DAY 1

Thursday, July 16, 2020 9:35 AM

SQL (STRUCTURED QUERY LANGUAGE)

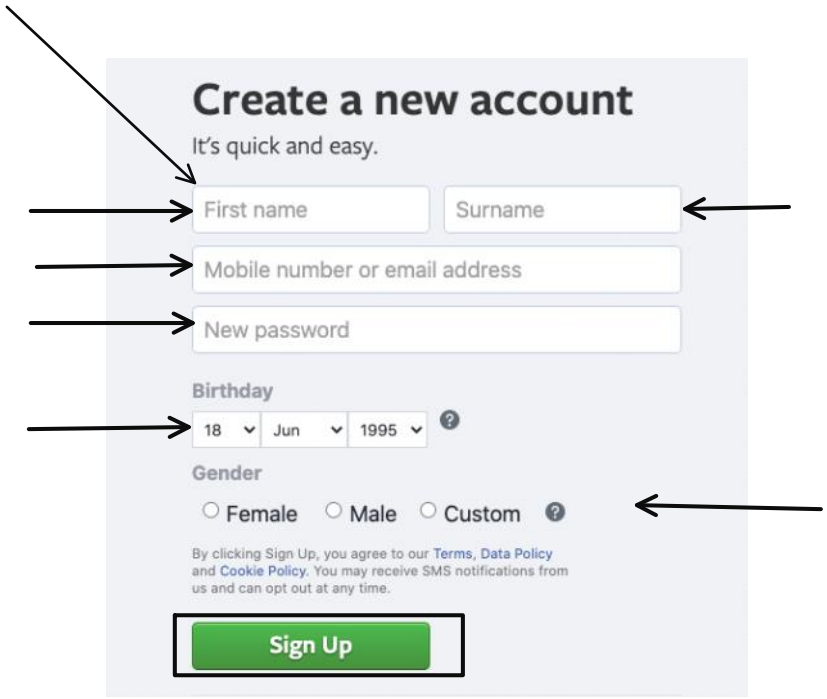
↓

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

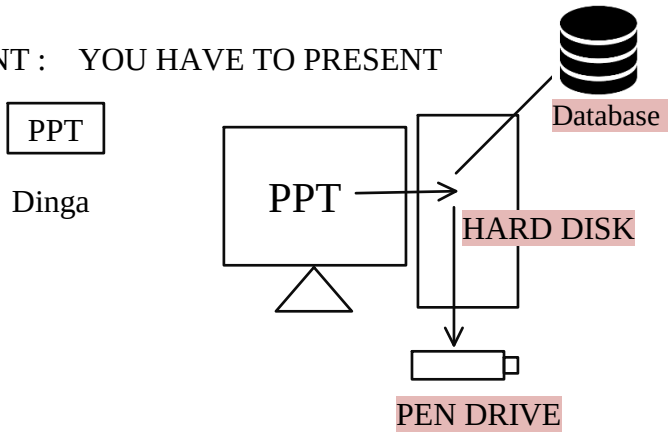
Contacts

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



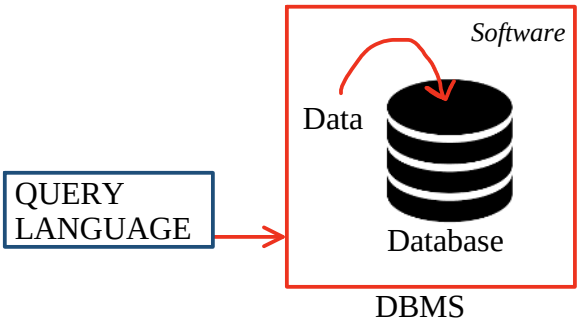
- The basic operations that can be performed on a database are
  - CREATE / INSERT
  - READ / RETRIEVE
  - UPDATE / MODIFY
  - DELETE / DROP

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

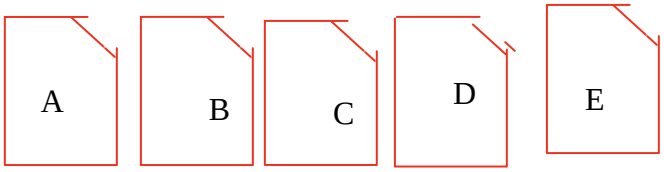
DATABASE MANAGEMENT SYSTEM (DBMS) :

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

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



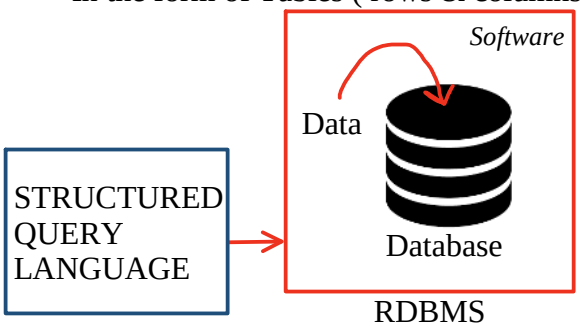
- We use query language to communicate or interact with DBMS
- DBMS stores the data in the form of *files* .



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

DAY 2

Friday, 17 July 2020 8:59 AM

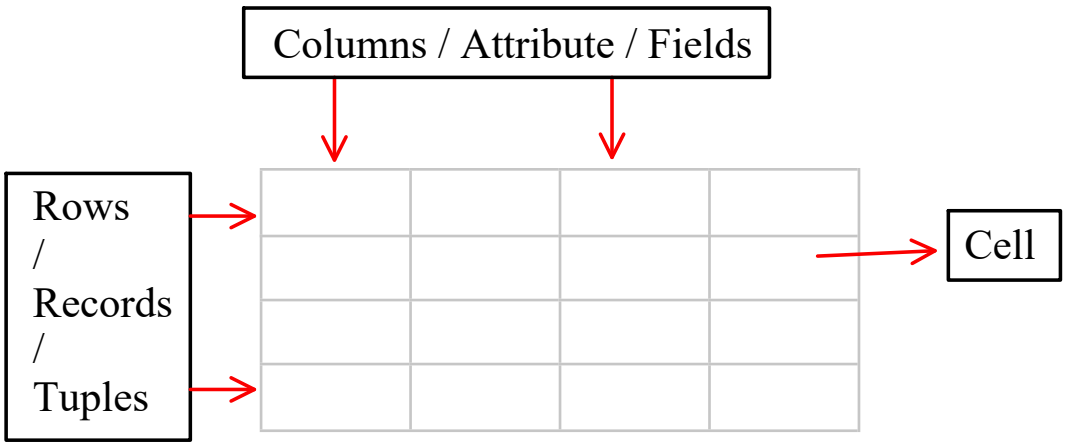
RELATIONAL MODEL :

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 .

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 :

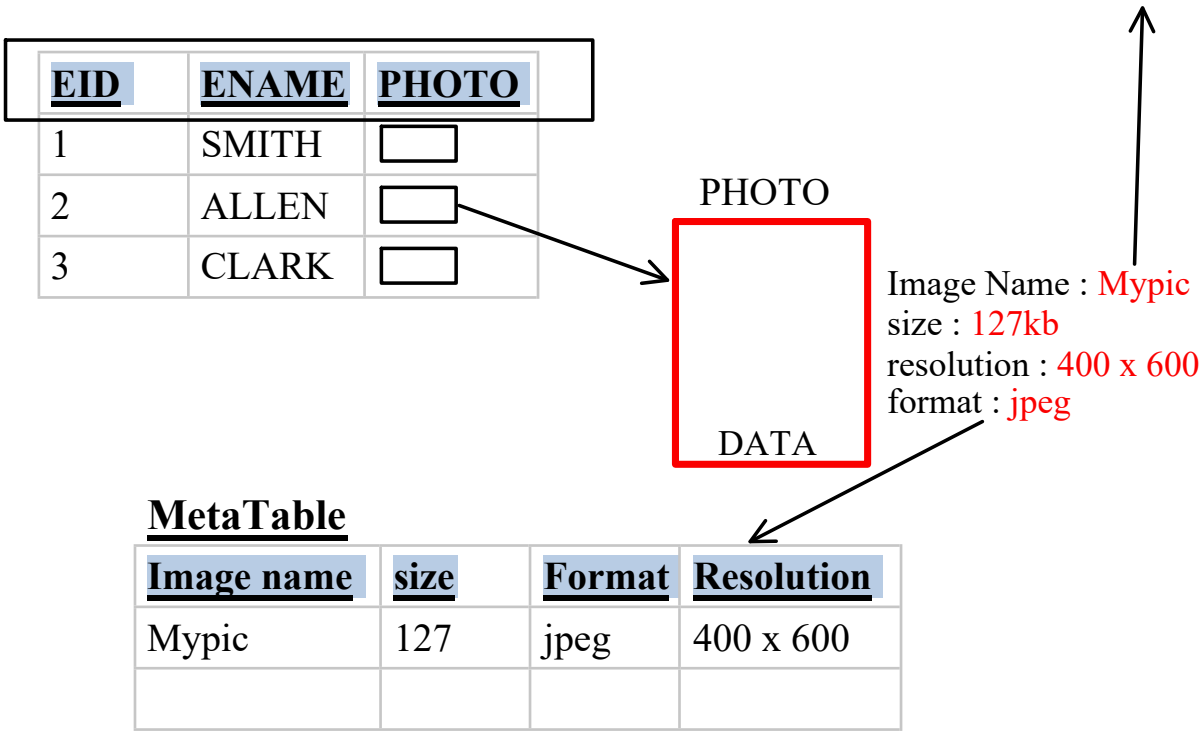
EID	ENAME	PHONE_NO
1	SMITH	101
2	ALLEN	102 , 202
3	CLARK	103

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 .

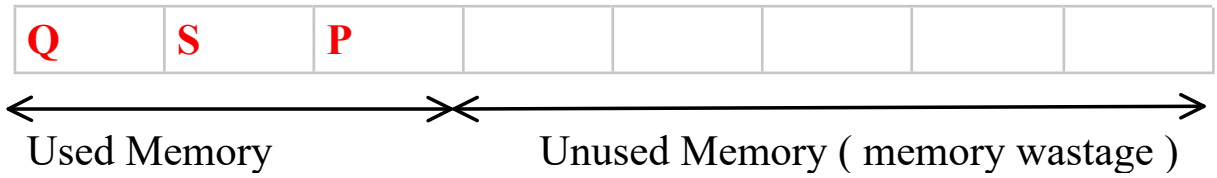
NOTE : SQL is not a Case Sensitive Language .

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)

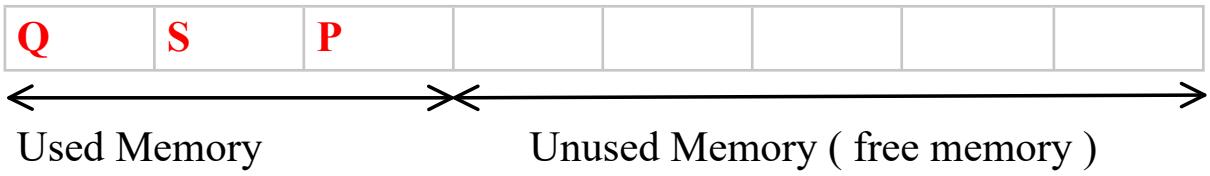


2. **VARCHAR :** In varchar datatype we can store 'A-Z' , 'a-z' , '0-9' And Special Characters($, & , @ , ! …) .

- 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)



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)
QSP1	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

DAY 3

Monday, July 20, 2020 9:40 AM

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 .

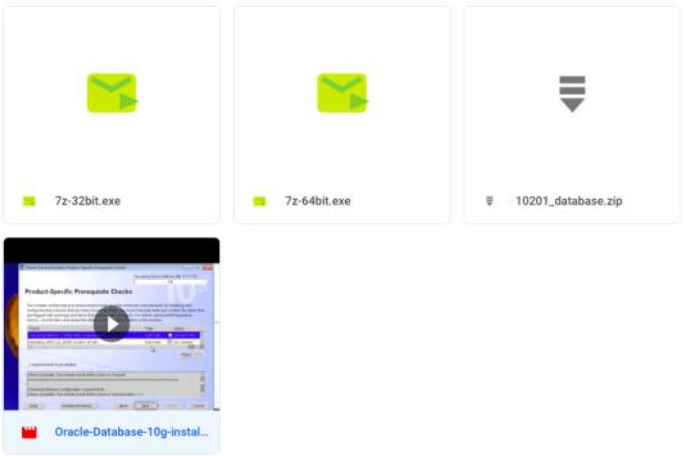
2. Binary large object (BLOB) :

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



DAY 4

Tuesday, July 21, 2020 9:34 AM

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 .
- We cannot Equate Null .

DAY 5

Wednesday, July 22, 2020 9:02 AM

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)

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

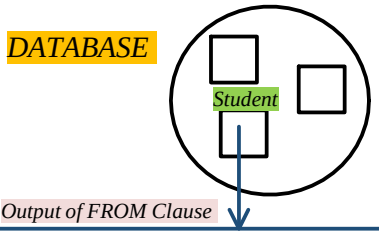
- "It is a process of retrieving the data by *selecting only the columns* is known as Projection " .
- In projection all the records / values present in a particular column are by default selected .

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 .

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



SELECT SNAME
FROM STUDENT ;

Output of FROM Clause

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.

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

DAY 6

Thursday, July 23, 2020 9:00 AM

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

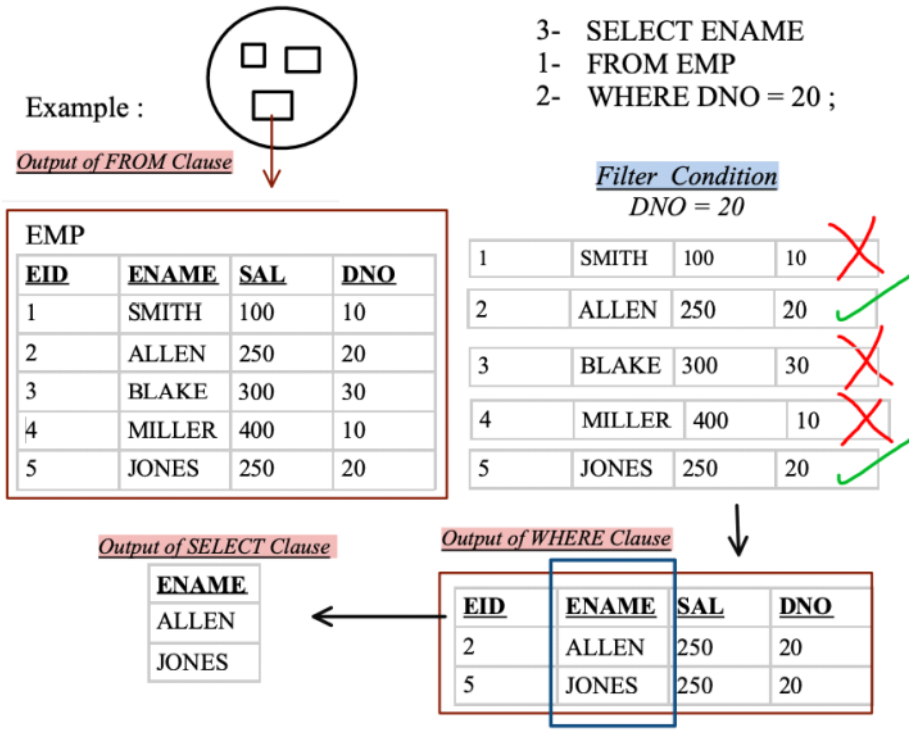
1. FROM
2. WHERE
3. SELECT

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

DAY 7

Friday, July 24, 2020 9:44 AM

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 ;
```

ASSIGNMENT ON LOGICAL OPERATORS :

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. **IN :** *It is a multi-valued operator which can accept multiple values At the RHS* **.**

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

Example :

- 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. **NOT IN :** *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 ;
```

DAY 8

Monday, July 27, 2020 9:32 AM

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;
```

DAY 9

Tuesday, July 28, 2020 9:37 AM

FUNCTIONS

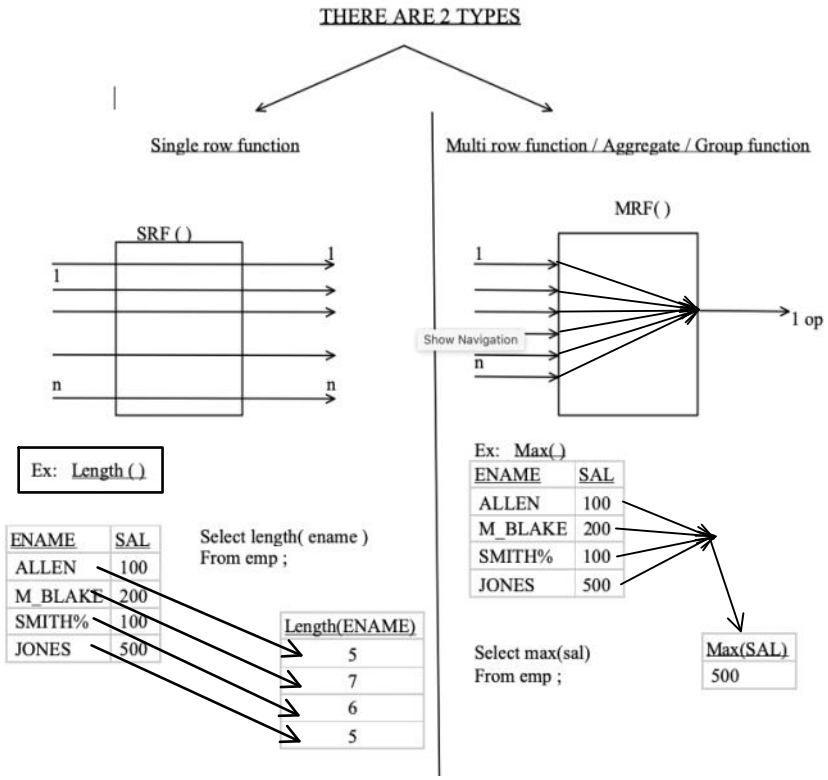
Are a block of code or list of instructions which are used to perform a specific task .

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 ;
```

ASSIGNEMENT ON MRF( )

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' ) ;
```

.

DAY 10

Wednesday, July 29, 2020 9:44 AM

GROUP & FILTERING

GROUPING : GROUP BY Clause

Group by clause is used to *group the records* .

SYNTAX:

```
SELECT  group_by_expression / group_function
FROM table_name
[WHERE <filter_condition>]
GROUP BY column_name/expression ;
```

ORDER OF EXECUTION:

1-FROM
2-WHERE(if used) [ROW-BY-ROW]
3-GROUP BY [ROW-BY-ROW]
4-SELECT [GROUP-BY-GROUP]

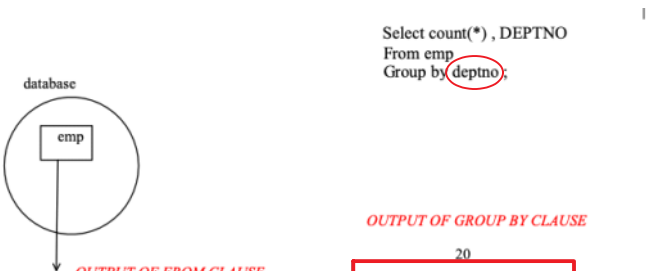
EMP

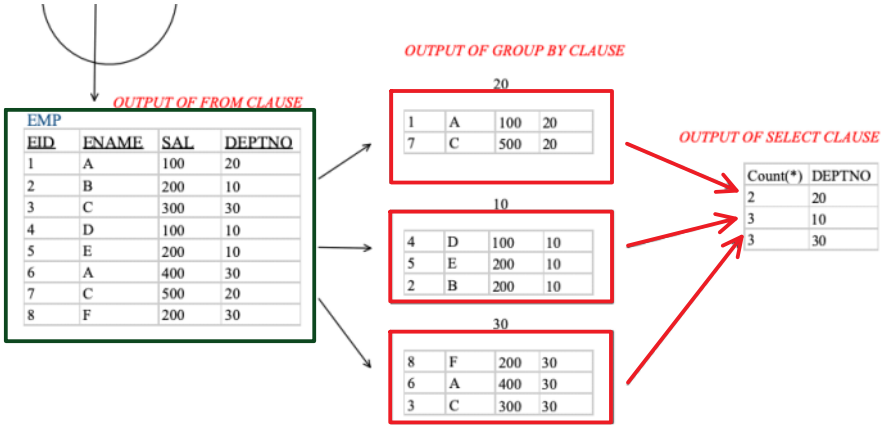
EID	ENAME	SAL	DEPTNO
1	A	100	20
2	B	200	10
3	C	300	30
4	D	100	10
5	E	200	10
6	A	400	30
7	C	500	20
8	F	200	30

Example :

- WAQTD number of employees working in each dept .

