECT ENAME , JOB
  FROM EMP
  WHERE JOB IN ('CLERK' , 'MANAGER' , 'SALESMAN') ;

- WAQTD empno , ename and salary of the employees whose empno Is 7902 or 7839 and getting salary more than 2925.

  SELECT EMPNO , ENAME , SAL
  FROM EMP
  WHERE EMPNO IN (7902 , 7839) AND SAL> 2925 ;

2. **<u>NOT IN :</u>** *It is a multi-valued operator which can accept multiple values At the RHS . It is similar to IN op instead of selecting it Rejects the values .*

Syntax: Column_Name / Exp **NOT IN** (v1 , v2 , . . vn)

Example :

- WAQTD name and deptno of all the employees except the emp Working in dept 10 or 40 .

  SELECT ENAME , DEPTNO
  FROM EMP
  WHERE DEPTNO NOT IN (10 , 40) ;

- WAQTD name , deptno and job of the employee working in dept 20 but not as a clerk or manager .

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE DEPTNO = 20 AND
JOB NOT IN ( 'CLERK' ,'MANAGER' ) ;
```

ANSWERS :

1. WAQTD DETAILS OF THE EMPLOYEES WORKING AS CLERK AND EARNING LESS THAN1500

```
SELECT *
FROM EMP
WHERE JOB ='CLERK' AND SAL< 1500 ;
```

2. WAQTD NAME AND HIREDATE OF THE EMPLOYEES WORKING AS MANAGER IN DEPT 30

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE JOB ='MANAGER' AND DEPTNO=30 ;
```

3. WAQTD DETAILS OF THE EMP ALONG WITH ANNUAL SALARY IF THEY ARE WORKING INDEPT 30 AS SALESMAN AND THEIR ANNUAL SALARY HAS TO BE GREATER THAN 14000

```
SELECT EMP.* , SAL*12 ANNUAL_SALARY
FROM EMP
WHERE DEPTNO = 30 AND JOB ='SALESMAN' AND SAL*
12 > 14000 ;
```

4. WAQTD ALL THE DETAILS OF THE EMP WORKING IN DEPT 30 OR AS ANALYST

```
SELECT *
FROM EMP
WHERE DEPTNO = 30 OR JOB ='ANALYST' ;
```

5. WAQTD NAMES OF THE EMPMLOYEES WHOS SALARY IS LESS THAN 1100 AND THEIR DESIGNATION IS CLERK

```
SELECT ENAME
FROM EMP
WHERE SAL< 1100 AND JOB ='CLERK' ;
```

6. WAQTD NAME AND SAL , ANNUAL SAL AND DEPTNO IF DEPTNO IS 20 EARNING MORE THAN 1100 AND ANNUAL SALARY EXCEEDS 12000

```
SELECT ENAME , SAL , SAL*12 , DEPTNO
FROM EMP
WHERE DEPTNO = 20 AND SAL > 1100 AND SAL*12 > 12000 ;
```

7. WAQTD EMPNO AND NAMES OF THE EMPLOYEES WORKING AS MANAGER IN DEPT 20

```
SELECT EMPNO , ENAME
FROM EMP
WHERE DEPTNO = 20 AND JOB ='MANAGER' ;
```

8. WAQTD DETAILS OF EMPLOYEES WORKING IN DEPT 20 OR 30

```
SELECT *
FROM EMP
```

```
WHERE DEPTNO = 10 OR DEPTNO = 30 ;
```

9. WAQTD DETAILS OF EMPLOYEES WORKING AS ANALYST IN DEPT 10

```
SELECT *
FROM EMP
WHERE DEPTNO = 10 AND JOB ='ANALYST' ;
```

10. WAQTD DETAILS OF EMPLOYEE WORKING AS PRESIDENT WITH SALARY OF RUPEES 4000

```
SELECT *
FROM EMP
WHERE SAL=4000 AND JOB ='PRESIDENT' ;
```

11. WAQTD NAMES AND DEPTNO , JOB OF EMPS WORKING AS CLERK IN DEPT 10 OR 20

```
SELECT ENAME, DEPTNO, JOB
FROM EMP
WHERE JOB = ‘CLERK’ AND ( DEPTNO =10 OR DEPTNO =
20);
```

