Introduction to Applications / Software

Thursday, June 11, 2020 9:41 AM

What is an Application ?

" Is a list of programs which helps us to perform a particular task "

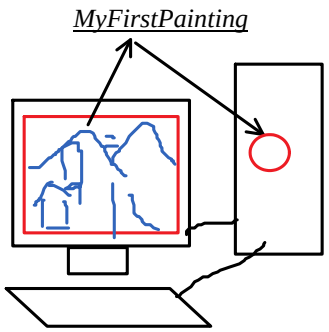
Example : whatsapp
Ms office
Web browser
Facebook
Intsagram
GTA Vice city

Types of Applications :

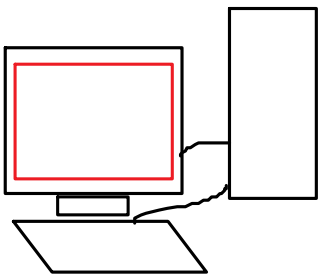
1. Stand Alone Application .
2. Web Application .
3. Client / Server Application (Mobile Application) .

1. **STAND ALONE APPLICATION :**

Dinga :



Dingi :



software installed in one computer and used by **only one** person.
For ex – Installing s/w of a Calculator, Adobe Photoshop, MS Office, AutoCAD, Paint etc..

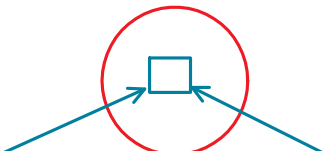
Advantages:

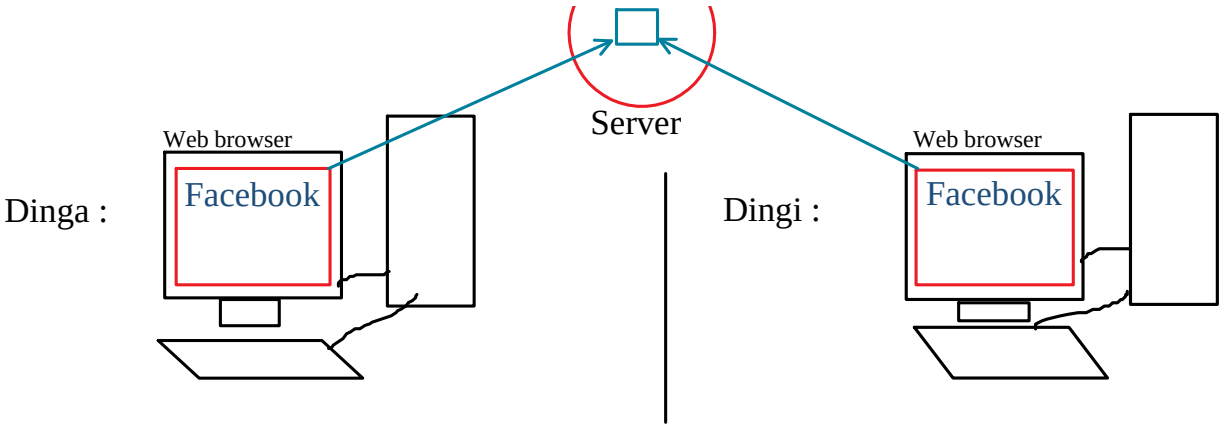
- Faster in access.
- Secured from Data hacking and virus.

Disadvantages:

- Single user access at a time.
- Installation is required.