```
SELECT  COUNT(*)
FROM EMP
GROUP BY DEPTNO ;
```





NOTE :

- Group By clause is used to group the records .
- Group By clause executes row by row .
- After the execution of Group By clause we get Groups .
- Therefore any clause that executes after group by must execute Group By Group .
- The Column_Name or expression used for grouping can be used In select clause .
- Group By clause can be used without using Where clause .

Questions :

1. WAQTD number of employees working in each dept except the Employee working as analyst .

```
SELECT DEPTNO , COUNT(*)
FROM EMP
WHERE JOB NOT IN 'ANALYST'
GROUP BY DEPTNO ;
```

2. WAQTD maximum salary given to each job .

```
SELECT JOB , MAX( SAL )
FROM EMP
GROUP BY JOB ;
```

3. WAQTD number of employees working in each job if the employees Have character 'A' in their names .

```
SELECT JOB , COUNT(*)
FROM EMP
WHERE ENAME LIKE '%A%'
GROUP BY JOB ;
```

4. WAQTD number of employees getting commission in each dept .

```
SELECT DEPTNO , COUNT( COMM )
```

```
FROM EMP
GROUP BY DEPTNO ;
```

ASSIGNMENT QUESTIONS ON GROUP BY

1.WAQTD NUMBER OF EMPLOYEES WORKING IN EACH DEPARTEMENT EXCEPT PRESIDENT.
2.WAQTD TOTAL SALARY NEEDED TO PAY ALL THE EMPLOYEES IN EACH JOB.
3.WAQTD NUMBER OF EMPLOYEEES WORKING AS MANAGER IN EACH DEPARTMENT .
4.WAQTD AVG SALARY NEEDED TO PAY ALL THE EMPLOYEES IN EACH DEPARTMENT EXCLUDING THE EMPLOYEES OF DEPTNO 20.
5.WAQTD NUMBER OF EMPLOYEES HAVING CHARACTER 'A' IN THEIR NAMES IN EACH JOB .
6.WAQTD NUMBER OF EMPLOYEES AND AVG SALARY NEEDED TO PAY THE EMPLOYEES WHO SALARY IN GREATER THAN 2000 IN EACH DEPT.
7.WAQDTD TOTAL SALARY NEEDED TO PAY AND NUMBER OF SALESMANS IN EACH DEPT.
8.WAQTD NUMBER OF EMPLOYEES WITH THEIR MAXIMUM SALARIES IN EACH JOB.
9.WAQTD MAXIMUM SALARIES GIVEN TO AN EMPLOYEE WORKING IN EACH DEPT.
10.WAQTD NUMBER OF TIMES THE SALARIES PRESENT IN EMPLOYEE TABLE .

FILTERING : HAVING Clause

" Having Clause is used to Filter the Group "

SYNTAX:

```
SELECT  group_by_expression / group_function
FROM table_name
[WHERE <filter_condition>]
GROUP BY column_name/expression
HAVING  <group_filter_condition>
```

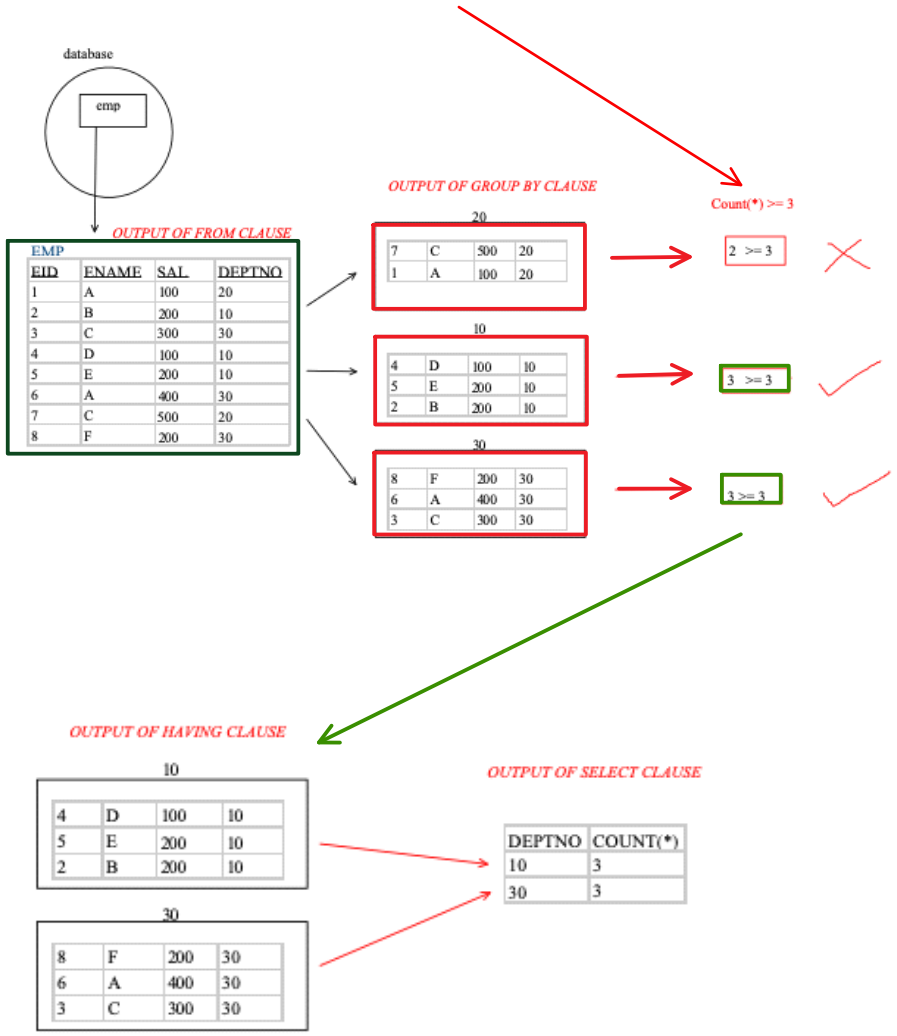
ORDER OF EXECUTION:

1-FROM
2-WHERE(if used) *[ROW-BY-ROW]*
3-GROUP BY(if used) *[ROW-BY-ROW]*
4-HAVING (if used) *[GROUP-BY-GROUP]*
5-SELECT *[GROUP-BY-GROUP]*

Example :

- WAQTD to find number of employees working in each Dept if there are at least 3 employees in each dept .

```
SELECT DEPTNO , COUNT(*)
FROM EMP
GROUP BY DEPTNO
HAVING COUNT(*)>=3 ;
```



Questions :

1. WAQTD the designations in which there are at lest 2 employees Present .

   ```
   SELECT JOB , COUNT(*)
   FROM EMP
   GROUP BY JOB
   HAVING COUNT(*)>=2 ;
   ```

2. WAQTD the names that are repeated .

   ```
   SELECT ENAME , COUNT(*)
   FROM EMP
   GROUP BY ENAME
   HAVING COUNT(*) > 1 ;
   ```

3. WAQTD names that are repeated exactly twice .

```
SELECT ENAME , COUNT(*)
FROM EMP
GROUP BY ENAME
HAVING COUNT(*)  = 2  ;
```

4. WAQTD the salary that is repeated .

```
SELECT SAL, COUNT(*)
FROM EMP
GROUP BY SAL
HAVING COUNT(*) > 1 ;
```

5. WAQTD number of employees working in each dept having At least 2 emp's Character 'A' or 'S' in their names .

```
SELECT DEPTNO , COUNT(*)
FROM EMP
WHERE ENAME LIKE '%A%' OR ENAME LIKE '%S%'
GROUP BY DEPTNO
HAVING COUNT(*)>=2 ;
```

6. WAQTD job and total salary of each job , if the total salary Of each job is greater than 3450 .

```
SELECT JOB , SUM( SAL )
FROM EMP
GROUP BY JOB
HAVING SUM( SAL ) > 3450 ;
```

7. WAQTD job and total salary of the employees if the employees Are earning more than 1500.

```
SELECT JOB , SUM( SAL )
FROM EMP
WHERE SAL > 1500
GROUP BY JOB ;
```

NOTE :

Differentiate between Where and Having .

WHERE	**HAVING**
➢ Where clause is used to Filter the records	➢ Having clause is used to Filter the groups .
➢ Where clause executes row By row .	➢ Having clause executes Group by group
➢ In Where Clause we cannot Use MRF( )	➢ Can use MRF( ).
➢ Where clause executes before Group by clause .	➢ Having clause executes After group by clause .

8. WAQTD Job wise maximum salary if the maximum salary Of each job exceeds 2000 .

```
SELECT JOB , MAX( SAL )
FROM EMP
GROUP BY JOB
HAVING MAX( SAL ) > 2000 ;
```

9. WAQTD number of emp earning sal more than 1200 in each job and the total sal needed to pay emp of each job must exceeds 3800.

```
SELECT JOB , COUNT(*) , SUM( SAL )
FROM EMP WHERE SAL > 1200
GROUP BY JOB
HAVING SUM( SAL ) > 3800 ;
```

ASSIGNMENT QUESTIONS ON HAVING CLAUSE

1.WAQTD DNO AND NUMBER OF EMP WORKING IN EACH DEPT IF THERE
ARE ATLEAST 2 CLERKS IN EACH DEPT
2.WAQTD DNO AND TOTAL SAALARYNEEDED TO PAY ALL EMP
IN EACH DEPT IF THERE ARE ATLEAST 4 EMP IN EACH DEPT
3.WAQTD NUMBER OF EMP EARNING SAL MORE THAN 1200 IN EACH JOB
AND THE TOTAL SAL NEEDED TO PAY EMP OF EACH JOB MUST EXCEES 3800
4.WAQTD DEPTNO AND NUMBER OF EMP WORKING ONLY IF THERE ARE 2 EMP WORKING IN EACH DEPT AS MANAGER .
5.WAQTD JOB AND MAX SAL OF EMP IN EACH JOB IF THE MAX SAL EXCEEDS 2600
6.WAQTD THE SALARIES WHICH ARE REPEATED IN EMP TABLE
7.WAQTD THE HIREDATE WHICH ARE DUPLICATED IN EMP TABLE
8.WAQTD AVG SALARY OF EACH DEPT IF AVG SAL IS LESS THAN 3000
9.WAQTD DEPTNO IF THERE ARE ATLEAST 3 EMP IN EACH DEPT WHOS NAME
HAS CHAR 'A' OR 'S' .
10.WAQTD MIN AND MAX SALARIES OF EACH JOB IF MIN SAL IS MORE THAN 1000 AND MAX SAL IS LESS THAN 5000 .

ANSWERS :

1.WAQTD NUMBER OF EMPLOYEES WORKING IN EACH DEPARTEMENT EXCEPT PRESIDENT

```
SELECT DEPTNO, COUNT(*)
FROM EMP
WHERE JOB NOT IN 'PRESIDENT'
GROUP BY DEPTNO;
```

2.WAQTD TOTAL SALARY NEEDED TO PAY ALL THE EMPLOYEES IN EACH JOB

```
SELECT JOB , SUM(SAL)
FROM EMP
GROUP BY JOB
```

3.WAQTD NUMBER OF EMPLOYEEES WORKING AS MANAGER IN EACH DEPARTMENT

```
SELECT DEPTNO, COUNT(*)
FROM EMP
WHERE JOB='MANAGER'
GROUP BY DEPTNO;
```

4.WAQTD AVG SALARY NEEDED TO PAY ALL THE EMPLOYEES IN EACH DEPARTMENT EXCLUDING THE EMPLOYEES OF DEPTNO 20

```
SELECT DEPTNO, AVG(SAL)
FROM EMP
WHERE DEPTNO NOT IN 20
GROUP BY DEPTNO;
```

5.WAQTD NUMBER OF EMPLOYEES HAVING CHARACTER 'A' IN THEIR NAMES IN EACH JOB

```
SELECT JOB, COUNT(*)
FROM EMP
WHERE ENAME LIKE '%A%'
GROUP BY JOB;
```

6.WAQTD NUMBER OF EMPLOYEES AND AVG SALARY NEEDED TO PAY THE EMPLOYEES WHO SALARY IN GREATER THAN 2000 IN EACH DEPT

```
SELECT DEPTNO, COUNT(*) , AVG(SAL)
FROM EMP
WHERE SAL > 2000
GROUP BY DEPTNO;
```

7.WAQDTD TOTAL SALARY NEEDED TO PAY AND NUMBER OF SALESMANS IN EACH DEPT

```
SELECT DEPTNO, COUNT(*) , SUM(SAL)
FROM EMP
WHERE JOB='SALESMAN'
GROUP BY DEPTNO;
```

8.WAQTD NUMBER OF EMPLOYEES WITH THEIR MAXIMUM

SALARIES IN EACH JOB

```
SELECT JOB, COUNT(*) , MAX(SAL)
FROM EMP
GROUP BY JOB;
```

9.WAQTD MAXIMUM SALARIES GIVEN TO AN EMPLOYEE WORKING IN EACH DEPT

```
SELECT DEPTNO, MAX(SAL)
FROM EMP
GROUP BY DEPTNO;
```

10.WAQTD NUMBER OF TIMES THE SALARIES PRESENT IN EMPLOYEE TABLE

```
SELECT SAL , COUNT(*)
FROM EMP
GROUP BY SAL;
```

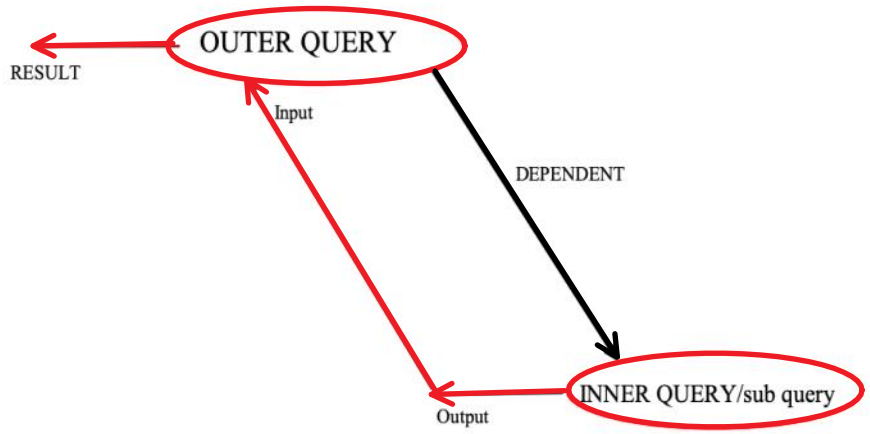
DAY 11

Thursday, July 30, 2020 9:40 AM

SUB-QUERY

" A QUERY WRITTEN INSIDE ANOTHER QUERY IS KNOWN AS SUB QUERY "

Working Principle :



Let us consider two queries Outer Query and Inner Query .

- Inner Query executes first and produces an Output .
- The Output of Inner Query is given / fed as an Input to Outer Query .
- The Outer Query generates the Result.
- Therefore we can state that 'the Outer Query is dependent on Inner Query' and this is the Execution Principle of Sub Query .

Why / When Do we use SUB QUERY :

Case 1 : Whenever we have Unknowns present in the Question We use sub query to find the Unknown .

Example :

EMP

EID	ENAME	SAL	DEPTNO
1	ALLEN	1000	20
2	BLAKE	2000	10
3	CLARK	3000	30
4	MILLER	1500	10
5	SMITH	2500	10

1. WAQTD names of the employees earning more than 2500 .

   SELECT ENAME
   FROM EMP

```
WHERE SAL > 2500 ;
```

2. WAQTD names of the employees earning less than MILLER .

```
SELECT ENAME
FROM EMP
WHERE SAL < ( SELECT SAL
              FROM EMP
              WHERE ENAME = 'MILLER'  );
```

3. WAQTD name and deptno of the employees working in the same Dept as SMITH .

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
                 FROM EMP
                 WHERE ENAME ='SMITH' ) ;
```

4. WAQTD name and hiredate of the employees if the employee Was hired after JONES .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE > ( SELECT HIREDATE
                   FROM EMP
                   WHERE ENAME ='JONES' ) ;
```

5. WAQTD all the details of the employee working in the same Designation as KING .

```
SELECT *
FROM EMP
WHERE JOB = ( SELECT JOB
              FROM EMP
              WHERE ENAME ='KING' );
```

6. WAQTD name , sal , deptno of the employees if the employees Earn more than 2000 and work in the same dept as JAMES .

```
SELECT ENAME , SAL , DEPTNO
FROM EMP
WHERE SAL > 2000 AND DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME ='JAMES' ) ;
```

7. WAQTD all the details of the employees working in the Same designation as MILLER and earning more than 1500.

```
SELECT *
FROM EMP
WHERE SAL > 1500 AND JOB = ( SELECT JOB
                             FROM EMP
                             WHERE ENAME ='MILLER' ) ;
```

```
SELECT *
FROM EMP
WHERE  JOB = ( SELECT JOB
FROM EMP
WHERE ENAME ='MILLER' )  AND SAL > 1500  ;
```

8. WAQTD details of the employees earning more than SMITH
   But less than KING .

```
SELECT *
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='SMITH' ) AND SAL < ( SELECT SAL
FROM EMP
WHERE ENAME ='KING' ) ;
```

9. WAQTD name , sal and deptno of the employees if the employee Is
   earning commission in dept 20 and earning salary more than Scott .