12. WAQTD DETAILS OF EMPLOYEES WORKING AS CLERK OR MANAGER IN DEPT 10

```
SELECT *
FROM EMP
WHERE ( JOB = ‘CLERK’OR  JOB = ‘MANAGER’ ) AND
DEPTNO = 10;
```

13. WAQTD NAMES OF EMPLOYEES WORKING IN DEPT 10 , 20 , 30 , 40

```
SELECT ENAME
FROM EMP
WHERE DEPTNO = 10 OR DEPTNO = 20 OR DEPTNO = 30 OR
DEPTNO =40 ;
```

14. WAQTD DETAILS OF EMPLOYEES WITH EMPNO 7902, 7839

```
SELECT *
FROM EMP
WHERE EMPNO = 7902 OR EMPNO =  7839;
```

15. WAQTD DETAILS OF EMPLOYEES WORKING AS MANAGER OR SALESMAN OR CLERK

```
SELECT *
FROM EMP
WHERE JOB = 'MANAGER' OR JOB = 'SALESMAN' OR JOB =
'CLERK';
```

16. WAQTD NAMES OF EMPLOYEES HIRED AFTER 81 AND BEFORE 87

```
SELECT ENAME
FROM EMP
WHERE HIREDATE > '31-DEC-81' AND HIREDATE <'01-
JAN-87'
```

17. WAQTD DETAILS OF EMPLOYEES EARNING MORE THAN 1250 BUT LESS THAN 3000

```
SELECT *
```

```
FROM EMP
WHERE SAL > 1250 AND SAL < 3000 ;
```

18. WAQTD NAMES OF EMPLOYEES HIRED AFTER 81 INTO DEPT 10 OR 30

```
SELECT ENAME
FROM EMP
WHERE HIREDARE > '31-DEC-81' AND ( DEPTNO = 10 OR
DEPTNO = 20 ) ;
```

19. WAQTD NAMES OF EMPLOYEES ALONG WITH ANNUAL SALARY FOR THE EMPLOYEES WORKING AS MANAGER OR CLERK INTO DEPT 10 OR 30

```
SELECT ENAME , SAL*12
FROM EMP
WHERE ( JOB = 'MANAGER' OR JOB ='CLERK') AND
( DEPTNO = 10 OR DEPTNO = 30 ) ;
```

20. WAQTD ALL THE DETAILS ALONG WITH ANNUAL SALARY IF SAL IS BETWEEN 1000 AND 4000 ANNUAL SALARY MORE THAN 15000

```
SELECT EMP.* , SAL*12
FROM EMP
WHERE SAL > 1000 AND SAL < 4000 AND SAL*12 > 15000 ;
```

3. **BETWEEN :** *"It is used whenever we have range of values "*
[Start value and Stop Value] .

Syntax:

```
Column_Name BETWEEN Lower_Range AND Higher_Range ;
```

- *Between Op works including the range .*

Example :

- WAQTD name and salary of the employees if the emp is earning Salary in the range 1000 to 3000 .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL BETWEEN 1000 AND 3000 ;
```

- WAQTD name and deptno of the employees working in dept 10 And hired during 2019 (the entire year of 2019) .

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE DEPTNO = 10 AND HIREDATE BETWEEN '01-JAN-2019' AND '31-DEC-2019' ;
```

- WAQTD name , sal and hiredate of the employees hired during 2017 into dept 20 with a salary greater that 2000 .

```
SELECT ENAME , SAL , HIREDATE
FROM EMP
WHERE DEPTNO = 20 AND SAL> 2000 AND HIREDATE
BETWEEN '01-JAN2017' AND 31-DEC-2017' ;
```

4. **NOT BETWEEN :** It is Opposite of Between .