```
SELECT ENAME , SAL , DEPTNO
FROM EMP
WHERE COMM IS NOT NULL AND DEPTNO = 20 AND
SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='SCOTT' ) ;
```

10. WAQTD name and hiredate of the employees who's name ends with
    'S' and hired after James .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE ENAME LIKE '%S' AND
HIREDATE > ( SELECT HIREDATE
FROM EMP
WHERE ENAME ='JAMES' ) ;
```

11. WAQTD names of the employees working in the same dept as
    JAMES and earning salary more than ADAMS and working in the
    same job role as MILLER and hired after MARTIN .

```
SELECT ENAME
FROM EMP
WHERE DEPTNO=(SELECT DEPTNO
 FROM EMP
WHERE ENAME='JAMES') AND
SAL>(SELECT SAL
FROM EMP
WHERE ENAME='ADAMS') AND
JOB=(SELECT JOB
FROM EMP
WHERE ENAME='MILLER') AND
```

```
HIREDATE>(SELECT HIREDATE
FROM EMP
WHERE ENAME='MARTIN');
```

12. WAQTD all the details of the employees working as salesman in the dept 20 and earning commission more than Smith and hired after KING .

```
SELECT *
FROM EMP
WHERE JOB ='SALESMAN' AND
DEPTNO = 20 AND
COMM > ( SELECT COMM
FROM EMP
WHERE ENAME ='SMITH' ) AND
HIREDATE > ( SELECT HIREDATE
FROM EMP
WHERE ENAME ='KING' ) ;
```

13. WAQTD number of employees earning more than SMITH and less than MARTIN .

```
SELECT COUNT(*)
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='SMITH') AND
SAL < ( SELECT SAL
FROM EMP
WHERE ENAME ='MARTIN' )
```

14. WAQTD Ename and SAL for all the employees earning more than JONES .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME =JONES' ) ;
```

15. WAQTD all the details of the employees working as a manager .

```
SELECT *
FROM EMP
WHERE JOB ='MANAGER' ;
```

NOTE :

- In the Inner Query / Sub Query we cannot select more than One column .
- The corresponding columns need not be same , but the datatypes of those has to be same .

ASSIGNMENT ON CASE 1

1.WAQTD NAME OF THE EMPLOYEES EARNING MORE THAN ADAMS
2.WAQTD NAME AND SALARY OF THE EMPLOYEES EARNING LESS THAN KING
3.WAQTD NAME AND DEPTNO OF THE EMPLOYEES IF THEY ARE WORKING IN THE SAME DEPT AS JONES
4.WAQTD NAME AND JOB OF ALL THE EMPLOYEES WORKING IN THE SAME DESIGNATION AS JAMES
5.WAQTD EMPNO AND ENAME ALONG WITH ANNUAL SALARY OF ALL THEEMPLOYEES IF THEIR ANNUAL SALARY IS GREATER THAN WARDS ANNUAL SALARY.
6.WAQTD NAME AND HIREDATE OF THE EMPLOYEES IF THEY ARE HIRED BEFORE SCOTT
7.WAQTD NAME AND HIREDATE OF THE EMPLOYEES IF THEY ARE HIRED AFTER THE PRESIDENT
8.WAQTD NAME AND SAL OF THE EMPLOYEE IF THEY ARE EARNING SAL LESS THAN THE EMPLOYEE WHOS EMPNO IS 7839
9.WAQTD ALL THE DETAILS OF THE EMPLOYEES IF THE EMPLOYEES ARE HIRED BEFORE MILLER
10.WAQTD ENAME AND EMPNO OF THE EMPLOYEES IF EMPLOYEES ARE EARNING MORE THAN ALLEN
11.WAQTD ENAME AND SALARY OF ALL THE EMPLOYEES WHO ARE EARNING MORE THAN MILLER BUT LESS THAN ALLEN .
12.WAQTD ALL THE DETAILS OF THE EMPLOYEES WORKING IN DEPT 20 AND WORKING IN THE SAME DESIGNATION AS SMITH
13.WAQTD ALL THE DETAILS OF THE EMPLOYEES WORKING AS MANAGER IN THE SAME DEPT AS TURNER
14.WAQTD NAME AND HIREDATE OF THE EMPLOYEES HIRED AFTER 1980 AND BEFORE KING
15.WAQTD NAME AND SAL ALONG WITH ANNUAL SAL FOR ALL EMPLOYEES WHOS SAL IS LESS THAN BLAKE AND MORE THAN 3500
16.WAQTD ALL THE DETAILS OF EMPLOYEES WHO EARN MORE THAN SCOTT BUT LESS THAN KING
17.WAQTD NAME OF THE EMPLOYEES WHOS NAME STARTS WITH 'A' AND WORKS IN THE SAME DEPT AS BLAKE
18.WAQTD NAME AND COMM IF EMPLOYEES EARN COMISSION AND WORK IN THE SAME DESIGNATION AS SMITH
19.WAQTD DETAILS OF ALL THE EMPLOYEES WORKING AS CLERK IN THE SAME DEPT AS TURNER .
20.WAQTD ENAME, SAL AND DESIGNATION OF THE EMPLOYEES WHOS ANNUAL SALARY IS MORE THAN SMITH AND LESS THAN KING.

1.WAQTD NAME OF THE EMPLOYEES EARNING MORE THAN ADAMS

```
SELECT ENAME
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='ADAMS' );
```

2.WAQTD NAME AND SALARY OF THE EMPLOYEES EARNING LESS
THAN KING

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL < ( SELECT SAL
FROM EMP
WHERE ENAME ='KING' );
```

3.WAQTD NAME AND DEPTNO OF THE EMPLOYEES IF THEY ARE WORKING
IN THE SAME DEPT AS JONES

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME =JONES');
```

4.WAQTD NAME AND JOB OF ALL THE EMPLOYEES WORKING IN THE SAME
DESIGNATION AS JAMES

```
SELECT ENAME , JOB
FROM EMP
WHERE JOB = ( SELECT JOB
FROM EMP
WHERE ENAME ='JAMES' );
```

5.WAQTD EMPNO AND ENAME ALONG WITH ANNUAL SALARY OF ALL THE
EMPLOYEES IF THEIR ANNUAL SALARY IS GREATER THAN WARDS
ANNUAL SALARY.

```
SELECT EMPNO , ENAME . SAL*12
FROM EMP
WHERE SAL * 12 > ( SELECT SAL*12
FROM EMP
WHERE ENAME = 'WARD' );
```

6.WAQTD NAME AND HIREDATE OF THE EMPLOYEES IF THEY ARE HIRED
BEFORE SCOTT

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE < ( SELECT HIREDATE
FROM EMP
WHERE ENAME ='SCOTT' );
```

7.WAQTD NAME AND HIREDATE OF THE EMPLOYEES IF THEY ARE HIRED
AFTER THE PRESIDENT

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE > ( SELECT HIREDATE
FROM EMP
```

WHERE JOB = 'PRESIDENT');

8.WAQTD NAME AND SAL OF THE EMPLOYEE IF THEY ARE EARNING SAL
LESS THAN THE EMPLOYEE WHOS EMPNO IS 7839

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL < ( SELECT SAL
FROM EMP
WHERE EMPNO = 7839 );
```

9.WAQTD ALL THE DETAILS OF THE EMPLOYEES IF THE EMPLOYEES ARE
HIRED BEFORE MILLER

```
SELECT *
FROM EMP
WHERE HIREDATE < ( SELECT HIREDATE
FROM EMP
WHERE ENAME ='MILLER' );
```

10.WAQTD ENAME AND EMPNO OF THE EMPLOYEES IF EMPLOYEES ARE
EARNING MORE THAN ALLEN

```
SELECT ENAME , EMPNO
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='ALLEN' );
```

11.WAQTD ENAME AND SALARY OF ALL THE EMPLOYEES WHO ARE EARNING
MORE THAN MILLER BUT LESS THAN ALLEN

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='MILLER ) AND  SAL < ( SELECT SAL
FROM EMP
WHERE ENAME ='ALLEN' );
```

12.WAQTD ALL THE DETAILS OF THE EMPLOYEES WORKING IN DEPT 20
AND WORKING IN THE SAME DESIGNATION AS SMITH

```
SELECT *
FROM EMP
WHERE DEPTNO = 20 AND JOB = ( SELECT JOB
FROM EMP
WHERE ENAME ='SMITH');
```

13.WAQTD ALL THE DETAILS OF THE EMPLOYEES WORKING AS MANAGER
IN THE SAME DEPT AS TURNER

```
SELECT *
FROM EMP
```

WHERE JOB ='MANAGER' AND DEPTNO = (SELECT DEPTNO
FROM EMP
WHERE ENAME ='TURNER') ;

14.WAQTD NAME AND HIREDATE OF THE EMPLOYEES HIRED AFTER 1980
AND BEFORE KING

SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE > '31-DEC-1980 ' AND HIREDATE < (SELECT HIREDATE
FROM EMP
WHERE ENAME ='KING');

15.WAQTD NAME AND SAL ALONG WITH ANNUAL SAL FOR ALL EMPLOYEES
WHOS SAL IS LESS THAN BLAKE AND MORE THAN 3500

*SELECT ENAME , SAL , SAL*12*
FROM EMP
WHERE SAL > 3500 AND SAL < (SELECT SAL
FROM EMP
WHERE ENAME ='BLAKE);

16.WAQTD ALL THE DETAILS OF EMPLOYEES WHO EARN MORE THAN SCOTT
BUT LESS THAN KING

*SELECT **
FROM EMP
WHERE SAL > (SELECT SAL
FROM EMP
WHERE ENAME ='SCOTT') AND SAL < (SELECT SAL
FROM EMP
WHERE ENAME ='KING');

17.WAQTD NAME OF THE EMPLOYEES WHOS NAME STARTS WITH ‘A’ AND
WORKS IN THE SAME DEPT AS BLAKE

SELECT ENAME
FROM EMP
WHERE ENAME LIKE 'A%' AND DEPTNO = (SELECT DEPTNO
FROM EMP
WHERE ENAME ='BLAKE') ;

18.WAQTD NAME AND COMM IF EMPLOYEES EARN COMISSION AND WORK IN
THE SAME DESIGNATION AS SMITH

SELECT ENAME , COMM
FROM EMP
WHERE COMM IS NOT NULL AND JOB = (SELECT JOB
FROM EMP
WHERE ENAME ='SMITH');

19.WAQTD DETAILS OF ALL THE EMPLOYEES WORKING AS

CLERK IN THE
SAME DEPT AS TURNER

```
SELECT *
FROM EMP
WHERE JOB ='CLERK'  AND DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME ='TURNER' ) ;
```

20.WAQTD ENAME, SAL AND DESIGNATION OF THE EMPLOYEES WHOS
ANNUAL SALARY IS MORE THAN SMITH AND LESS THAN KING

```
SELECT ENAME , SAL , JOB
FROM EMP
WHERE SAL*12 > ( SELECT SAL *12
FROM EMP
WHERE ENAME ='SMITH') AND  SAL < ( SELECT SAL *12
FROM EMP
WHERE ENAME ='KING' );
```

DAY 12

Saturday, August 1, 2020 9:38 AM

CASE-2 : Whenever the data to be selected and the condition to be executed are present in different tables we use Sub Query .

Example :

Emp

EID	ENAME	SAL	DEPTNO
1	ALLEN	1000	20
2	BLAKE	2000	10
3	CLARK	3000	30
4	MILLER	1500	10
5	ADAMS	2500	20

DEPT

DEPTNO	DNAME	LOC
10	D1	L1
20	D2	L2
30	D3	L3

1. WAQTD deptno of the employee whose name is Miller .

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

2. WAQTD **dname** of the employee whose name is **Miller** .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
                 FROM EMP
                 WHERE ENAME ='MILLER') ;
```

3. WAQTD Location of ADAMS

```
SELECT LOC
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME ='ADAMS' ) ;
```

4. WAQTD names of the employees working in Location L2.

```
SELECT ENAME
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE LOC ='L2' ) ;
```

5. WAQTD number of employees working in dept D3 .

```
SELECT COUNT(*)
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
```

```
FROM DEPT
WHERE DNAME ='D3' ) ;
```

6. WAQTD ename , sal of all the employee earning more than Scott and working in dept 20 .

```
SELECT ENAME , SAL
FROM EMP
WHERE DEPTNO = 20 AND SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='SCOTT' ) ;
```

7. WAQTD all the details of the employee working as a Manager In the dept Accounting .

```
SELECT *
FROM EMP
WHERE JOB ='MANAGER' AND
DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE DNAME ='ACCOUNTING' )  ;
```

8. WAQTD all the details of the employee working in the same designation as Miller and works in location New York .

```
SELECT *
FROM EMP
WHERE JOB = ( SELECT JOB
FROM EMP
WHERE ENAME ='MILLER' )  AND DEPTNO = ( SELECT
DEPTNO FROM DEPT WHERE LOC ='NEW YORK' ) ;
```

9. WAQTD number of employees working as a clerk in the same deptno as SMITH and earning more than KING hired after MARTIN in the location BOSTON .

```
SELECT COUNT(*)
FROM EMP
WHERE JOB ='CLERK' AND
DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME ='SMITH') AND
SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='KING' ) AND
HIREDATE > ( SELECT HIREDATE
FROM EMP
WHERE ENAME ='MARTIN' ) AND
DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE LOC ='BOSTON' ) ;
```

10. WAQTD maximum salary given to a person working in

DALLAS .

```
SELECT MAX( SAL )
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE LOC ='DALLAS' ) ;
```

ASSIGNMENT ON CASE 2 :

21.WAQTD DNAME OF THE EMPLOYEES WHOS NAME IS SMITH
22.WAQTD DNAME AND LOC OF THE EMPLOYEE WHOS ENAME IS KING
23.WAQTD LOC OF THE EMP WHOS EMPLOYEE NUMBER IS 7902
24.WAQTD DNAME AND LOC ALONG WITH DEPTNO OF THE EMPLOYEE WHOS NAME ENDS WITH 'R' .
25.WAQTD DNAME OF THE EMPLOYEE WHOS DESIGNATION IS PRESIDENT
26.WAQTD NAMES OF THE EMPLOYEES WORKING IN ACCOUNTING DEPARTMENT
27.WAQTD ENAME AND SALARIES OF THE EMPLOYEES WHO ARE WORKING IN THE LOCATION CHICAGO
28.WAQTD DETAILS OF THE EMPLOYEES WORKING IN SALES
29.WAQTD DETAILS OF THE EMP ALONG WITH ANNUAL SALARY IF EMPLOYEES ARE WORKING IN NEW YORK
30.WAQTD NAMES OF EMPLOYEES WORKING IN OPERATIONS DEPARTMENT

ASSIGNMENT ON CASE 1 & 2

31.WAQTD NAMES OF THE EMPLOYEES EARNING MORE THAN SCOTT IN ACCOUNTING DEPT
32.WAQTD DETAILS OF THE EMPLOYEES WORKING AS MANAGER IN THE LOCATION CHICAGO
33.WAQTD NAME AND SAL OF THE EMPLOYEES EARNING MORE THAN KING IN THE DEPT ACCOUNTING
34.WAQTD DETAILS OF THE EMPLOYEES WORKING AS SALESMAN IN THE DEPARTEMENT SALES
35.WAQTD NAME , SAL , JOB , HIREDATE OF THE EMPLOYEES WORKING IN OPERATIONS DEPARTMENT AND HIRED BEFORE KING
36.DISPLAY ALL THE EMPLOYEES WHOSE DEPARTMET NAMES ENDING 'S'.
37.WAQTD DNAME OF THE EMPLOYEES WHOS NAMES HAS CHARACTER 'A' IN IT .
38.WAQTD DNAME AND LOC OF THE EMPLOYEES WHOS SALARY IS RUPEES 800 .
39.WAQTD DNAME OF THE EMPLOYEES WHO EARN COMISSION
40.WAQTD LOC OF THE EMPLOYEES IF THEY EARN COMISSION IN DEPT 40

MAX & MIN :

EID	ENAME	SAL	DEPTNO
1	ALLEN	1000	20
2	BLAKE	2000	10
3	CLARK	3000	30
4	MILLER	1500	10
5	ADAMS	2500	20

1. WAQTD maximum salary of an employee .

   SELECT MAX(SAL)
   FROM EMP ;

2. WAQTD name of the employee getting maximum salary .

   SELECT **ENAME** , MAX(SAL)
   FROM EMP ;

   SELECT ENAME
   FROM EMP
   WHERE SAL = **MAX(SAL)** ;

   SELECT ENAME
   FROM EMP
   WHERE SAL = (SELECT MAX(SAL)
   FROM EMP) ;

3. WAQTD name and salary earned by the employee getting Minimum salary .

   SELECT ENAME , SAL
   FROM EMP
   WHERE SAL = (SELECT MIN(SAL)
   FROM EMP) ;

ASSIGNMENT ON MAX & MIN :

41.WAQTD NAME OF THE EMPLOYEE EARNING MAXIMUM SALARY
42.WAQTD NAME OF THE EMPLOYEE EARNING MINIMUM SALARY
43.WAQTD NAME AND HIREDATE OF THE EMPLOYEE HIRED BEFORE
ALL THE EMPLOYEES (FIRST EMP)
44.WAQTD NAME AND HIREDATE OF THE EMPLOYEES HIRED AT THE LAST
45.WAQTD NAME, COMM OF THE EMPLOYEE WHO EARNS MIN COMISSION
46.WAQTD NAME, SAL AND COMM OF THE EMPLOYEE EARNING MAXIMUM
COMISSION

47.WAQTD DETAILS OF THE EMPLOYEE WHO HAS GREATEST EMPNO
48.WAQTD DETAILS OF THE EMPLOYEES HAVING THE LEAST HIREDATE
49.WAQTD DETAILS OF THE EMPLOYEES EARNING LEAST ANNUAL SALARY
50.WAQTD NAME , ANNUAL SALARY OF THE EMPLOYEES IF THEIR ANNUAL SALARY IS MORE THAN ALL THE SALESMAN

ASSIGNMENT ANSWERS ON CASE 2 :

21.WAQTD DNAME OF THE EMPLOYEES WHOS NAME IS SMITH

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME ='SMITH' ) ;
```

22.WAQTD DNAME AND LOC OF THE EMPLOYEE WHOS ENAME IS KING

```
SELECT DNAME ,LOC
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME =''KING' ) ;
```

23.WAQTD LOC OF THE EMP WHOS EMPLOYEE NUMBER IS 7902

```
SELECT LOC
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE EMPNO=7902 ) ;
```

24.WAQTD DNAME AND LOC ALONG WITH DEPTNO OF THE EMPLOYEE WHO'S NAME ENDS WITH 'R'.

```
SELECT DNAME , LOC
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE ENAME LIKE '%R' ) ;
```

25.WAQTD DNAME OF THE EMPLOYEE WHOS DESIGNATION IS PRESIDENT

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE JOB = 'PRESIDENT'  ) ;
```

26.WAQTD NAMES OF THE EMPLOYEES WORKING IN

ACCOUNTING DEPARTMENT

```
SELECT ENAME
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE  DNAME = 'ACCOUNTING' );
```

27.WAQTD ENAME AND SALARIES OF THE EMPLOYEES WHO ARE WORKING IN
THE LOCATION ‘CHICAGO’

```
SELECT ENAME ,SAL
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE  LOC = 'CHICAGO' );
```

28.WAQTD DETAILS OF THE EMPLOYEES WORKING IN SALES

```
SELECT *
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE  DNAME ='SALES' );
```

29.WAQTD DETAILS OF THE EMP ALONG WITH ANNUAL SALARY IF EMPLOYEES
ARE WORKING IN NEW YORK

```
SELECT EMP.* , SAL*12
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE  LOC ='NEW YORK' );
```

30.WAQTD NAMES OF EMPLOYEES WORKING IN OPERATIONS DEPARTMENT

```
SELECT ENAME
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE  DNAME ='OPERATIONS' );
```

ANSWERS ON CASE 1 & 2 :

31.WAQTD NAMES OF THE EMPLOYEES EARNING MORE THAN SCOTT IN
ACCOUNTING DEPT

```
SELECT ENAME
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='SCOTT' )  AND DEPTNO = ( SELECT DEPTNO
                                       FROM DEPT
```

```
                                                    WHERE  DNAME =
'ACCOUNTING');
```

32.WAQTD DETAILS OF THE EMPLOYEES WORKING AS MANAGER IN THE
LOCATION CHICAGO

```
SELECT *
FROM EMP
WHERE JOB ='MANAGER'  AND DEPTNO = ( SELECT DEPTNO
                                                    FROM DEPT
                                                    WHERE  LOC
='CHICAGO');
```

33.WAQTD NAME AND SAL OF THE EMPLOYEES EARNING MORE THAN KING
IN THE DEPT ACCOUNTING

```
SELECT ENAME ,SAL
FROM EMP
WHERE SAL > ( SELECT SAL
FROM EMP
WHERE ENAME ='KING' )  AND DEPTNO = ( SELECT DEPTNO
                                                    FROM DEPT
                                                    WHERE  DNAME =
'ACCOUNTING');
```

34.WAQTD DETAILS OF THE EMPLOYEES WORKING AS SALESMAN IN THE
DEPARTEMENT SALES

```
SELECT *
FROM EMP
WHERE JOB ='SALESMAN'  AND DEPTNO = ( SELECT DEPTNO
                                                    FROM DEPT
                                                    WHERE  DNAME
='SALES');
```

35.WAQTD NAME , SAL , JOB , HIREDATE OF THE EMPLOYEES WORKING IN OPERATIONS DEPARTMENT AND HIRED BEFORE KING

```
SELECT ENAME  ,SAL , JOB , HIREDATE
FROM EMP
WHERE HIREDATE < ( SELECT HIREDATE
FROM EMP
WHERE ENAME ='KING')  AND DEPTNO = ( SELECT DEPTNO
                                                    FROM DEPT
                                                    WHERE  DNAME
='OPERATIONS');
```

36.DISPLAY ALL THE EMPLOYEES WHOSE DEPARTMET NAMES ENDING 'S'.

```
SELECT ENAME
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM DEPT
WHERE  DNAME LIKE '%S' );
```

37.WAQTD DNAME OF THE EMPLOYEES WHOS NAMES HAS CHARACTER ‘A’ IN IT .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO IN( SELECT DEPTNO
FROM EMP
WHERE ENAME LIKE '%A%' ) ;
```

38.WAQTD DNAME AND LOC OF THE EMPLOYEES WHOS SALARY IS RUPEES 800 .

```
SELECT DNAME , LOC
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE SAL = 800) ;
```

39.WAQTD DNAME OF THE EMPLOYEES WHO EARN COMISSION

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE COMM IS NOT NULL );
```

40.WAQTD LOC OF THE EMPLOYEES IF THEY EARN COMISSION IN DEPT 40

```
SELECT LOC
FROM DEPT
WHERE DEPTNO = 40 AND DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE COMM IS NOT NULL ) ;

SELECT LOC
FROM DEPT
WHERE  DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE COMM IS NOT NULL AND DEPTNO = 40 ) ;
```

ANSWERS ON MAX & MIN :

41.WAQTD NAME OF THE EMPLOYEE EARNING MAXIMUM SALARY

```
SELECT ENAME
FROM EMP
WHERE SAL = ( SELECT MAX(SAL)
FROM EMP );
```

42.WAQTD NAME OF THE EMPLOYEE EARNING MINIMUM SALARY

```
SELECT ENAME
FROM EMP
WHERE SAL = ( SELECT MIN(SAL)
FROM EMP );
```

43.WAQTD NAME AND HIREDATE OF THE EMPLOYEE HIRED BEFORE
ALL THE EMPLOYEES (FIRST EMP)

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE = ( SELECT MIN(HIREDATE)
FROM EMP );
```

44.WAQTD NAME AND HIREDATE OF THE EMPLOYEES HIRED AT THE LAST

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE = ( SELECT MAX(HIREDATE)
FROM EMP );
```

45.WAQTD NAME, COMM OF THE EMPLOYEE WHO EARNS MIN COMISSION

```
SELECT ENAME , COMM
FROM EMP
WHERE COMM= ( SELECT MIN(COMM)
FROM EMP );
```

46.WAQTD NAME, SAL AND COMM OF THE EMPLOYEE EARNING MAXIMUM COMISSION

```
SELECT ENAME ,SAL, COMM
FROM EMP
WHERE COMM= ( SELECT MAX(COMM)
FROM EMP );
```

47.WAQTD DETAILS OF THE EMPLOYEE WHO HAS GREATEST EMPNO

```
SELECT *
FROM EMP
WHERE EMPNO= ( SELECT MAX(EMPNO)
FROM EMP );
```

48.WAQTD DETAILS OF THE EMPLOYEES HAVING THE LEAST HIREDATE

```
SELECT *
FROM EMP
WHERE EMPNO= ( SELECT MIN(EMPNO)
FROM EMP );
```

49.WAQTD DETAILS OF THE EMPLOYEES EARNING LEAST ANNUAL SALARY

```
SELECT ENAME
FROM EMP
WHERE SAL*12= ( SELECT MIN(SAL*12)
FROM EMP );
```

50.WAQTD NAME , ANNUAL SALARY OF THE EMPLOYEES IF

THEIR ANNUAL
SALARY IS MORE THAN ALL THE SALESMAN

```
SELECT ENAME , SAL*12
FROM EMP
WHERE SAL*12 >  ( SELECT MAX(SAL*12)
FROM EMP
WHERE JOB ='SALESMAN' );
OR
SELECT ENAME , SAL*12
FROM EMP
WHERE SAL*12 > ALL ( SELECT SAL*12
FROM EMP
WHERE JOB ='SALESMAN' );
```

DAY 13

Monday, August 3, 2020 9:47 AM

TYPES OF SUB - QUERY :

1. **SINGLE ROW SUB QUERY**
2. **MULTI ROW SUB QUERY**

Example :

Emp

EID	ENAME	SAL	DEPTNO
1	ALLEN	1000	20
2	BLAKE	2000	10
3	CLARK	3000	30
4	MILLER	1500	10
5	SMITH	2500	10

DEPT

DEPTNO	DNAME	LOC
10	D1	L1
20	D2	L2
30	D3	L3

1. WAQTD dname of ALLEN .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO    =   ( SELECT DEPTNO
                      FROM EMP
                      WHERE ENAME = 'ALLEN' ) ;
```

20

2. WAQTD dnames of allen and smith .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO    =   ( SELECT DEPTNO
                      FROM EMP
                      WHERE ENAME IN
                      ( 'ALLEN' ,'SMITH' )  ) ;
```

20
10

DEPTNO
20
10
30
10
10

Here , since the sub query returns 2 records we cannot use '=' Op . We've to use IN Op .

1. SINGLE ROW SUB QUERY :

- If the sub query returns exactly 1 record / value we call it as Single Row Sub Query .
- *If it returns only 1 value then we can use the normal operators Or the Special Operators to compare the values .*

2. MULTI ROW SUB QUERY :

- If the sub query returns more than1 record / value we call it as Multi Row Sub Query .
- *If it returns more than 1 value then we **cannot use the normal operators** We have to use only Special Operators to compare the values .*

Note : *It is difficult to identify whether a query Belongs Single or Multi row So , it is always recommended to use Special Operators to Compare The values .*

1. WAQTD ename and salary of the employees earning *more than* Employees of dept 10 .

EID	ENAME	SAL	DEPTNO
1	ALLEN	1000	20
2	BLAKE	2000	10
3	CLARK	3000	30
4	MILLER	1500	10
5	SMITH	2500	10

2000
1500
2500

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

Here we cannot use > symbol to compare Multiple values .

We cant use IN or. NOT IN as well because It is used for = and != symbols .

*Therefore we have to use **Sub Query Operators** For Comparing Relational Operators such as (> , < , >= , <=) .*

Sub Query Operators :

1. **ALL :**

"It is special Op used along with a relational Op (> , < , > = , <=) to compare the values present at the RHS ".

- *ALL Op returns true if all the values at the RHS have satisfied the condition .*

Example :

2000

CLARK 3000

Example :

CLARK ,3000

2000
1500
2500

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL  >  ALL  ( SELECT SAL
                     FROM EMP
                     WHERE DEPTNO = 10 ) ;
```

SAL
1000
2000
3000
1500
2500

1000 > ALL (2000 , 1500 , 2500)

1000 > 2000	False
1000 > 1500	False
1000 > 2500	False

2000 > ALL (2000 , 1500 , 2500)

2000 > 2000	False
2000 > 1500	True
2000 > 2500	False

3000 > ALL (2000 , 1500 , 2500)

3000 > 2000	True
3000 > 1500	True
3000 > 2500	True

1500 > ALL (2000 , 1500 , 2500)

1500 > 2000	False
1500 > 1500	False
1500 > 2500	False

2500 > ALL (2000 , 1500 , 2500)

2500 > 2000	True
2500 > 1500	True
2500 > 2500	False

2. **ANY :**

"It is special Op used along with a relational Op (> , < , > = , <=) to compare the values present at the RHS ".

- *ANY Op returns true if one of the values at the RHS have satisfied the condition .*

Example :

2000
1500
2500

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL   > ANY   ( SELECT SAL
                      FROM EMP
                      WHERE DEPTNO = 10 ) ;
```

SAL
1000
2000
3000
1500
2500

1000 > ANY (2000 , 1500 , 2500)

1000 > 2000	False
1000 > 1500	False
1000 > 2500	False

2000 > ANY (2000 , 1500 , 2500)

2000 > 2000	False
2000 > 1500	True
2000 > 2500	False

3000 > ANY (2000 , 1500 , 2500)

3000 > 2000	True
3000 > 1500	True
3000 > 2500	True

1500 > ANY (2000 , 1500 , 2500)

1500 > 2000	False
1500 > 1500	False
1500 > 2500	False

2500 > ANY (2000 , 1500 , 2500)

2500 > 2000	True
2500 > 1500	True
2500 > 2500	False

1. WAQTD name of the employee if the employee earns less than The employees working as salesman .

```
SELECT ENAME
FROM EMP
WHERE SAL < ALL ( SELECT SAL
```

```
FROM EMP
WHERE JOB='SALESMAN' ) ;
```

2. WAQTD name of the employee if the employee earns less than At least a salesman .

```
SELECT ENAME
FROM EMP
WHERE SAL < ANY ( SELECT SAL
FROM EMP
WHERE JOB ='SALESMAN' ) ;
```

3. WAQTD names of the employees earning more than ADAMS .

```
SELECT ENAME
FROM EMP
WHERE SAL > ALL  ( SELECT SAL
FROM EMP
WHERE ENAME ='ADAMS' ) ;
```

ASSIGNMENT ON TYPES OF SUB QUERY .

51.WAQTD NAME OF THE EMPLOYEES EARNING SALARY MORE THAN THE SALESMAN
52.WAQTD DETAILS OF THE EMPLOYEES HIRED AFTER ALL THE CLERKS
53.WAQTD NAME AND SALARY FOR ALL THE EMPLOYEES IF THEY ARE EARNING LESS THAN ATLEST A MANAGER
54.WAQTD NAME AND HIREDATE OF EMPLOYEES HIRED BEFORE ALL THE MANAGERS
55.WAQTD NAMES OF THE EMPLOYEES HIRED AFTER ALL THE MANAGERS AND EARNING SALARY MORE THAN ALL THE CLERKS
56.WAQTD DETAILS OF THE EMPLOYEES WORKING AS CLERK AND HIRED BEFORE ATLEST A SALESMAN
57.WAQTD DETAILS OF EMPLOYEES WORKING IN ACCOUNTING OR SALES DEPT
58.WAQTD DEPARTMENT NAMES OF THE EMPOYEES WITH NAME SMITH , KING AND MILLER
59.WAQTD DETAILS OF EMPLOYEES WORKING NEWYORK OR CHICAGO
60.WAQTD EMP NAMES IF EMPLOYEES ARE HIRED AFTER ALL THE EMPLOYEES OF DEPT 10

NESTED SUB QUERY :

" A sub query written inside a sub query is known as Nested Subquery "

SAL
1000

- WE CAN NEST ABOUT **255** SUB QUERIES

1000
2000
4000
3000
5000

1. WAQTD maximum salary given to an employee .

   SELECT MAX(SAL) **5000**
   FROM EMP ;

2. WAQTD second maximum salary given to an employee .

   4000
   SELECT MAX(SAL)
   FROM MP **5000**
   WHERE SAL < (SELECT MAX(SAL)
   FROM EMP) ;

SAL
1000
2000
4000
3000
5000

3. WAQTD 3rd maximum salary .

   SELECT MAX(SAL) 3000
   FROM EMP
   WHERE SAL < (SELECT MAX(SAL) 4000
   FROM EMP
   WHERE SAL < (SELECT MAX(SAL) 5000
   FROM EMP))

4. WAQTD 4th maximum salary .

   SELECT MAX(SAL) 2000
   FROM EMP
   WHERE SAL < (SELECT MAX(SAL) 3000
   FROM EMP
   WHERE SAL < (SELECT MAX(SAL) 4000
   FROM EMP
   WHERE SAL < (SELECT MAX(SAL) 5000
   FROM EMP)))

5. WAQTD 3 minimum salary .

   SELECT MIN(SAL)

```
FROM EMP
WHERE SAL > ( SELECT MIN(SAL )
              FROM EMP
              WHERE SAL > ( SELECT MIN ( SAL )
                            FROM EMP ) ) ;
```

6. WAQTD Dept name of the employee getting 2nd Minimum salary .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE SAL = (SELECT MIN( SAL )
FROM EMP
WHERE SAL > ( SELECT MIN( SAL )
FROM EMP ) ) );
```

REMEMBER :

MAXIMUM	**MAX( )**	**<**
MINIMUM	**MIN( )**	**>**

ASSIGNMENT ON NESTED SUB QUERY :

61.WAQTD 2ND MINIMUM SALARY
62.WAQTD 5TH MAXIMUM SALARY
63.WAQTD NAME OF THE EMPLOYEE EARNING 3RD MAXIMUM SALARY
64.WAQTD EMPNO OF THE EMPLOYEE EARNING 2D MAXIMUM SALARY
65.WAQTD DEPARTMENT NAME OF AN EMPLOYEE GETTING 4TH MAX SAL
66.WAQTD DETAILS OF THE EMPLOYEE WHO WAS HIRED 2nd
67.WAQTD NAME OF THE EMPLOYEE HIRED BEFORE THE LAST EMPLOYEE
68.WAQTD LOC OF THE EMPLOYEE WHO WAS HIRED FIRST
69.WAQTD DETAILS OF THE EMPLOYEE EARNING 7TH MINIMUM SALARY
70.WAQTD DNAME OF EMPLOYEE GETTING 2ND MAXIMUM SALARY

DAY 14

Tuesday, August 4, 2020 9:37 AM

EMPLOYEE AND MANAGER RELATION :

EID	ENAME	MGR
1	ALLEN	3
2	SMITH	1
3	JAMES	2
4	KING	3

CASE 1 :

- WAQTD name of Allen's manager .

JAMES

```
SELECT ENAME
FROM EMP                3
WHERE EID = ( SELECT MGR
              FROM EMP
              WHERE ENAME ='ALLEN' )
```

1
2
3
4

- WAQTD name of SMITH's manager .

```
SELECT ENAME
FROM EMP
WHERE EID = ( SELECT MGR
              FROM EMP
              WHERE ENAME ='SMITH' ) ;
```

- WAQTD name of SMITH's manager's manager .

EID	ENAME	MGR
1	ALLEN	3
2	SMITH	1
3	JAMES	2
4	KING	3

```
SELECT ENAME
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
```

```
WHERE ENAME ='SMITH' ) ) ;
```

- WAQTD dname of King's Manager .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
WHERE ENAME ='KING' ) ) ;
```

- WAQTD Location of Adams's manager's manager .

```
SELECT LOC
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
WHERE ENAME ='ADAMS' ) ) ) ;
```

CASE -2

- WAQTD Names of the employees reporting to KING.

```
SELECT ENAME
FROM EMP
WHERE MGR = ( SELECT EID
FROM EMP
WHERE ENAME ='KING' ) ;
```

- WAQTD Name and salary given to the employees reporting To James .

```
SELECT ENAME , SAL
FROM EMP
WHERE MGR = ( SELECT EID
FROM EMP
WHERE ENAME ='JAMES' ) ;
```

To find **Manager**	Select **MGR** in Sub Q
To find **Employees**	Select **EID** in Sub Q

- WAQTD dname of the employee reporting to President .

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
```

```
FROM EMP
WHERE MGR = ( SELECT EID
FROM EMP
WHERE JOB ='PRESIDENT' ) ) ;
```

- WAQTD Department details of the employees who are reporting to MILLER .

```
SELECT *
FROM DEPT
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE MGR = ( SELECT EID
FROM EMP
WHERE ENAME ='MILLER' ));
```

ASSIGNMENT ON EMP AND MANAGER RELATION .

71.WAQTD SMITHS REPORTING MANAGER'S NAME
72.WAQTD ADAMS MANAGER'S MANAGER NAME
73.WAQTD DNAME OF JONES MANAGER
74.WAQTD MILLER'S MANAGER'S SALARY
75.WAQTD LOC OF SMITH'S MANAGER'S MANAGER.
76.WAQTD NAME OF THE EMPLOYEES REPORTING TO BLAKE
77.WAQTD NUMBER OF EMPLPOYEES REPORTING TO KING
78.WAQTD DETAILS OF THE EMPLOYEES REPORTING TO JONES
79.WAQTD ENAMES OF THE EMPLOYEES REPORTING TO BLAKE'S MANAGER
80.WAQTD NUMBER OF EMPLOYEES REPORTING TO FORD'S MANAGER

SUB QUERY :

- **What is Sub Query ?**
- **Explain ? (draw)**
- **Why ? When ?**
- **Types of Sub Query**
  - **Single Row Sub Query**
  - **Multi Row Sub Query**
- **Sub Query Operators**
  - **ALL**
  - **ANY**
- **Nested Sub Query .**

71.WAQTD SMITHS REPORTING MANAGER'S NAME

```
SELECT ENAME
FROM EMP
```

```
WHERE EID=( SELECT MGR
FROM EMP
WHERE ENAME ='SMITH'  );
```

72.WAQTD ADAMS MANAGER'S MANAGER NAME

```
SELECT ENAME
FROM EMP
WHERE EID=( SELECT MGR
FROM EMP
WHERE ENAME ='ADAMS'  );
```

73.WAQTD DNAME OF JONES MANAGER

```
SELECT DNAME
FROM EMP
WHERE DEPTNO = ( SELECT DEPTNO
FROM EMP
WHERE EID=( SELECT MGR
FROM EMP
WHERE ENAME ='JONES'  ));
```

74.WAQTD MILLER'S MANAGER'S SALARY

```
SELECT SAL
FROM EMP
WHERE EID=( SELECT MGR
FROM EMP
WHERE ENAME ='MILLER'  );
```

75.WAQTD LOC OF SMITH'S MANAGER'S MANAGER.