Syntax:

```
Column_Name NOT BETWEEN Lower_Range AND Higher_Range ;
```

Example :

- WAQTD name and salary of the employees if the emp is not earning Salary in the range 1000 to 3000 .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL NOT BETWEEN 1000 AND 3000 ;
```

- WAQTD name and deptno of the employees working in dept 10 And not hired during 2019 .

  SELECT ENAME , DEPTNO
  FROM EMP
  WHERE DEPTNO = 10 AND HIREDATE **NOT BETWEEN** '01-JAN-2019' AND '31-DEC-2019' ;

- WAQTD name , sal and hiredate of the employees who were not hired during 2017 into dept 20 with a salary greater that 2000 .

  SELECT ENAME , SAL , HIREDATE
  FROM EMP
  WHERE DEPTNO = 20 AND SAL> 2000 AND HIREDATE **NOT BETWEEN** '01-JAN2017' AND 31-DEC-2017' ;

5. **IS :** *"It is used to compare only NULL "*

Syntax: Column_Name **IS** NULL ;

Example :

EID	**ENAME**	**SAL**	**COMM**
1	A	1000	100
2	B	null	null
3	C	null	200
4	D	2000	null

- WAQTD name of the employee who is not getting salary .

  SELECT ENAME
  FROM EMP
  WHERE SAL **IS** NULL ;

- WAQTD name of the emp who doesn't get commission .

  SELECT ENAME
  FROM EMP
  WHERE COMM **IS** NULL ;

- WAQTD name , sal and comm of the emp if the emp doesn't earn both .

  SELECT ENAME , SAL , COMM
  FROM EMP
  WHERE COMM **IS** NULL AND SAL **IS** NULL ;

6. **IS NOT :** *"It is used to compare the values with NOT NULL ".*

Syntax: Column_Name **IS NOT** NULL ;

Example :

- WAQTD name of the employee who is getting salary .

```
SELECT ENAME
FROM EMP
WHERE SAL IS NOT NULL ;
```

- WAQTD name of the emp who gets commission .

```
SELECT ENAME
FROM EMP
WHERE COMM IS NOT NULL ;
```

- WAQTD name , sal and comm of the emp if the emp doesn't earn commission but gets salary .

```
SELECT ENAME , SAL , COMM
FROM EMP
WHERE COMM IS NULL AND SAL IS NOT NULL ;
```

7. **LIKE :** *"It is used for Pattern Matching ".*

To achieve pattern matching we use special characters .

- Percentile (%)
- Underscore (_)

Syntax: Column_Name LIKE 'pattern' ;

Example :

- WAQTD details of an employee whose name is SMITH .

```
SELECT *
FROM EMP
WHERE ENAME ='SMITH' ;
```

- WAQTD details of the employee who's name starts with 'S' .

```
SELECT *
FROM EMP
WHERE ENAME LIKE 'S%' ;
```

- WAQTD details of the employee who's name ends with 'S' .

```
SELECT *
FROM EMP
WHERE ENAME LIKE '%S' ;
```

- WAQTD names of the employees who have character 'S' in their names .