```
SELECT LOC
FROM EMP
WHERE DEPTNO = (SELECT DEPTNO
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
WHERE EID=( SELECT MGR
FROM EMP
WHERE ENAME ='JONES'  )));
```

76.WAQTD NAME OF THE EMPLOYEES REPORTING TO BLAKE

```
SELECT ENAME
FROM EMP
WHERE MGR=( SELECT EID
FROM EMP
WHERE ENAME ='BLAKE'  );
```

77.WAQTD NUMBER OF EMPLPOYEES REPORTING TO KING

```
SELECT COUNT(ENAME)
FROM EMP
WHERE MGR=( SELECT EID
FROM EMP
WHERE ENAME ='KING'  );
```

78.WAQTD DETAILS OF THE EMPLOYEES REPORTING TO

JONES

```
SELECT *
FROM EMP
WHERE MGR=( SELECT EID
FROM EMP
WHERE ENAME ='JONES'  );
```

79.WAQTD ENAMES OF THE EMPLOYEES REPORTING TO BLAKE'S MANAGER

```
SELECT ENAME
FROM EMP
WHERE MGR =( SELECT EID
FROM EMP
WHERE EID = ( SELECT MGR
FROM EMP
WHERE ENAME ='BLAKE' ) );
```

OR

```
SELECT ENAME
FROM EMP
WHERE MGR =( SELECT MGR
FROM EMP
WHERE ENAME ='BLAKE'  );
```

80.WAQTD NUMBER OF EMPLOYEES REPORTING TO FORD'S MANAGER

```
SELECT COUNT(ENAME)
FROM EMP
WHERE MGR =( SELECT MGR
FROM EMP
WHERE ENAME ='FORD'  );
```

DAY 15

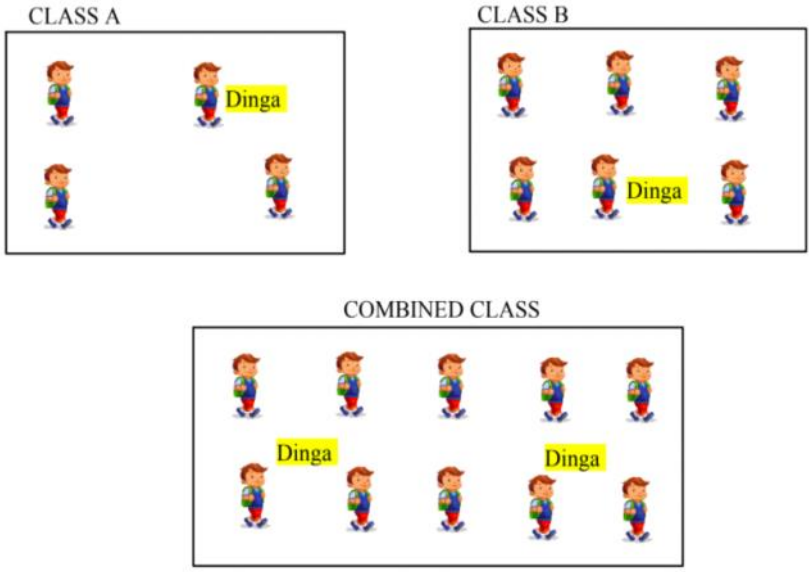
Wednesday, August 5, 2020 9:33 AM

JOINS

"The process of retrieval of data from multiple tables simultaneously is known as JOINS ".

WHY ? WHEN ?

Whenever the attributes is to be selected from both the tables we use Joins



Types of JOINS .

We have 5 types of joins

1. CARTESIAN JOIN / CROSS JOIN
2. INNER JOIN / EQUI JOIN
3. OUTER JOIN
    i. LEFT OUTER JOIN
    ii. RIGHT OUTER JOIN
    iii. FULL OUTER JOIN
4. SELF JOIN
5. NATURAL JOIN .

1. **CARTESIAN JOIN / CROSS JOIN :**

   In Cartesian Join a record from table 1 will be merged with All the records of table 2 .

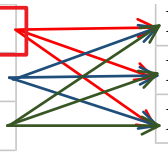
EMP

ENAME	DEPTNO
A	20
B	30

DEPT

DNAME	DEPTNO
D1	10
D2	20

A	20
B	30
C	10



D1	10
D2	20
D3	30

- **Number of Columns in the Result table :** will be equivalent to the summations of columns present in both the tables .

  Number of Col = Number of Col T1 + Number of Col T2
  2 + 2
  = 4 Columns

- **Number of Rows in the Result table :** will be equivalent to the product of number of rows present in the both the tables .

  Number of Rows = Number of Rows T1 x Number of Rows T2
  3 x 3
  = 9 Rows .

Result Table :

ENAME	DEPTNO	DNAME	DEPTNO
A	20	D1	10
A	20	D2	20
A	20	D3	30
B	30	D1	10
B	30	D2	20
B	30	D3	30
C	10	D1	10
C	10	D2	20
C	10	D3	30

SYNTAX:

1. **ANSI** [American National Standard Institute]

   SELECT Column_Name
   FROM Table_Name1 **CROSS JOIN** Table_Name2 ;

2. **Oracle**

   SELECT Column_Name
   FROM Table_Name1 **,** Table_Name2 ;

Example :

1. WAQTD ename and dept name for all the employees .

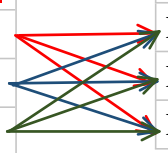
```
SELECT ENAME , DNAME
FROM EMP , DEPT ;

SELECT ENAME , DNAME
FROM EMP CROSS JOIN DEPT ;
```

2. **INNER JOIN :**

"It is used to Obtain only Matching Records "
Or " A records which has a Pair " .

EMP

ENAME	DEPTNO
A	20
B	30
C	10



DEPT

DNAME	DEPTNO
D1	10
D2	20
D3	30

JOIN Condition : It is a condition on which the two tables Are merged .

Syntax: Table_Name1.Col_Name = Table_Name2.Col_Name

Join Condition :EMP.DEPTNO = DEPT.DEPTNO

20 = 10	False
20 = 20	True
20 = 30	False

30 = 10	False
30 = 20	False
30 = 30	True

10 = 10	True
10 = 20	False
10 = 30	False

Result Table :

ENAME	EMP.DEPTNO	DNAME	DEPT.DEPTNO
A	20	D2	20
B	30	D3	30
C	10	D1	10

SYNTAX:

1. **ANSI** [American National Standard Institute]

```
SELECT Column_Name
FROM Table_Name1 INNER JOIN Table_Name2
ON < JOIN_CONDITION> ;
```

FROM Table_Name1 **INNER JOIN** Table_Name2
ON < JOIN_CONDITION> ;

SELECT *
FROM EMP **INNER JOIN** DEPT
ON EMP.DEPTNO = DEPT.DEPTNO ;

2. **Oracle**

SELECT Column_Name
FROM Table_Name1 **,** Table_Name2
WHERE <JOIN_CONDITION > ;

SELECT *
FROM EMP **,** DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO ;

1. WAQTD ename and dept name for all the employees .

   SELECT ENAME , DNAME
   FROM EMP , DEPT
   WHERE EMP.DEPTNO = DEPT.DEPTNO ;

2. WAQTD ename and loc for all the employees working as Manager .

   SELECT ENAME , LOC
   FROM EMP , DEPT
   WHERE EMP.DEPTNO = DEPT.DEPTNO AND JOB ='MANAGER' ;

3. WAQTD ename , sal and dname of the employee working as Clerk in dept 20 with a salary of more than 1800 .

   SELECT ENAME , SAL , DNAME
   FROM EMP , DEPT
   WHERE EMP.DEPTNO =DEPT.DEPTNO AND
   EMP.DEPTNO = 20 AND JOB ='CLERK' AND SAL > 1800 ;

4. WAQTD ename deptno , dname and loc of the employee earning more than 2000 in New York .

   SELECT ENAME , **EMP.DEPTNO** , DNAME
   FROM EMP , DEPT
   WHERE EMP.DEPTNO = DEPT.DEPTNO AND SAL > 2000 AND LOC ='NEW YORK' ;

ASSIGNMENT ON INNER JOIN :

1.NAME OF THE EMPLOYEE AND HIS LOCATION OF ALL THE EMPLOYEES .
2.WAQTD DNAME AND SALARY FOR ALL THE EMPLOYEE WORKING IN ACCOUNTING.

3.WAQTD DNAME AND ANNUAL SALARY FOR ALL EMPLOYEES WHOS SALARY IS MORE THAN 2340
4.WAQTD ENAME AND DNAME FOR EMPLOYEES HAVING CAHARACTER 'A' IN THEIR DNAME
5.WAQTD ENAME AND DNAME FOR ALL THE EMPLOYEES WORKING AS SALESMAN
6.WADTD DNAME AND JOB FOR ALL THE EMPLOYEES WHOS JOB AND DNAME STARTS WITH CHARACTER 'S'
7.WAQTD DNAME AND MGR NO FOR EMPLOYEES REPORTING TO 7839
8.WAQTD DNAME AND HIREDATE FOR EMPLOYEES HIRED AFTER 83 INTO ACCOUNTING OR RESEARCH DEPT
9.WAQTD ENAME AND DNAME OF THE EMPLOYEES WHO ARE GETTING COMM IN DEPT 10 OR 30
10.WAQTD DNAME AND EMPNO FOR ALL THE EMPLOYEES WHO'S EMPNO ARE (7839,7902) AND ARE WORKING IN LOC NEW YORK.

Answers :

1.NAME OF THE EMPLOYEE AND HIS LOCATION OF ALL THE EMPLOYEES .

```
SELECT ENAME , LOC
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO ;
```

2.WAQTD DNAME AND SALARY FOR ALL THE EMPLOYEE WORKING IN
ACCOUNTING.

```
SELECT DNAME , SAL
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND DNAME ='ACCOUNTING';
```

3.WAQTD DNAME AND ANNUAL SALARY FOR ALL EMPLOYEES WHOS SALARY
IS MORE THAN 2340

```
SELECT DNAME , SAL*12
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND SAL > 2340 ;
```

4.WAQTD ENAME AND DNAME FOR EMPLOYEES HAVING CAHARACTER 'A'
IN THEIR DNAME

```
SELECT ENAME , DNAME
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND ENAME LIKE '%A%' ;
```

5.WAQTD ENAME AND DNAME FOR ALL THE EMPLOYEES WORKING AS SALESMAN

```
SELECT ENAME , DNAME
```

Customer

CNAME	CID

5.WAQTD ENAME AND DNAME FOR ALL THE EMPLOYEES WORKING AS SALESMAN

SELECT ENAME , DNAME
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND JOB ='SALESMAN' ;

Customer

CNAME	CID
X	101
Y	102
Z	103

6.WADTD DNAME AND JOB FOR ALL THE EMPLOYEES WHOS JOB AND DNAME
STARTS WITH CHARACTER 'S'

SELECT DNAME ,JOB
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND JOB LIKE 'S%' AND DNAME LIKE 'S%' ;

7.WAQTD DNAME AND MGR NO FOR EMPLOYEES REPORTING TO 7839

SELECT DNAME , MGR
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND MGR = 7839 ;

8.WAQTD DNAME AND HIREDATE FOR EMPLOYEES HIRED AFTER 83 INTO
ACCOUNTING OR RESEARCH DEPT

SELECT DNAME , HIREDATE
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND HIREDATE > '31-DEC-83' AND DNAME IN
('ACCOUNTING' ,'RESEARCH');

9.WAQTD ENAME AND DNAME OF THE EMPLOYEES WHO ARE GETTING COMM
IN DEPT 10 OR 30

SELECT ENAME , DNAME
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND COMM IS NOT NULL AND EMP.DEPTNO IN (10 , 30) ;

10.WAQTD DNAME AND EMPNO FOR ALL THE EMPLOYEES WHO'S EMPNO ARE
(7839,7902) AND ARE WORKING IN LOC NEW YORK.

SELECT DNAME , EMPNO
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND EMPNO IN (7839,7902) AND LOC = 'NEW YORK' ;

DAY 16

Thursday, August 6, 2020 9:34 AM

OUTER JOIN

"It is used to Obtain Un-Matched Records "

1. **Left Outer Join :**

   "It is used to obtain Un-Matched Records of Left Table Along with Matching Records ".

Example :

EMP

ENAME	DEPTNO
A	20
B	Null
C	10
D	Null

Left

DEPT

DNAME	DEPTNO
D1	10
D2	20
D3	30
D4	40

Right

Result Table :

ENAME	EMP.DEPTNO	DNAME	DEPT.DEPTNO
A	20	D2	20
C	10	D1	10
B	Null	Null	Null
D	Null	Null	Null

SYNTAX:

1. **ANSI** [American National Standard Institute]

```
SELECT Column_Name
FROM Table_Name1 LEFT [OUTER] JOIN Table_Name2
ON < JOIN_CONDITION> ;

SELECT *
FROM EMP LEFT JOIN DEPT
ON EMP.DEPTNO = DEPT.DEPTNO ;
```

2. **Oracle**

```
SELECT Column_Name
```

2. **Oracle**

```
SELECT Column_Name
FROM Table_Name1 , Table_Name2
WHERE Table1.Col_Name = Table2.Col_Name (+) ;

SELECT *
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO (+) ;
```

- WAQTD names and dnames of all the employees even though the employees Don't work in any dept .

```
SELECT ENAME , DNAME
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO(+) ;
```

ENAME	DNAME
A	D2
C	D1
B	Null
D	Null

2. **Right Outer Join :**

"It is used to obtain Un-Matched Records of Right Table Along with Matching Records ".

Example :

EMP

ENAME	DEPTNO
A	20
B	Null
C	10
D	Null

Left

DEPT

DNAME	DEPTNO
D1	10
D2	20
D3	30
D4	40

Right

Result Table :

ENAME	EMP.DEPTNO	DNAME	DEPT.DEPTNO
A	20	D2	20
C	10	D1	10
Null	Null	D3	30

A	20	D2	20
C	10	D1	10
Null	Null	D3	30
Null	Null	D4	40

SYNTAX:

1. **ANSI** [American National Standard Institute]

   SELECT Column_Name
   FROM Table_Name1 **RIGHT[OUTER] JOIN** Table_Name2
   ON < JOIN_CONDITION> ;

   SELECT *
   FROM EMP **RIGHT JOIN** DEPT
   ON EMP.DEPTNO = DEPT.DEPTNO ;

2. **Oracle**

   SELECT Column_Name
   FROM Table_Name1 **,** Table_Name2
   WHERE Table1.Col_Name **(+)** = Table2.Col_Name ;

   SELECT *
   FROM EMP **,** DEPT
   WHERE EMP.DEPTNO**(+)** = DEPT.DEPTNO ;

- WAQTD names and dnames of all the employees even though the there are no employees in a dept .

  SELECT ENAME , DNAME
  FROM EMP , DEPT
  WHERE EMP.DEPTNO(+) = DEPT.DEPTNO ;

ENAME	DNAME
A	D2
C	D1
Null	D3
Null	D4

3. **Full Outer Join :**

   "It is used to obtain Un-Matched Records of both Left & Right Table Along with Matching Records ".

Example :

EMP **DEPT**

Example :

EMP

ENAME	DEPTNO
A	20
B	Null
C	10
D	Null

Left

DEPT

DNAME	DEPTNO
D1	10
D2	20
D3	30
D4	40

Right

Result Table :

ENAME	EMP.DEPTNO	DNAME	DEPT.DEPTNO
A	20	D2	20
C	10	D1	10
B	Null	Null	Null
D	Null	Null	Null
Null	Null	D3	30
Null	Null	D4	40

SYNTAX:

1. **ANSI** [American National Standard Institute]

```
SELECT Column_Name
FROM Table_Name1 FULL [OUTER] JOIN Table_Name2
ON < JOIN_CONDITION> ;

SELECT *
FROM EMP FULL JOIN DEPT
ON EMP.DEPTNO = DEPT.DEPTNO ;
```

- WAQTD names and dnames of all the employees and depts even though the employees Don't work in any dept and a dept having no employees .

```
SELECT ENAME , DNAME
FROM EMP FULL OUTER  DEPT
ON EMP.DEPTNO = DEPT.DEPTNO;
```

ENAME	DNAME
A	D2
C	D1
B	Null
D	Null

Null	D3
Null	D4

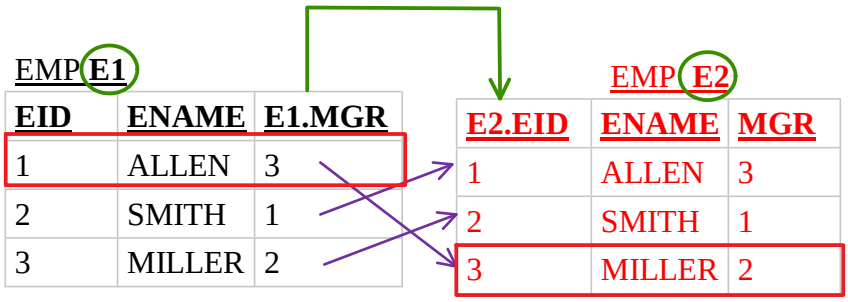
SELF JOIN :

"Joining a table by itself is known as Self Join "

Why ? / When ?

"Whenever the data to select is in the same table but present In different records we use self-join ".

Example :



EMP E1

EID	ENAME	E1.MGR
1	ALLEN	3
2	SMITH	1
3	MILLER	2

EMP E2

E2.EID	ENAME	MGR
1	ALLEN	3
2	SMITH	1
3	MILLER	2

Join Condition : **E1.MGR = E2.EID**

Result table :

E1.eid	E1.ename	E1.mgr	E2.eid	E2.ename	E2.mgr
1	ALLEN	3	3	MILLER	2
2	SMITH	1	1	ALLEN	3
3	MILLER	2	2	SMITH	1

Employees Details - E1 Managers Details - E2

SYNTAX:

1. **ANSI** [American National Standard Institute]

```
SELECT Column_Name
FROM Table_Name1 JOIN Table_Name2
ON < JOIN_CONDITION> ;

SELECT *
FROM EMP E1 JOIN EMP E2
ON E1.MGR = E2.EID ;
```

2. **Oracle**

```
SELECT Column_Name
FROM Table_Name1T1 , Table_Name2 T2
```

```
SELECT Column_Name
FROM Table_Name1T1  , Table_Name2 T2
WHERE < Join_Condition > ;

SELECT *
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EID  ;
```

1. WAQTD Ename and Manager's name for all the employees .

   ```
   SELECT E1.ENAME , E2.ENAME
   FROM EMP E1 , EMP E2
   WHERE E1.MGR = E2.EMPNO ;
   ```

2. WAQTD Ename , sal along with manager's name and manager's salary for all the employees .

   ```
   SELECT E1.ENAME , E1.SAL , E2.ENAME , E2.SAL
   FROM EMP E1 , EMP E2
   WHERE E1.MGR = E2.EMPNO ;
   ```

3. WAQTD ename , manager's name along with their deptno If employee is working as clerk .

   ```
   SELECT E1.ENAME , E2.ENAME , E1.DEPTNO ,
   E2.DEPTNO
   FROM EMP E1 , EMP E2
   WHERE E1.MGR = E2.EMPNO AND E1.JOB='CLERK' ;
   ```

4. WAQTD ename , manager's job if manager works as Analyst .

   ```
   SELECT E1.ENAME , E2.JOB
   FROM EMP E1 , EMP E2
   WHERE E1.MGR = E2.EMPNO
   AND E2.JOB ='ANALYST' ;
   ```

5. WAQTD ename and manager's name along with their job if emp and manager are working for same designation .

   ```
   SELECT E1.ENAME , E2.ENAME , E1.JOB , E2.JOB
   FROM EMP E1 , EMP E2
   WHERE E1.MGR = E2.EMPNO
   AND E1.JOB = E2.JOB ;
   ```

6. WAQTD ename emp salary manager's name manager's salary If manager earns more than employee .

   ```
   SELECT E1.ENAME , E1.SAL , E2.ENAME , E2.SAL
   FROM EMP E1 , EMP E2
   WHERE E1.MGR = E2.EMPNO
   AND E2.SAL > E1.SAL ;
   ```

7. WAQTD ename and manager's name along with manager's commission if manager earns commission .

```
SELECT E1.ENAME , E2.ENAME , E2.COMM
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E2.COMM IS NOT NULL ;
```

NOTE : TO join 'N' number of tables we need to write 'N-1' number of join conditions

ASSIGNMENT ON SELF JOIN :

1.WAQTD NAME OF THE EMPLOYEE AND HIS MANAGER'S NAME IF EMPLOYEE IS WORKING AS CLERK

2.WAQTD NAME OF THE EMPLOYEE AND MANAGER'S DESIGNATION IF MANAGER WORKS IN DEPT 10 OR 20

3.WAQTD NAME OF THE EMP AND MANAGERS SALARY IF EMPLOYEE AND MANAGER BOTH EARN MORE THAN 2300

4.WAQTD EMP NAME AND MANAGER'S HIREDATE IF EMPLOYEE WAS HIRED BEFORE1982

5.WAQTD EMP NAME AND MANAGER'S COMM IF EMPLOYEE WORKS AS SALESMAN AND MANAGER WORKS IN DEPT 30

6.WAQTD EMP NAME AND MANAGER NAME AND THEIR SALARIES IF EMPLOYEE EARNS MORE THAN MANAGER

7.WAQTD EMP NAME AND HIREDATE , MANAGER NAME AND HIREDATE IF
MANAGER WAS HIRED BEFORE EMPLOYEE

8.WAQTD EMP NAME AND MANAGER NAME IF BOTH ARE WORKING IN SAME JOB

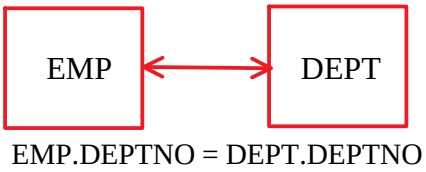
9.WAQTD EMP NAME AND MANAGER NAME IF MANAGER IS WORKING AS ACTUAL MANAGER

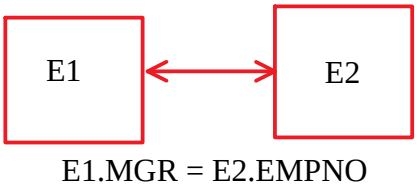
10.WAQTD EMP NAME AND MANAGER NAME ALONG WITH THEIR ANNUAL SALARIES IF EMPLOYEE WORKS IN DEPT 10 , 20 AND MANAGER'S SAL IS GREATER THAN EMPLOYEES SALARY .

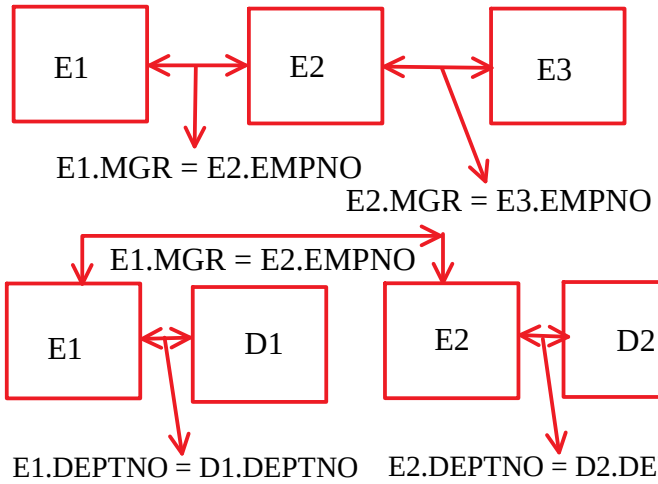
11.WAQTD EMPLOYEE'S NAME AND MANAGER'S DESIGNATION FOR ALL THE EMPLOYEES

12.WAQTD EMPLOYEE'S NAME AND MANAGER'S SALARY FOR ALL THE EMPLOYEES IF MANAGER'S SALARY ENDS WITH 50

Examples :







5. **NATURAL JOIN :**

"It behaves as **INNER JOIN** if there is a relation between the given two tables , else it behaves as **CROSS JOIN" .**

Syntax:

ANSI :

```
SELECT Col_Name
FROM Table_Name1 NATURAL JOIN Table_Name2;
```

Emp

ENAME	DEPTNO
A	20
B	30
C	10

DEPT

DNAME	DEPTNO
D1	10
D2	20
D3	30

Result Table : has a relation (inner join)

DEPTNO	ENAME	DNAME
20	A	D2
30	B	D3
10	C	D1

Emp

ENAME	DEPTNO
A	20
B	30
C	10

CUSTOMER

CNAME	CID
X	101
Y	102
Z	103

Result Table : has no relation (cross join)

ENAME	DEPTNO	CNAME	CID
A	20	X	101
A	20	Y	102
A	20	Z	103
B	30	X	101
B	30	Y	102
B	30	Z	103
C	10	X	101
C	10	Y	102
C	10	Z	103

QUESTIONS:

1.WAQTD NAME OF THE EMPLOYEE AND HIS MANAGER'S NAME IF EMPLOYEE IS WORKING AS CLERK

```
SELECT E1.ENAME ,  E2.ENAME
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E1.JOB = 'CLERK';
```

2.WAQTD NAME OF THE EMPLOYEE AND MANAGER'S DESIGNATION IF MANAGER WORKS IN DEPT 10 OR 20

```
SELECT E1.ENAME ,  E2.JOB
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E2.DEPTNO IN ( 10 , 20 );
```

3.WAQTD NAME OF THE EMP AND MANAGERS SALARY IF EMPLOYEE AND MANAGER BOTH EARN MORE THAN 2300

```
SELECT E1.ENAME ,  E2.SAL
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
```

AND E1.SAL > 2300 AND E2.SAL>2300 ;

4.WAQTD EMP NAME AND MANAGER'S HIREDATE IF EMPLOYEE WAS HIRED BEFORE1982

```
SELECT E1.ENAME ,  E2.HIREDATE
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E1.HIREDATE < '01-JAN-82' ;
```

5.WAQTD EMP NAME AND MANAGER'S COMM IF EMPLOYEE WORKS AS SALESMAN AND MANAGER WORKS IN DEPT 30

```
SELECT E1.ENAME ,  E2.COMM
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E1.JOB = 'SALESMAN' AND E2.DEPTNO = 30 ; ;
```

6.WAQTD EMP NAME AND MANAGER NAME AND THEIR SALARIES IF EMPLOYEE EARNS MORE THAN MANAGER

```
SELECT E1.ENAME, E1.SAL ,  E2.ENAME , E2.SAL
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E1.SAL > E2.SAL ;
```

7.WAQTD EMP NAME AND HIREDATE , MANAGER NAME AND HIREDATE IF
MANAGER WAS HIRED BEFORE EMPLOYEE

```
SELECT E1.ENAME ,E1.HIREDATE ,  E2.ENAME , E2.HIREDATE
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E2.HIREDATE < E1.HIREDATE ;
```

8.WAQTD EMP NAME AND MANAGER NAME IF BOTH ARE WORKING IN SAME JOB

```
SELECT E1.ENAME ,  E2.ENAME
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E1.JOB = E2.JOB ;
```

9.WAQTD EMP NAME AND MANAGER NAME IF MANAGER IS WORKING AS ACTUAL MANAGER

```
SELECT E1.ENAME ,  E2.ENAME
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E2.JOB = 'MANAGER';
```

10.WAQTD EMP NAME AND MANAGER NAME ALONG WITH THEIR ANNUAL SALARIES IF EMPLOYEE WORKS IN DEPT 10 , 20 AND MANAGER'S SAL IS GREATER THAN EMPLOYEES SALARY .

```
SELECT E1.ENAME , E1.SAL*12 ,  E2.ENAME , E2.SAL*12
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E1.DEPTNO IN ( 10,20) AND E2.SAL > E1.SAL ;
```

11.WAQTD EMPLOYEE'S NAME AND MANAGER'S DESIGNATION FOR ALL THE EMPLOYEES

```
SELECT E1.ENAME ,  E2.JOB
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO ;
```

12.WAQTD EMPLOYEE'S NAME AND MANAGER'S SALARY FOR ALL THE EMPLOYEES IF MANAGER'S SALARY ENDS WITH 50

```
SELECT E1.ENAME ,  E2.SAL
FROM EMP E1 , EMP E2
WHERE E1.MGR = E2.EMPNO
AND E2.SAL LIKE '%50' ;
```

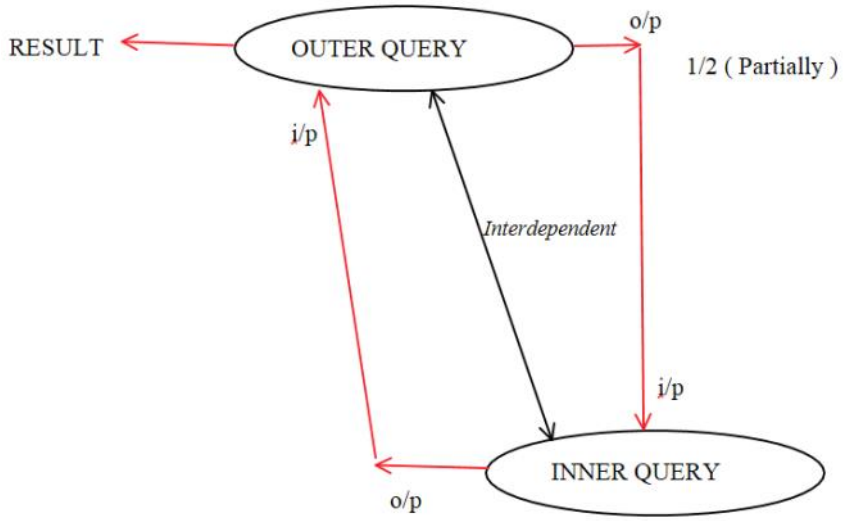
DAY 17

Friday, August 7, 2020 9:50 AM

CO - RELATED SUB QUERY

" *A query written inside another query such that the outer query and the inner query are Dependent on each other , this is known as Co-Related Sub-Query* ".

WORKING PRINCIPLE :



Let us consider two queries inner and outer query respectively ,

1. Outer query executes first but partially
2. The partially executed output is given as an input to the inner Query
3. The inner query executes completely and generates an output
4. The output of inner query is fed as an input to the Outer query and Outer
   Query produces the result .
5. Therefore, we can state that the outer query and the inner query both are INTERDEPENDENT (dependent on each other) .

NOTE :

i. In co-related sub query a Join condition is a must , And must be written only in the Inner Query .
ii. Co-Related sub query works with the principles of both SUB QUERY & JOINS .

DIFFERENCE BETWEEN SUB QUERY AND CO RELATED SUB QUERY .

SUB QUERY	CO-RELATED SUB QUERY
Inner query executes first	Outer query executes first
Outer query is dependent on inner query	Both are interdependent

Join condition not mandatory	Join condition is mandatory and must be written in inner query
Outer query executes Once	Outer query executes Twice .

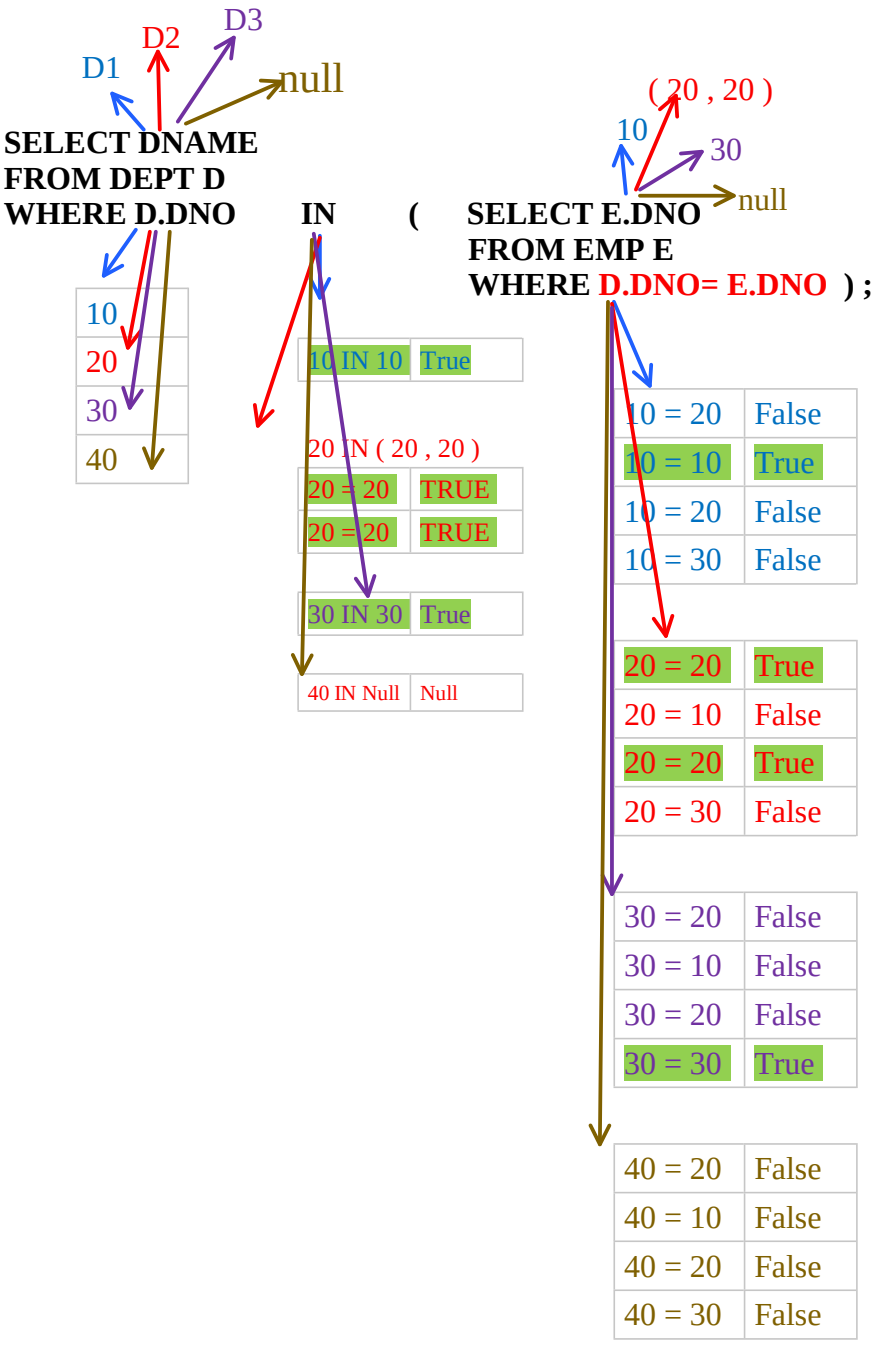
Example :

DEPT

DNAME	DNO
D1	10
D2	20
D3	30
D4	40

EMP

ENAME	DNO
A	20
B	10
C	20
D	30



1. WAQTD dnames in which there are employees working .

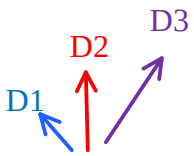
```
SELECT DNAME
FROM DEPT D
WHERE D.DEPTNO IN ( SELECT E.DEPTNO
                    FROM EMP E
                    WHERE D.DEPTNO = E.DEPTNO ) ;
```

2. WAQTD dname in which there are no employees working .

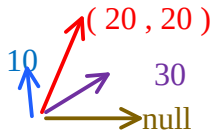
```
SELECT DNAME
FROM DEPT D
WHERE D.DEPTNO NOT IN ( SELECT E.DEPTNO
                        FROM EMP E
                        WHERE D.DEPTNO = E.DEPTNO ) ;
```

EXISTS & NOT EXISTS OPERATORS :

1. **EXISTS :** " Exists Op is a Unary Op (One Operand) which can accept One Operand Towards RHS and that Operand has to be A Co-related Sub Query "
   - *Exists Op returns true if the Sub Query returns Any value other than Null.*



SELECT DNAME
FROM DEPT D
WHERE EXISTS (SELECT E.DNO
FROM EMP E
WHERE D.DNO= E.DNO.) ;



10 = 20	False
10 = 10	True
10 = 20	False
10 = 30	False

20 = 20	True
20 = 10	False
20 = 20	True
20 = 30	False

30 = 20	False
30 = 10	False

30 = 20	False
30 = 10	False
30 = 20	False
30 = 30	True

40 = 20	False
40 = 10	False
40 = 20	False
40 = 30	False

2. **NOT EXISTS :** " Not Exists Op is a Unary Op (One Operand) which can accept
   One Operand Towards RHS and that Operand has to be
   A Co-related Sub Query "

➢ *Not Exists Op returns true if the Sub Query returns NULL .*

To Find MAX & MIN salary :

To find MAXIMUM salary :

```
SELECT SAL
FROM EMP E1
WHERE ( SELECT COUNT( DISTINCT SAL )
        FROM EMP E2
        WHERE E1.SAL < E2.SAL ) = N-1  ;
```

E1.SAL
1000
3000
2000
3000
2000
4000
5000

3 MAX SAL :

3000

```
SELECT SAL
FROM EMP E1                    2
WHERE ( SELECT COUNT( DISTINCT SAL )
        FROM EMP E2
        WHERE E1.SAL < E2.SAL ) = 2 ;
```

E1.SAL
1000
3000
2000
3000
2000
4000
5000

E2.SAL
1000
3000
2000
3000
2000
4000
5000

4000
5000

4000
5000

<u>2nd , 4th , 5th , 7th MAX salary</u>

```
SELECT SAL
FROM EMP E1
WHERE ( SELECT COUNT( DISTINCT SAL )
        FROM EMP E2
        WHERE E1.SAL < E2.SAL ) in ( 1 , 3, 4 , 6 )  ;
```

<u>To find MINUMUM salary :</u>

```
SELECT SAL
FROM EMP E1
WHERE ( SELECT COUNT( DISTINCT SAL )
        FROM EMP E2
        WHERE E1.SAL > E2.SAL ) = N-1  ;
```

DAY 18

Monday, August 10, 2020 9:47 AM

SINGLE ROW FUNCTIONS

1. LENGTH( )
2. CONCAT( )
3. UPPER( )
4. LOWER( )
5. INITCAP( )
6. REVERSE( )
7. SUSBTR( )
8. INSTR( )
9. REPLACE( )
10. MOD( )
11. TRUNC( )
12. ROUND( )
13. MONTHS_BETWEEN( )
14. LAST_DAY( )
15. TO_CHAR( )
16. NVL( )

1. **LENGTH :** "It is used to count the number of characters present In the given string ".

SYNTAX: **LENGTH ('string')**

Example :

- WAQT count number of characters present in 'SMITH' .