```
SELECT *
FROM EMP
```

WHERE ENAME LIKE '**%S%**' ;

- WAQTD names that starts with 'J' and ends with 'S' .

  SELECT ENAME
  FROM EMP
  WHERE ENAME LIKE '**J%S**' ;

- WAQTD names of the employee if the emp has char 'A' as his second character .

  SELECT ENAME
  FROM EMP
  WHERE ENAME LIKE '**_A%**' ;

- WAQTD names of the employee if the emp has char 'A' as his Third character .

  SELECT ENAME
  FROM EMP
  WHERE ENAME LIKE '**__A%**' ;

- WAQTD names of the employee if the emp has char 'A' as his second character and 'S' is last character .

  SELECT ENAME
  FROM EMP
  WHERE ENAME LIKE '**_A%S**' ;

- WAQTD names of the employee if the emp has char 'A' present at at least 2 times .

  SELECT ENAME
  FROM EMP
  WHERE ENAME LIKE '**%A%A%**' ;

- WAQTD names of the employee if the emp name starts with 'A' and ends with 'A' .

  SELECT ENAME
  FROM EMP
  WHERE ENAME LIKE '**A%A**' ;

- WAQTD names of the employee if the emp's salary's last 2 digit is 50 rupees .

  SELECT ENAME
  FROM EMP
  WHERE SAL LIKE '**%50**' ;

- WAQTD names of the employees hired in November .

  SELECT ENAME
  FROM EMP

WHERE HIREDATE LIKE '**%NOV%**' **;**

8. **NOT LIKE :**Opposite of Like .

Syntax: Column_Name **NOT LIKE** 'pattern' ;

ASSIGNMENT ON SEPCIAL OPERATORS :

1) LIST ALL THE EMPLOYEES WHOSE COMMISSION IS NULL
2) LIST ALL THE EMPLOYEES WHO DON'T HAVE A REPORTING MANAGER
3) LIST ALL THE SALESMEN IN DEPT 30
4) LIST ALL THE SALESMEN IN DEPT NUMBER 30 AND HAVING SALARY GREATER THAN 1500
5) LIST ALL THE EMPLOYEES WHOSE NAME STARTS WITH 'S' OR 'A'
6) LIST ALL THE EMPLOYEES EXCEPT THOSE WHO ARE WORKING IN DEPT 10 & 20.
7) LIST THE EMPLOYEES WHOSE NAME DOES NOT START WITH 'S'
8) LIST ALL THE EMPLOYEES WHO ARE HAVING REPORTING MANAGERS IN DEPT 10
9) LIST ALL THE EMPLOYEES WHOSE COMMISSION IS NULL AND WORKING AS CLERK
10) LIST ALL THE EMPLOYEES WHO DON'T HAVE A REPORTING MANAGER IN DEPTNO 10 OR 30
11) LIST ALL THE SALESMEN IN DEPT 30 WITH SAL MORE THAN 2450
12) LIST ALL THE ANALYST IN DEPT NUMBER 20 AND HAVING SALARY GREATER THAN 2500
13) LIST ALL THE EMPLOYEES WHOSE NAME STARTS WITH 'M' OR 'J'
14) LIST ALL THE EMPLOYEES WITH ANNUAL SALARY EXCEPT THOSE WHO ARE WORKING IN DEPT 30
15) LIST THE EMPLOYEES WHOSE NAME DOES NOT END WITH 'ES' OR 'R'
16) LIST ALL THE EMPLOYEES WHO ARE HAVING REPORTING MANAGERS IN DEPT 10 ALONG WITH 10% HIKE IN SALARY