```
SELECT LENGTH ( ENAME )
FROM EMP
WHERE ENAME ='SMITH' ;
```

LENGTH(ENAME)
5

```
SELECT LENGTH( 'SMITH' )
FROM DUAL ;

SELECT LENGTH( 'HELLO WORLD' ) → 11
FROM DUAL;
```

NOTE : DUAL TABLE

It is a DUMMY table which has 1 col and 1 row . Which is used to output the result .

- DESC DUAL ;
- SELECT *
  FROM DUAL ;

2. **CONCAT( ) :** "It is used to join the given two strings '

SYNTAX : CONCAT ('string1' , 'String2')

Example :

Input : Smith
Output : Mr. Smith

```
SELECT CONCAT( 'Mr. ' , ENAME )
FROM EMP
WHERE ENAME ='SMITH' ;
```

3. **UPPER( )** : "It is used to convert a given string to upper case "

SYNTAX: UPPER ('string')

4. **LOWER( ) :**"It is used to convert a given string to lower case "

SYNTAX: LOWER('string')

5. **INITCAP( ):**"It is used to convert a given string to initial capital letter case ".

SYNTAX: INITCAP('string')

6. **REVERSE( ):** "It is used to reverse a given string ".

SYNTAX: REVERSE('string')

Example :

```
SELECT REVERSE( 'SMITH' ).
FROM DUAL  ;
```

REVERSE('SMITH')
HTIMS

```
SELECT UPPER( 'smith' ).
FROM DUAL  ;
```

UPPER('smith')
SMITH

```
SELECT LOWER( 'SMITH' ).
FROM DUAL  ;
```

LOWER('SMITH')
smith

```
SELECT INITCAP( 'SMITH' ).
FROM DUAL  ;
```

INITCAP('SMITH')
Smith

7. **SUBSTR :** "It is used to extract a part of string from the given Original string " .

SYNTAX: **SUBSTR** ('Original_String' , Position [, Length])

***NOTE**: Length is not mandatory ,*
If length is not mentioned then

Example :

SYNTAX: **SUBSTR** (Original_String , Position [, Length])

Example :

***NOTE**: Length is not mandatory ,*
If length is not mentioned then
Consider the complete string .

-ve	-7	-6	-5	-4	-3	-2	-1
	Q	**S**	**P**	**I**	**D**	**E**	**R**
+ve	1	2	3	4	5	6	7

Example :	SUBSTR('QSPIDER' , 2 , 3)	SPI
Example :	SUBSTR('QSPIDER' , 3 , 3)	PID
Example :	SUBSTR('QSPIDER' , 2)	SPIDER
Example :	SUBSTR('QSPIDER' , 1 , 6)	QSPIDE
Example :	SUBSTR('QSPIDER' , 4, 1)	I
Example :	SUBSTR('QSPIDER' , 1 , 1)	Q
Example :	SUBSTR('QSPIDER' , 7 , 1)	R
Example :	SUBSTR('QSPIDER' , 6)	ER
Example :	SUBSTR('QSPIDER' , 0 , 3)	QSP
Example :	SUBSTR('QSPIDER' , 6 , 6)	ER
Example :	SUBSTR('QSPIDER' , -2 , 1)	E
Example :	SUBSTR('QSPIDER' , -5 , 3)	PID
Example :	SUBSTR('QSPIDER' , -7 , 2)	QS
Example :	SUBSTR('QSPIDER' , -1)	R

- WAQT extract first 3 characters of the emp names .

```
SELECT SUBSTR(ENAME, 1,3)
FROM EMP;
```

- WAQT extract last 3 characters of the employee names .

```
SELECT SUBSTR(ENAME, -3 )
FROM EMP;
```

- WAQT to display **first half of employee names .**

ENAME	OUTPUT
SMITH	SM
MILLER	MIL
JONES	JO
WARD	WA

```
SELECT SUBSTR( ENAME , 1 ,  LENGTH( ENAME ) / 2 )
FROM EMP ;
```

SMITH	SUBSTR(ENAME , 1 , **LENGTH(ENAME) / 2**)

	SUBSTR('SMITH' , 1 , LENGTH ('SMITH') / 2)
	SUBSTR('SMITH' , 1 , 5 / 2)
	SUBSTR('SMITH' , 1 , 2)
	SM

WARD	SUBSTR(ENAME , 1 , LENGTH(ENAME) / 2)
	SUBSTR('WARD' , 1 , LENGTH ('WARD') / 2)
	SUBSTR('WARD' , 1 , 4 / 2)
	SUBSTR('WARD' , 1 , 2)
	WA

- WAQT to display **second half of employee names .**

ENAME	OUTPUT
SMITH	ITH
MILLER	LER
JONES	NES
WARD	RD

SELECT SUBSTR(ENAME , LENGTH(ENAME) / 2 + 1)
FROM EMP ;

SMITH	SUBSTR(ENAME , LENGTH(ENAME) / 2 +1)
	SUBSTR('SMITH' , LENGTH ('SMITH') / 2 +1)
	SUBSTR('SMITH' , 5 / 2 +1)
	SUBSTR('SMITH' , 3)
	ITH

WARD	SUBSTR(ENAME , LENGTH(ENAME) / 2+1)
	SUBSTR('WARD' , LENGTH ('WARD') / 2+1)
	SUBSTR('WARD' , 4 / 2 +1)
	SUBSTR('WARD' , 3)
	RD

8. **REPLACE ( ) :** "It is used to replace a string with another string in The original string.

Null

SYNTAX:**REPLACE** ('Original_String' , 'string' [, 'new_String'])

Example :	REPLACE ('**B**A**N**A**N**A**A**' , '**A**' , '**C**')	BCNCNC
Example :	REPLACE ('BANANA' , 'N' , 'ABC')	BAABCAABCA
Example :	REPLACE ('OPPO' , 'O' , 'J')	JPPJ
Example :	REPLACE ('BANANA' , 'A')	BNN
Example :	REPLACE ('ENGINEERING' , 'E')	NGINRING

Example :	REPLACE ('ENGINEERING' , 'E' , '123')	123N123123GINRING

NOTE : if the third argument is not mentioned the default Value of it is Null .

1. WAQTD the number of times char 'A' is present in BANANA !!!

```
SELECT LENGTH('BANANA') - LENGTH ( REPLACE( 'BANANA','A' )
FROM DUAL ;
```

Length ('BANANA') - LENGTH(REPLACE('BANANA','A'))
Length ('BANANA') - LENGH ('BNN')
6 - 3
= 3 times 'A' is present in BANANA

2. WAQTD to count number of time 'A' is present in 'MALAYALAM'

```
SELECT LENGTH('MALAYALAM') - LENGTH
( REPLACE( 'MALAYALAM','A' )
FROM DUAL ;
```

DAY 19

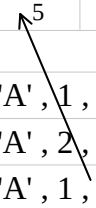
Tuesday, August 11, 2020 9:56 AM

9. **INSTR ( ) :** "*it is used to obtain the position in which the string is present in the Original string* ".
   It is used to search for a string in the Original string if present it returns the POSITION
   Else it returns **0** ".

   Syntax: **INSTR('Original_String' , 'String' , Position [, Occurrence])**

   Note : *if occurrence is not Mentioned then , the default value of Occurrence is 1* .

B	A	N	A	N	A
1	2	3	4	5	6



Example : INSTR('BANANA' , 'A' , 1 , 1)	POS: 2
Example : INSTR('BANANA' , 'A' , 2 , 1)	POS: 2
Example : INSTR('BANANA' , 'A' , 1 , 2)	POS: 4
Example : INSTR('BANANA' , 'A' , 1 , 3)	POS: 6
Example : INSTR('BANANA' , 'A' , 1 , 4)	POS: 0
Example : INSTR('BANANA' , 'A' , 4 , 2)	POS: 6
Example : INSTR('BANANA' , 'A' , 2)	POS: 2
Example : INSTR('BANANA' , 'N' , 2 , 1)	POS: 3
Example : INSTR('BANANA' , 'O' , 1 ,1)	POS: 0
Example : INSTR('BANANA' , 'NA' , 2 , 2)	POS: 5
Example : INSTR('BANANA' , 'A' , 3 , 3)	POS: 0
Example : INSTR('BANANA' , 'ANA' , 1 , 2)	POS: 4

1. WAQTD NAMES OF THE EMPLOYEES IF THEY HAVE CHAR 'A' PRESENT IN THEIR NAMES

   ```
   SELECT ENAME
   FROM EMP
   WHERE INSTR( ENAME , 'A' , 1 ,1 ) > 0 ;
   ```

2. WAQTD NAMES OF THE EMPLOYEES IF THEY HAVE CHAR 'A' PRESENT ATLEAST TWICE IN THEIR NAMES

   ```
   SELECT ENAME
   FROM EMP
   WHERE INSTR( ENAME , 'A' , 1 , 2 ) > 0 ;
   ```

3. WAQTD NAMES OF THE EMPLOYEES IF THEY HAVE CHAR 'A' PRESENT ATLEAST THRICE IN THEIR NAMES

   ```
   SELECT ENAME
   FROM EMP
   ```

WHERE INSTR(ENAME , 'A' , 1 , 3) > 0 ;

4. WAQTD NAMES OF THE EMPLOYEES IF THEY HAVE CHAR 'A' ***EXACTLY TWICE***

```
SELECT ENAME
FROM EMP
WHERE INSTR( ENAME , 'A' , 1 , 2 ) > 0 AND  INSTR( ENAME , 'A' , 1 , 3 ) = 0   ;
```

OR

```
SELECT ENAME
FROM EMP
WHERE (   LENGTH( ENAME ) - LENGTH( REPLACE( ENAME ,'A') )  ) = 2;
```

ALLEN	INSTR('ALLEN','A',1,2)	Pos:0	INSTR('ALLEN','A',1,3)	Pos:0
ADAMS	INSTR('ADAMS','A',1,2)	Pos:3	INSTR('ADAMS','A',1,3)	Pos:0
AATISH	INSTR('AATISH','A',1,2)	Pos:2	INSTR('AATISH','A',1,3)	Pos:0
AAA	INSTR('AAA' ,'A',1 ,2)	Pos:2	INSTR('AAA' ,'A',1 ,3)	Pos:3
MALAYALAM	INSTR('MALAYALAM' , 'A' , 1, 2)	Pos:4	INSTR('MALAYALAM' , 'A' , 1, 3)	Pos:6

ALLEN	LENGTH('ALLEN') - LENGTH(REPLACE('ALLEN' ,A'))	= 2
	5 - LENGTH('LLEN')	
	5 - 4	
	1	!= 2
ADAMS	**5 - LENGTH('DMS')**	
	5 - 3	
	2	= 2
AAAAO	**5 - LENGTH('O')**	
	5 - 1	
	4	!= 2

SINGLE ROW FUNCTIONS

10. MOD( )
11. TRUNC( )
12. ROUND( )
13. MONTHS_BETWEEN( )
14. LAST_DAY( )
15. TO_CHAR( )
16. NVL( )

10. MOD( ) : "It is used to obtain modulus/remainder of the given number "

Syntax: **MOD** (m , n) ⟶ n) m (

Example : SELECT MOD(5 , 2)
FROM DUAL ; ⟶ 1

1 . WAQTD ENAMES OF THE EMPLOYEES WHO EARN SALARY IN MULTIPLES OF 3

```
SELECT ENAME
FROM EMP
WHERE MOD( SAL , 3 ) = 0 ;
```

2. WAQTD DETAILS OF THE EMPLOYEE WHO HAVE ODD EID

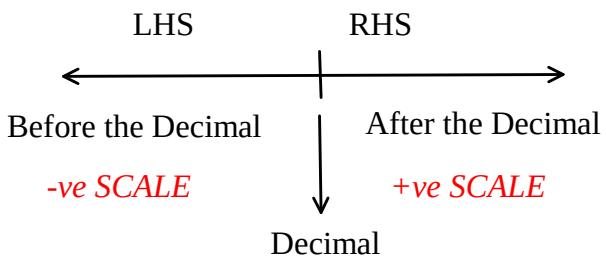
```
SELECT *
FROM EMP
WHERE MOD( EID , 2 ) = 1 ;
```

11. ROUND( ) : " It is used to Round-off the given number based on the scale value "

Syntax: **ROUND (Number [, Scale])**

The default value of scale is 0

Example : ROUND (5.6)	6
Example : ROUND (5.5)	6
Example : ROUND (5.4)	5
Example : ROUND (9.9)	10
Example : ROUND (9.4)	9
Example : ROUND (8.6 , 0)	9



When the scale is -ve it indicated the digits before the decimal
And the digit count begins from 1 .

ROUND (8421 .12 , -1)

Example : ROUND (8421.12 , -1)	8420
Example : ROUND (8426.12 , -1)	8430
Example : ROUND (154264.12 , -2)	154300
Example : ROUND (338222 , -4)	340000
Example : ROUND (2514 , -3)	3000

5.6	6	Rounding off to units place

When the scale is +ve it indicated the digits **after the decimal**
And the digit count begins from 0 .

Example : ROUND (124.23541 , 0)	124

(124.23541 , 0) = 124

And the digit count begins from 0 .

Example : ROUND (124.23541 , 0)	124
Example : ROUND (124. 23541 , 1)	124.2
Example : ROUND (124. 23541 , 2)	124.24
Example : ROUND (124. 2354391 , 5)	124.23544

(124.23541 , 0) = 124
(124.23541 , 1) = 124.2
(123.671263 8723 , 6) = 123.671264

12. TRUNC( ): "It is similar to ROUND() but it always rounds-off the given number to the lower value "

Syntax: TRUNC(Number [, Scale])

Example : TRUNC (5.6)	5
Example : TRUNC (5.5)	5
Example : TRUNC (5.4)	5
Example : TRUNC (9.9)	9
Example : TRUNC (9.4)	9
Example : TRUNC (8.6 , 0)	8
Example: TRUNC(451258.32541 , -5)	400000

NOTE :

DATE COMMANDS :

i. **SYSDATE** : " it is used to obtain Todays Date "
ii. **CURRERNT_DATE** : " it is also used to obtain todays date "
iii. **SYSTIMESTAMP** : "It is used to obtain date , time and time zone "

```
SQL> SELECT SYSDATE
  2  FROM DUAL ;

SYSDATE
---------------
17-MAY-20

SQL> SELECT CURRENT_DATE
  2  FROM DUAL ;

CURRENT_D
-----------------
17-MAY-20

SQL> SELECT SYSTIMESTAMP
  2  FROM DUAL ;

SYSTIMESTAMP
---------------------------------------------------
17-MAY-20 05.05.52.356000 PM +05:30
```

13. **MONTHS_BETWEEN( ) :"It is used to Obtain the number of months present between the Given two dates "**

Syntax: **MONTHS_BETWEEN** (DATE1 , DATE2)

```
SELECT TRUNC( MONTHS_BETWEEN( SYSDATE , HIREDATE ) ) || ' Months'
FROM EMP

TRUNC(MONTHS_BETWEEN(SYSDATE,HIREDATE))||'MONTH
--------------------------------------------------------------------------------
473 Months
470 Months
```

14. **LAST_DAY( ): " it is used to Obtain the last day in the particular of the given date" .**

Syntax: LAST_DAY(DATE) ;

```
SQL> SELECT LAST_DAY( SYSDATE )
  2  FROM DUAL ;

LAST_DAY
---------------
31-JUL-20
```

SYSDATE = 08-JUL-2020

15. **TO_CHAR( ) :"It is used to convert the given date into String format based on the Model given "**

Syntax: **TO_CHAR(DATE , 'Format _ Models')**

Format Models :

i. YEAR : TWENTY TWENTY
ii. YYYY : 2020
iii. YY : 20
iv. MONTH : JULY
v. MON : JUL
vi. MM : 07
vii. DAY : WEDNESDAY
viii. DY : WED
ix. DD : 08
x. D : 4 (day of the week)
xi. HH24 : 17 hours
xii. HH12 : 5 hours
xiii. MI : 22 minutes
xiv. SS : 53 seconds
xv. 'HH12:MI:SS' : 5 : 22 : 53
xvi. 'DD-MM-YY' : 17 - 05 - 20
xvii. 'MM-DD-YYYY' : 05 - 17 - 2020

1. WAQTD DETAILS OF THE EMPLOYEE WHO WAS HIRED ON A SUNDAY .

```
SELECT *
FROM EMP
WHERE TO_CHAR( HIREDATE , 'DAY' ) = 'SUNDAY' ;
```

2. WAQTD DETAILS OF AN EMPLOYEE HIRED ON MONDAY AT 10AM

```
SELECT *
FROM EMP
WHERE TO_CHAR( HIREDATE , 'D' ) = 2 AND TO_CHAR( HIREDATE , 'HH24' ) = 10 ;
```

16. NVL( ) : [NULL VALUE LOGIC] " It is used to eliminate the side effects of using null in arithmetic operations " .

ENAME	SAL	COMM
A	500	100
B	1000	NULL
C	2000	200
D	2000	NULL

WAQTD NAME AND TOTAL SALALRY OF ALL THE EMPLOYEES?

```
SELECT ENAME , SAL + COMM
FROM EMP ;
```

ENAME	SAL+COMM
A	600
B	NULL
C	2200
D	NULL

Null value logic : Syntax : **NVL (Argument1 , Argument2)**

Argument 1 : Here write any column / exp which can result In null .
Argument 2 : Here we write a numeric value which will be substituted if argument 1 results in Null ,
If argument 1 is NOT NULL then the same value will be considered .

```
SELECT ENAME , SAL + NVL ( COMM , 0 )
FROM EMP ;
```

A	500 + NVL (100 , 0)	500 + 100	600
B	1000 + NVL (null , 0)	1000 + 0	1000
C	2000 + NVL (200 , 0)	2000+200	2200
D	2000 + NVL(null , 0)	2000 + 0	2000

After using NVL

ENAME	SAL+nvl(COMM ,0)
A	600

ENAME	SAL+nvl(COMM ,0)
A	600
B	1000
C	2200
D	2000

1. List employees whose name having 4 characters

```
SELECT *
FROM EMP
WHERE LENGTH(ENAME)=4 ;
```

2. List employees whose job is having 7 characters

```
SELECT *
FROM EMP
WHERE LENGTH(JOB)=4;
```

3. Find out how many times letter 'S' occurs in 'qspiders'

```
SELECT LENGTH('QSPIDERS') - LENGTH( REPLACE( 'QSPIDERS' , 'S' ) )
FROM DUAL ;
```

4. List the employees whose job is having last 3 characters as 'man'

```
SELECT *
FROM EMP
WHERE SUBSTR( JOB , -3 ) = 'MAN' ;
```

5. List employees whose job is having first 3 characters as 'man'.

```
SELECT *
FROM EMP
WHERE SUBSTR( JOB , 1 , 3 ) = 'MAN' ;
```

6. Display all the names whose name is having exactly 1 'L'

```
SELECT ENAME
FROM EMP
WHERE INSTR( ENAME , 'L' , 1,1 ) != 0 AND INSTR( ENAME , 'L' , 1, 2 ) = 0 ;
```

OR

```
SELECT ENAME
FROM EMP
WHERE LENGTH( ENAME ) - LENGTH( REPLACE( ENAME , 'L' ) ) = 1 ;
```

7. Display dept names which are having letter 'O'

```
SELECT DNAME
FROM DEPT
WHERE INSTR(DNAME,'O',1,1 ) !=0 ;
```

9. Calculate number of L in string 'HELLLLL'

```
SELECT LENGTH('HELLLLL') - LENGTH( REPLACE( 'HELLLLL' , 'L' ) )
FROM DUAL ;
```

10. Display all the employees whose job has a string 'MAN'

```
SELECT *
FROM EMP
WHERE INSTR(JOB,'MAN',1,1 ) !=0 ;
```

11. Display all the employees whose job starts with string 'MAN'

```
SELECT *
FROM EMP
WHERE INSTR(JOB,'MAN',1,1 ) =1 ;
```

OR

```
SELECT *
FROM EMP
WHERE SUBSTR( JOB ,1,3) = 'MAN' ;
```

12. Display all the employees whose job ends with string 'MAN'

```
SELECT *
FROM EMP
WHERE SUBSTR( JOB , -3 ) = 'MAN' ;
```

13. Display first 3 characters of ename in lower case and rest everything in upper case.
If ename is 'QSPIDERS' then display this as ‘qspIDERS’

```
SELECT LOWER(SUBSTR('QSPIDERS',1,3)) || UPPER( SUSBTR('QSPIDERS' , 4) )
FROM DUAL ;
```

14. Display the result from emp table as below.
SMITH is a CLERK and gets salary 2000
Here SMITH is ename column, CLERK is JOB and 2000 is SAL column and rest everything is literal strings.

```
SELECT ENAME || ' IS A '|| JOB || ' AND GETS SALARY ' || SAL
FROM EMP
WHERE ENAME = 'SMITH' ;
```

15.list the employees hired on a Wednesday

```
SELECT *
FROM EMP
WHERE TO_CHAR( HIREDATE , 'DY' ) = WED ;
```

16.list the employees hired on a leap year

```
SELECT *
FROM EMP
WHERE MOD( TO_CHAR( HIREDATE , 'YY' ) , 4 ) = 0 ;
```

17.list the employees hired on a Sunday in the month of may

```
SELECT *
FROM EMP
WHERE  TO_CHAR( HIREDATE , 'DY' ) = 'SUN'  AND  TO_CHAR( HIREDATE , 'MON' ) = 'MAY' ;
```

DAY 20

Wednesday, August 12, 2020 8:48 AM

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

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
    );
```

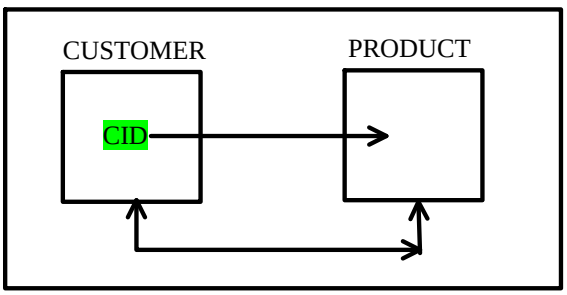
Example :

```
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 ;

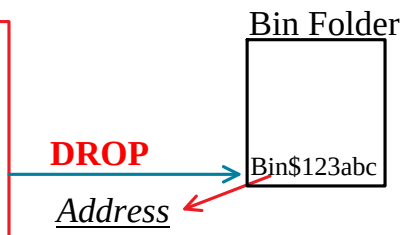
Syntax: DROP TABLE Table_Name ;

Example :

DATABASE

Cust

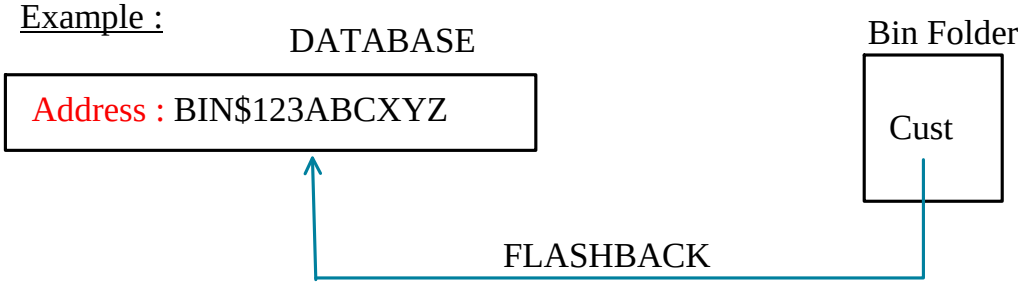
Cid	Cname	Phone_no	Address
1	A	1234567890	BANGALORE
2	B	1234567899	MYSORE
3	C	1234567880	MANGALORE



TO RECOVER THE TABLE :

Syntax: FLASHBACK TABLE Table_Name
TO BEFORE DROP ;

Example :

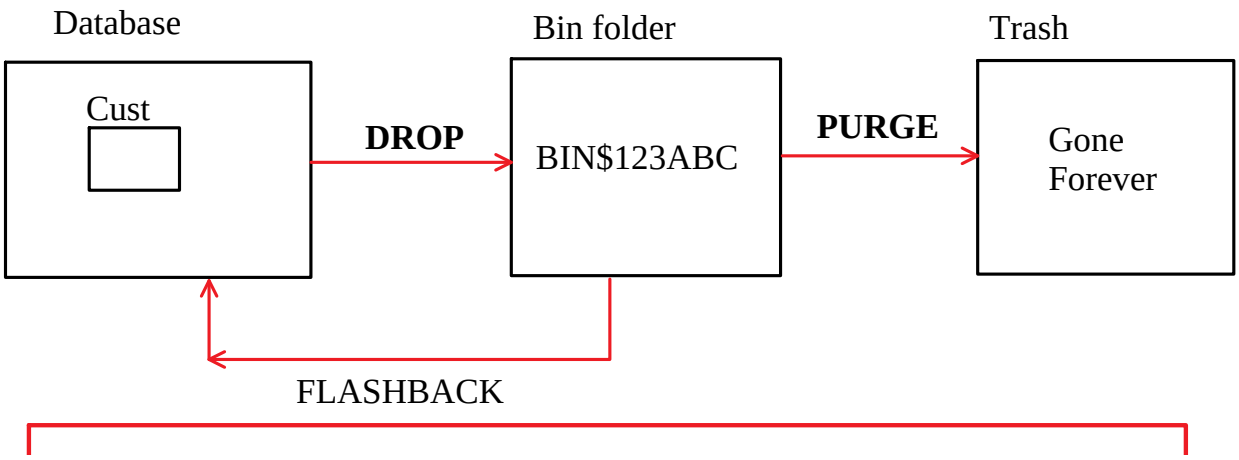


FLASHBACK TABLE Cust
TO BEFORE DROP ;

TO DELETE THE TABLE FROM BIN FOLDER :

Syntax: PURGE TABLE Table_Name ;

Example : PURGE TABLE Cust ;



FLASHBACK

NOTE : DDL STATEMENTS ARE AUTO-COMMIT STATEMENTS

DAY 21

Thursday, August 13, 2020 9:41 AM

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

8. Differentiate between TRUNCATE and DELETE statements .

TRUNCATE	**DELETE**
Belongs to DDL	Belongs to DML

Removes all the records from the Table permanently .	Removes a particular record from the Table .
Auto COMMIT	Not auto COMMIT .

3. TRANSACTION CONTROL LANGUAGE (TCL)

"It is used to control the transactions done on the database ".
The DML Operations performed on the Database are known as Transactions such as Insertion , Updating and Deletion .

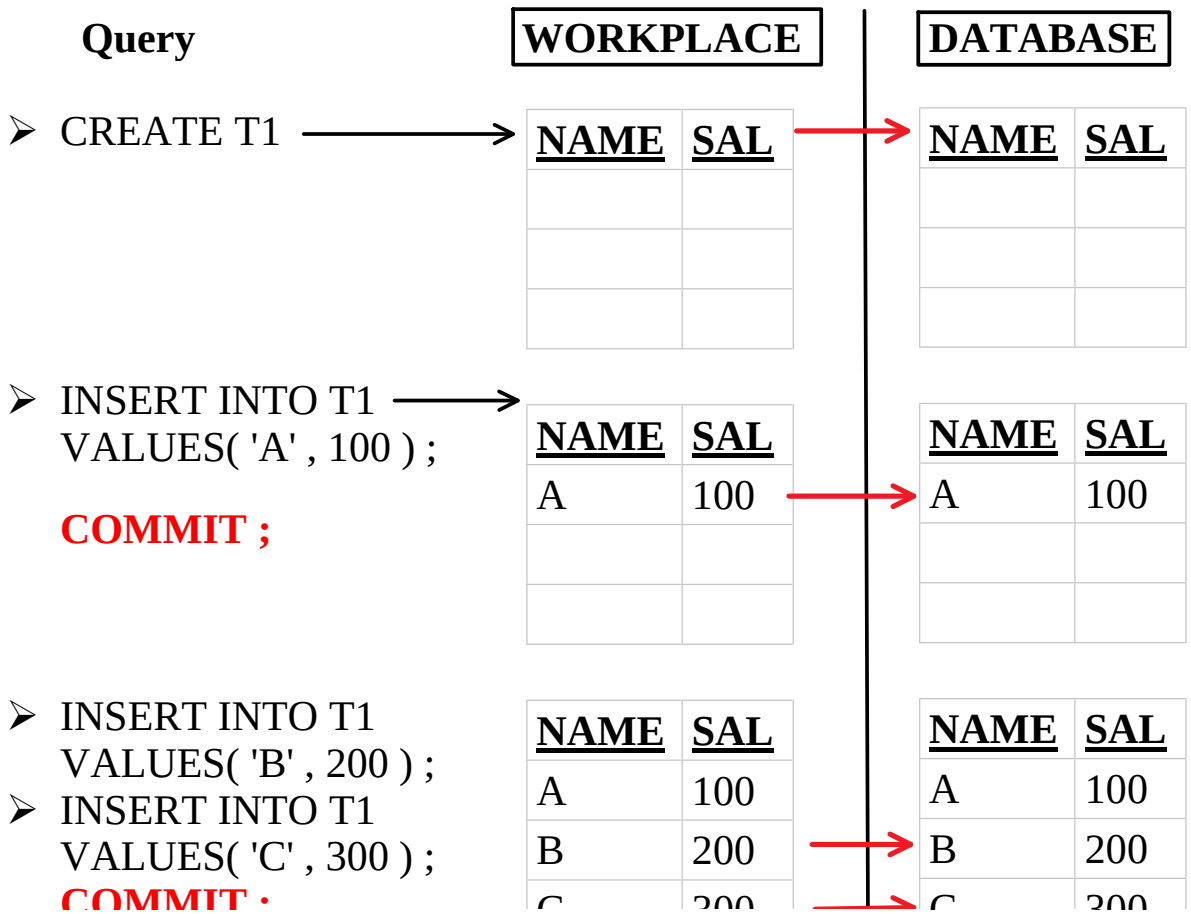
We have 3 Statements :

1. COMMIT
2. ROLLBACK
3. SAVEPOINT

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

Syntax: **COMMIT ;**

Example :



➢ INSERT INTO T1
VALUES('C' , 300) ;
COMMIT ;

B	200
C	300

B	200
C	300

NAME	SAL
A	1000
B	200
C	300

➢ UPDATE T1
SET SAL = 1000
WHERE NAME = 'A' ;

➢ **COMMIT ;**

NAME	SAL
A	1000
B	200
C	300

2. ROLLBACK :

This statement is used to Obtain only the saved data from the DB .
It will bring you to the point where you have committed for the last time .

SYNTAX: **ROLLBACK ;**

3. SAVEPOINT :

This statement is used to mark the positions or restoration points . (nothing related to DB) .

SYNTAX: **SAVEPOINT** Savepoint_Name ;

Example :

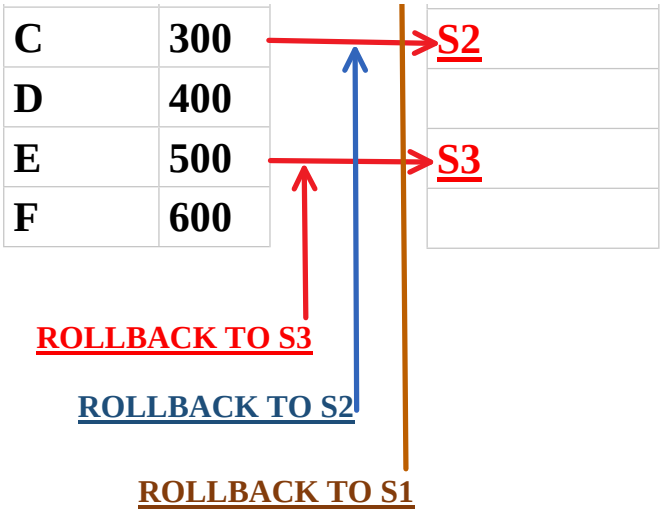
Query **WORKPLACE** **SAVEPOINT**

➢ INSERT INTO T1
VALUES('A',100) ;
➢ SAVEPOINT S1 ;
➢ INSERT INTO T1
VALUES('B' , 200) ;
➢ INSERT INTO T1

NAME	SAL
A	100
B	200
C	300
D	400

Savepoints
S1
S2

- INSERT INTO T1
  VALUES('B' , 200) ;
- INSERT INTO T1
  VALUES('C' , 300) ;
- SAVEPOINT S2 ;
- INSERT INTO T1
  VALUES('D' , 400) ;
- INSERT INTO T1
  VALUES('E' , 500) ;
- SAVEPOINT S3 ;
- INSERT INTO T1
  VALUES('F' , 600) ;



SYNTAX: **ROLLBACK TO** Savepoint_Name ;

4. DATA CONTROL LANGUAGE :

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

We have 2 statements :

1. GRANT
2. REVOKE

1.GRANT : THIS STATEMENT IS USED TO GIVE PERMISSION TO A USER .

```
SYNTAX: GRANT SQL_STATEMENT
        ON TABLE_NAME
        TO USER_NAME ;
```

2.REVOKE : THIS STATEMENT IS USED TO TAKE BACK THE PERMISSION FROM THE USER .

```
SYNTAX: REVOKE SQL_STATEMENT
        ON TABLE_NAME
        FROM USER_NAME ;
```

Example :

User 1 : SCOTT | **User 2 : HR**

User 1 : SCOTT

EMP

ENAME	SAL
A	100
B	200

User 2 : HR

```
SELECT *
FROM SCOTT.EMP ;
```

GRANT SELECT ON EMP TO HR ;

EMP

ENAME	SAL
A	100
B	200

```
SELECT *
FROM SCOTT.EMP ;
```

REVOKE SELECT ON EMP FROM HR ;

ENAME	SAL
A	100
B	200

```
SELECT *
FROM SCOTT.EMP ;
```

TRY !!!!

```
SQL> SHOW USER ;
USER is "SCOTT"
SQL> CONNECT
Enter user-name: HR
Enter password: *****
Connected.
SQL> SHOW USER ;
USER is "HR"
```

```
Connected.
SQL> SHOW USER ;
USER is "HR"
SQL> SELECT *
  2  FROM SCOTT.EMP;
FROM SCOTT.EMP
          *
ERROR at line 2:
ORA-00942: table or view does not exist


SQL> CONNECT
Enter user-name: SCOTT
Enter password: *****
Connected.
SQL> GRANT SELECT ON EMP TO HR;

Grant succeeded.

SQL> CONNECT
Enter user-name: HR
Enter password: *****
Connected.
SQL> SELECT *
  2  FROM SCOTT.EMP;

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

DAY 22

Friday, August 14, 2020 9:39 AM

What is Normalization ?

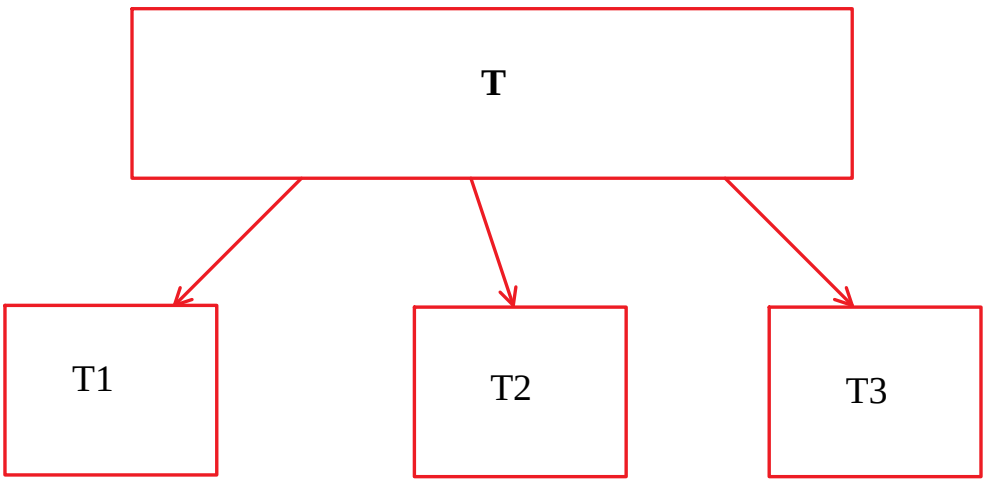
" It is the process of reducing a large table into smaller tables in order to remove redundancies and anomalies by identifying their functional dependencies is known as Normalization . "

Or

"The process of decomposing a large table into smaller table is known as Normalization ."

Or

"Reducing a table to its Normal Form is known as Normalization . "



What is **Normal Form** ?

A table without redundancies and anomalies are said to be in Normal Form .

Levels of Normal From .

1. **First Normal Form (1NF)**
2. **Second Normal Form (2NF)**
3. **Third Normal Form (3NF)**
4. **Boyce - Codd Normal Form (BCNF)**

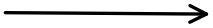
Note : **If any Table / entity is reduced to 3NF , then the table is said to be normalized.**

1. ***First Normal Form (1NF) :***

- No duplicates records .
- Multivalued data should not be present .

QSPIDERS

QID	NAME	COURSE
1	A	JAVA
2	B	JAVA , SQL
3	C	MT , SQL
1	A	MT



QID	NAME	C1	C2	C3
1	A	JAVA		MT
2	B	JAVA	SQL	
3	C		SQL	MT

2. ***Second Normal Form (2NF)***

- Table should be in 1NF
- Table should not have Partial Functional Dependency .

EMPLOYEE - (EID , ENAME , SAL , DEPTNO , DNAME , LOC)

Eid	ename	sal	Deptno	dname	Loc
1	A	100	10	D1	L1
2	B	120	20	D2	L2
3	C	320	10	D1	L1
4	D	251	10	D1	L1

Eid - ename ,sal
Deptno - dname , loc

:- (***Eid , deptno***) -> (Ename , Sal , Dname , Loc) composite key attribute results in PFD

R1 - (EID , ENAME , SAL)
R2 - (DEPTNO , DNAME , LOC)

Eid	ename	sal
1	A	100
2	B	120
3	C	320
4	D	251

Deptno	dname	Loc
10	D1	L1
20	D2	L2

3. ***Third Normal Form (3NF)***

- Table should be in 2NF .
- Table should not have Transitive Functional Dependency .

Employee - (EID , Ename , Sal , comm , Pin code , state , country)

EID -> ENAME
SAL
COMM
PINCODE -> STATE
COUNTRY

:- Transitive Functional Dep

R1- (eid , ename , comm)
R2- (pincode , state , country)