17) DISPLAY ALL THE EMPLOYEE WHO ARE 'SALESMAN'S HAVING 'E' AS THE LAST BUT ONE CHARACTER IN ENAME BUT SALARY HAVING EXACTLY 4 CHARACTER
18) DISPLAY ALL THE EMPLOYEE WHO ARE JOINED AFTER YEAR 81
19) DISPLAY ALL THE EMPLOYEE WHO ARE JOINED IN FEB
20) LIST THE EMPLOYEES WHO ARE NOT WORKING AS MANAGERS AND CLERKS IN DEPT 10 AND 20 WITH A SALARY IN THE RANGE OF 1000 TO 3000.

SPECIAL OPERATOR ANSWERS

ROHAN SINGH R

1) LIST ALL THE EMPLOYEES WHOSE COMMISSION IS NULL

SELECT ENAME

FROM EMP WHERE

COMM IS NULL;

2) LIST ALL THE EMPLOYEES WHO DON'T HAVE A REPORTING MANAGER

SELECT ENAME

FROM EMP

WHERE MGR IS NULL;

3) LIST ALL THE SALESMEN IN DEPT 30

SELECT ENAME

FROM EMP

WHERE JOB IN 'SALESMAN' AND DEPTNO IN 30;

4) LIST ALL THE SALESMEN IN DEPT NUMBER 30 AND HAVING SALARY GREATER THAN 1500

SELECT ENAME
FROM EMP
WHERE JOB IN 'SALESMAN' AND DEPTNO IN 30 AND SAL>1500;

5) LIST ALL THE EMPLOYEES WHOSE NAME STARTS WITH 'S' OR 'A'

SELECT ENAME

FROM EMP

WHERE ENAME LIKE 'S%' OR ENAME LIKE 'A%';

6) LIST ALL THE EMPLOYEES EXCEPT THOSE WHO ARE WORKING IN DEPT 10 & 20.

SELECT ENAME
FROM EMP
WHERE DEPTNO NOT IN (10,20);

7) LIST THE EMPLOYEES WHOSE NAME DOES NOT START WITH 'S'

SELECT ENAME

FROM EMP

WHERE ENAME NOT LIKE 'S%';

8) LIST ALL THE EMPLOYEES WHO ARE HAVING REPORTING MANAGERS IN DEPT 10

SELECT ENAME

FROM EMP
WHERE MGR IS NOT NULL AND DEPTNO IN 10;

9) LIST ALL THE EMPLOYEES WHOSE COMMISSION IS NULL AND WORKING AS CLERK

SELECT ENAME

FROM EMP WHERE

COMM IS NULL AND JOB IN 'CLERK';

10) LIST ALL THE EMPLOYEES WHO DON'T HAVE A REPORTING MANAGER IN DEPTNO 10 OR 30

SELECT ENAME
FROM EMP
WHERE MGR IS NULL AND DEPTNO IN (10,30);

11) LIST ALL THE SALESMEN IN DEPT 30 WITH SAL MORE THAN 2450

SELECT ENAME
FROM EMP
WHERE JOB IN 'SALESMAN' AND DEPTNO IN 30 AND SAL>2450;

12) LIST ALL THE ANALYST IN DEPT NUMBER 20 AND HAVING SALARY GREATER THAN 2500

SELECT ENAME
FROM EMP
WHERE JOB IN 'ANALYST' AND DEPTNO IN 30 AND SAL>2500;

13) LIST ALL THE EMPLOYEES WHOSE NAME STARTS WITH 'M' OR 'J'

SELECT ENAME

FROM EMP

WHERE ENAME LIKE 'M%' OR ENAME LIKE 'J%';

14) LIST ALL THE EMPLOYEES WITH ANNUAL SALARY EXCEPT THOSE WHO ARE WORKING IN DEPT 30

*SELECT ENAME,SAL*12 ANNUAL_SAL*

FROM EMP

WHERE DEPTNO NOT IN 30;

15) LIST THE EMPLOYEES WHOSE NAME DOES NOT END WITH 'ES' OR 'R'

SELECT ENAME
FROM EMP
WHERE ENAME NOT LIKE '%ES' AND ENAME NOT LIKE '%R';

16) LIST ALL THE EMPLOYEES WHO ARE HAVING REPORTING MANAGERS IN DEPT 10 ALONG WITH 10% HIKE IN SALARY

*SELECT ENAME, SAL+SAL*10/100*

FROM EMP

WHERE MGR IS NOT NULL AND DEPTNO IN 10;

17) DISPLAY ALL THE EMPLOYEE WHO ARE 'SALESMAN'S HAVING 'E' AS THE LAST BUT ONE CHARACTER IN ENAME BUT SALARY HAVING EXACTLY 4 CHARACTER

SELECT ENAME

FROM EMP

WHERE JOB IN 'SALESMAN' AND ENAME LIKE '%E_' AND SAL LIKE '____';

18) DISPLAY ALL THE EMPLOYEE WHO ARE JOINED AFTER YEAR 81

SELECT ENAME

FROM EMP

WHERE HIREDATE > '31-DEC-81';

19) DISPLAY ALL THE EMPLOYEE WHO ARE JOINED IN FEB

SELECT ENAME
FROM EMP
WHERE HIREDATE LIKE '%FEB%';

20) LIST THE EMPLOYEES WHO ARE NOT WORKING AS MANAGERS AND CLERKS IN DEPT 10 AND 20 WITH A SALARY IN THE RANGE OF 1000 TO 3000

SELECT ENAME
FROM EMP
WHERE JOB NOT IN('MANAGER','CLERK') AND DEPTNO IN(20,10) AND SAL BETWEEN 1000 AND 3000;

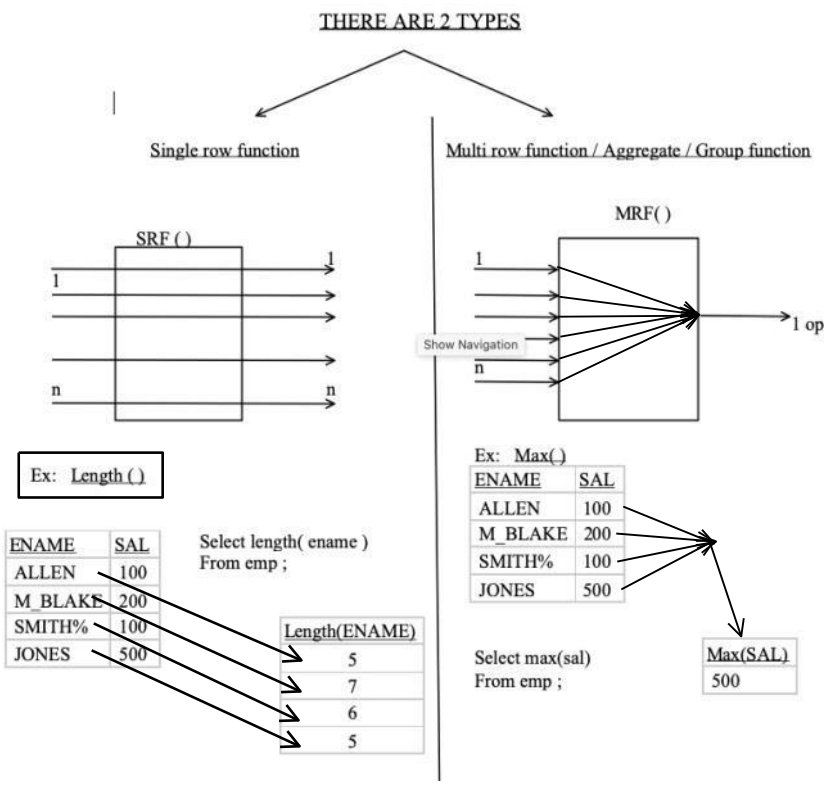
FUNCTIONS

Are a block of code or list of instructions which are used to perform a specific task .

There are 3 main components of a function
1. Function_Name
2. Number_of_arguments (no of inputs)
3. Return type

Types of Functions in SQL :

1. *SINGLE ROW FUNCTIONS*
2. *MUTLI ROW FUNCTIONS* / AGGREGATE / GROUP FUNCTIONS.



Multi Row Functions;

It takes all the inputs at one shot and then executes and provides A single output .

- If we pass 'n' number of inputs to a MRF( ) it returns '1' Output .

List of MRF ( )

1. MAX( ) : it is used to obtain the maximum value present in the column
2. MIN ( ) : it is used to obtain the minimum value present in the

column

3. SUM ( ) : it is used to obtain the summation of values present in the column
4. AVG( ) : it is used to obtain the average of values present in the column
5. COUNT( ) : it is used to obtain the number of values present in the column

NOTE :

- Multi row functions can accept only one argument , i.e a Column_Name or an Expression

  MRF (Column_Name / Exp)

- Along with a MRF( ) we are not supposed to use any other Column_Name in the select clause .

- MRF( ) ignore the Null .

- We cannot use a MRF( ) in where clause .

- COUNT( ) is the only MRF which can accept * as an Argument .

Examples :

1. WAQTD maximum salary given to a manager .

```
SELECT MAX( SAL )
FROM EMP
WHERE JOB ='MANAGER' ;
```

2. WAQTD Total salary given to dept 10

```
SELECT SUM( SAL )
FROM EMP
WHERE DEPTNO =10 ;
```

3. WAQTD number of employees earing more than 1500 in dept 20

```
SELECT COUNT(*)
FROM EMP
WHERE SAL > 1500 AND DEPTNO = 20 ;
```

4. WAQTD number of employee having 'E' in their names .

```
SELECT COUNT(*)
FROM EMP
WHERE ENAME LIKE '%E%' ;
```

5. WAQTD minimum salary given to the employees working as clerk in Dept 10 or 20 .

```
SELECT MIN( SAL )
FROM EMP
WHERE JOB='CLERK' AND DEPTNO IN ( 10 , 20 ) ;
```

6. WAQTD number of employees hired after 1982 and before 1985 into Dept 10 or 30 .

```
SELECT COUNT(*)
FROM EMP
WHERE JIREDATE >'31-DEC-1982' AND HIREDATE <'01-
JAN-1985' AND DEPTNO IN ( 10 , 30 ) ;
```

7. WAQTD number of employees getting commission .

```
SELECT COUNT(*)
FROM EMP
WHERE COMM IS NOT NULL ;


SELECT COUNT( COMM )
FROM EMP ;
```

8. WAQTD maximum salary given to employees if the emp has character 'S' in the name and works as a Manager in dept 10 with as salary of more than 1800 .

```
SELECT MAX( SAL )
FROM EMP
WHERE ENAME LIKE '%S%' AND JOB ='MANAGER' AND
DEPTNO = 10 AND SAL> 1800 ;
```

9. WAQTD number of employees working in dept 10 or 30 and getting commission without the salary .

```
SELECT COUNT(*)
FROM EMP
WHERE DEPTNO IN ( 10 , 30 ) AND COMM IS NOT NULL
AND SAL IS NULL ;


SELECT COUNT( COMM )
FROM EMP
WHERE DEPTNO IN ( 10 , 30 )  AND SAL IS NULL ;
```

10. WAQTD maximum salary given to a manager working in dept 20 and also his comm must be greater than his salary .

```
SELECT MAX( SAL )
FROM EMP
WHERE JOB ='MANAGER' AND DEPTNO = 20 AND COMM >
SAL ;
```

<u>ASSIGNEMENT ON MRF( )</u>

1. WAQTD NUMBER OF EMPLOYEES GETTING SALARY LESS THAN 2000 IN DEPTNO 10
2. WAQTD TOTAL SALARY NEEDED TO PAY EMPLOYEES WORKING AS CLERK
3. WAQTD AVERAGE SALARY NEEDED TO PAY ALL EMPLOYEES
4. WAQTD NUMBER OF EMPLOYEES HAVING 'A' AS THEIR FIRST CHARACTER
5. WAQTD NUMBER OF EMPLOYEES WORKING AS CLERK OR MANAGER
6. WAQTD TOTAL SALARY NEEDED TO PAY EMPLOYEES HIRED IN FEB
7. WAQTD NUMBER OF EMPLOYEES REPORTING TO 7839 (MGR)
8. WAQTD NUMBER OF EMPLOYEES GETTING COMISSION IN DEPTNO 30
9. WAQTD AVG SAL , TOTAL SAL , NUMBER OF EMPS AND MAXIMUM SALARY GIVEN TO EMPLOYEES WORKING AS PERSIDENT
10. WAQTD NUMBER OF EMPLOYEES HAVING 'A' IN THEIR NAMES
11. WAQTD NUMBER OF EMPS AND TOTAL SALARY NEEDED TO PAY THE EMPLOYEES WHO HAVE 2 CONSICUTIVE L's IN THEIR NAMES
12. WAQTD NUMBER OF DEPARTMENTS PRESENT IN EMPLOYEE TABLE
13. WAQTD NUMBER OF EMPLOYEES HAVING CHARACTER 'Z' IN THEIR NAMES
14. WAQTD NUMBER OF EMPLOYEES HAVING '$' IN THEIR NAMES .
15. WAQTD TOTAL SALARY GIVEN TO EMPLOYEES WORKING AS CLERK IN DEPT 30
16. WAQTD MAXIMUM SALARY GIVEN TO THE EMPLOYEES WORKING AS ANALYST
17. WAQTD NUMBER OF DISTINCT SALARIES PRESENT IN EMPLOYEE TABLE
18. WAQTD NUMBER OF JOBS PRESENT IN EMPLOYEE TABLE
19. WATQD AVG SALARY GIVEN TO THE CLERK
20. WAQTD MINIMUM SALARY GIVEN TO THE EMPLOYEES WHO WORK IN DEPT 10 AS MANAGER OR A CLERK

ANSWERS :

1. WAQTD NUMBER OF EMPLOYEES GETTING SALARY LESS THAN 2000 IN DEPTNO 10

```
SELECT COUNT(*)
FROM EMP
WHERE DEPTNO = 10 AND SAL < 2000 ;
```

2. WAQTD TOTAL SALARY NEEDED TO PAY EMPLOYEES WORKING AS CLERK

```
SELECT SUM(SAL)
FROM EMP
WHERE JOB ='CLERK';
```

3. WAQTD AVERAGE SALARY NEEDED TO PAY ALL EMPLOYEES

```
SELECT AVG(SAL)
FROM EMP ;
```

4. WAQTD NUMBER OF EMPLOYEES HAVING 'A' AS THEIR FIRST CHARACTER

```
SELECT COUNT(*)
FROM EMP
WHERE ENAME LIKE 'A%';
```

5. WAQTD NUMBER OF EMPLOYEES WORKING AS CLERK OR MANAGER

```
SELECT COUNT(*)
FROM EMP
WHERE JOB IN ('MANAGER' , 'CLERK' );
```

6. WAQTD TOTAL SALARY NEEDED TO PAY EMPLOYEES HIRED IN FEB

```
SELECT SUM(SAL)
FROM EMP
WHERE HIREDATE LIKE '%FEB%';
```

7. WAQTD NUMBER OF EMPLOYEES REPORTING TO 7839 (MGR)

```
SELECT COUNT(*)
FROM EMP
WHERE MGR = 7839 ;
```

8. WAQTD NUMBER OF EMPLOYEES GETTING COMISSION IN DEPTNO 30

```
SELECT COUNT(*)
FROM EMP
WHERE COMM IS NOT NULL AND DEPTNO = 30 ;
OR
SELECT COUNT(COMM)
FROM EMP
WHERE DEPTNO = 30 ;
```

9. WAQTD AVG SAL , TOTAL SAL , NUMBER OF EMPS AND MAXIMUM SALARY GIVEN TO EMPLOYEES WORKING AS PERSIDENT

```
SELECT AVG(SAL) , SUM(SAL) , COUNT(*) , MAX(SAL)
FROM EMP
WHERE JOB = 'PRESIDENT' ;
```

10. WAQTD NUMBER OF EMPLOYEES HAVING 'A' IN THEIR NAMES

```
SELECT COUNT(*)
FROM EMP
WHERE ENAME LIKE '%A%';
```

11. WAQTD NUMBER OF EMPS AND TOTAL SALary needed to pay THE EMPLOYEES WHO HAVE 2 CONSICUTIVE L's IN THEIR NAMES

```
SELECT COUNT(*) , SUM(SAL)
FROM EMP
WHERE ENAME LIKE '%LL%';
```

12. WAQTD NUMBER OF DEPARTMENTS PRESENT IN EMPLOYEE TABLE

```
SELECT COUNT( DISTINCT DEPTNO )
FROM EMP ;
```

13. WAQTD NUMBER OF EMPLOYEES HAVING CHARACTER '_'

IN THEIR NAMES

```
SELECT COUNT(*)
FROM EMP
WHERE ENAME LIKE '%!_%' ESCAPE '!' ;
```

14. WAQTD NUMBER OF EMPLOYEES HAVING ATLEAST 2 PERCENTILES IN THEIR NAMES

```
SELECT COUNT(*)
FROM EMP
WHERE ENAME LIKE '%!%%!%%' ESCAPE '%' ;
```

15. WAQTD TOTAL SALARY GIVEN TO EMPLOYEES WORKING AS CLERK IN DEPT 30

```
SELECT SUM(SAL)
FROM EMP
WHERE JOB ='CLERK' AND DEPTNO = 30 ;
```

16. WAQTD MAXIMUM SALARY GIVEN TO THE EMPLOYEES WORKING AS ANALYST

```
SELECT MAX(Sal)
FROM EMP
WHERE JOB ='ANALYST' ;
```

17. WAQTD NUMBER OF DISTINCT SALARIES PRESENT IN EMPLOYEE TABLE

```
SELECT COUNT( DISTINCT SAL )
FROM EMP ;
```

18. WAQTD NUMBER OF JOBS PRESENT IN EMPLOYEE TABLE

```
SELECT COUNT( DISTINCT JOB )
FROM EMP ;
```

19. WATQD AVG SALARY GIVEN TO THE CLERK

```
SELECT AVG(SAL)
FROM EMP
WHERE JOB ='CLERK' ;
```

20. WAQTD MINIMUM SALARY GIVEN TO THE EMPLOYEES WHO WORK IN DEPT 10 AS MANAGER OR A CLERK

```
SELECT MIN(SAL)
FROM EMP
WHERE DEPTNO = 10 AND JOB IN ( 'MANAGER' , 'CLERK' ) ;
```

.