2. **WEB APPLICATION :**

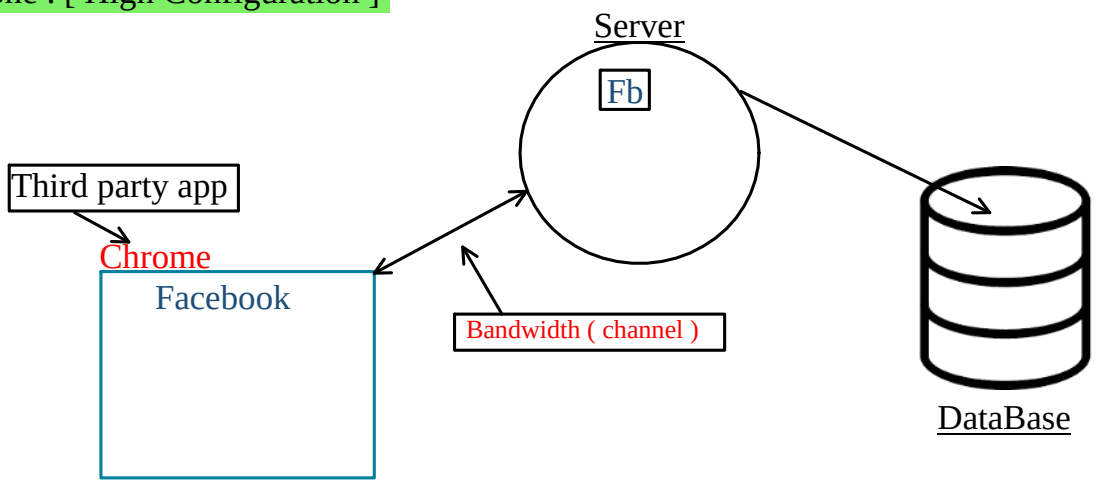




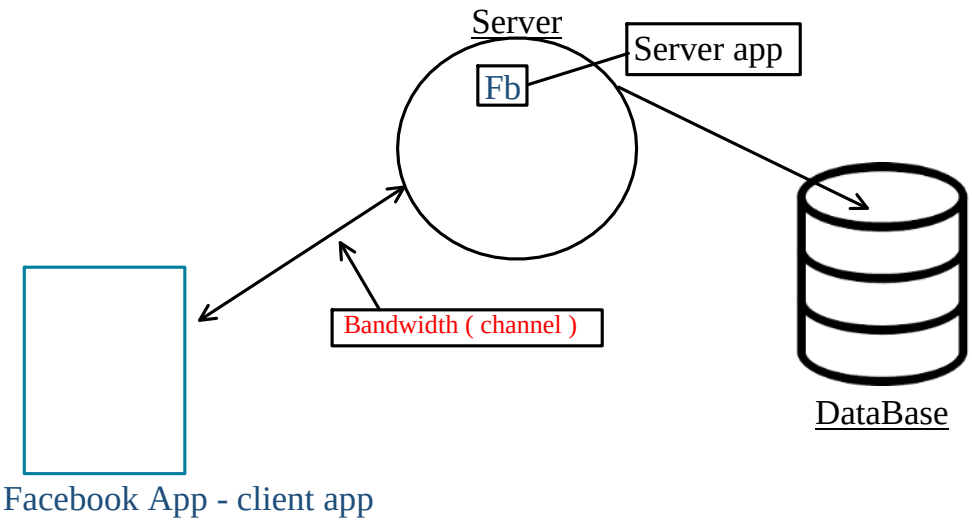
Any Application which is opened through a browser is known as WEB APPLICATION .

Examples : Facebook , wiki , youtube , gmail , amazon , SkillRary….. Etc ..

Server : it is nothing but a super computer where in all the applications are installed and can be accessed by anyone . [High Configuration]



3. CLIENT / SERVER APPLICATION :



In Client Server application, unlike Standalone Application, part of application is installed on to the client system and the remaining part is installed on to the server machine.

Example : Facebook , WhatsApp , Instagram , YouTube , Wiki , OLX , Flipkart .etc ….

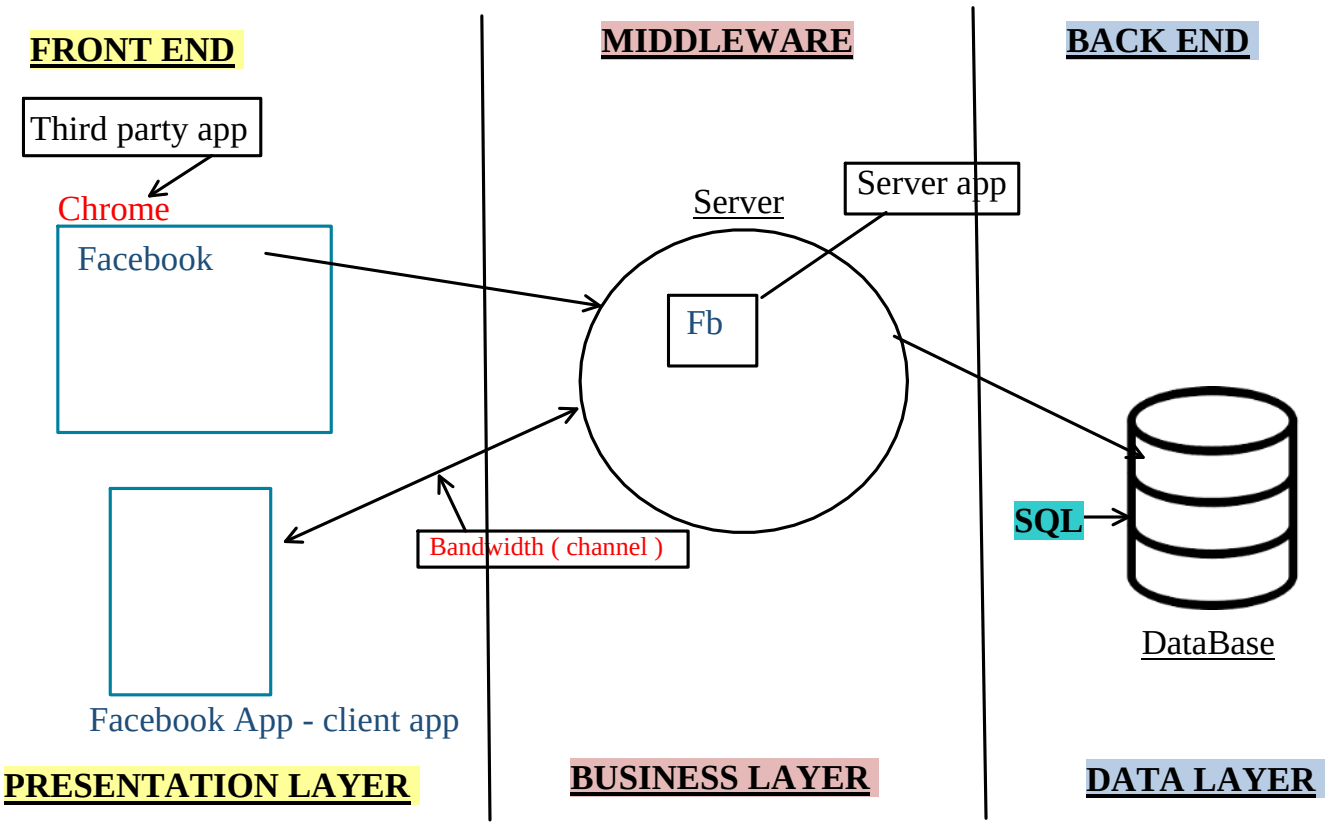
Advantages:

- Easy to access and faster in access if the bandwidth is more
- Data security from data hacking and virus
- Data sharing is possible
- Maintenance is not so tough
- Multiple users can access the application.

Disadvantages:

- Installation is still required at client's place
- End users resources are utilized low
- If the server goes down no one can access the application

Example :

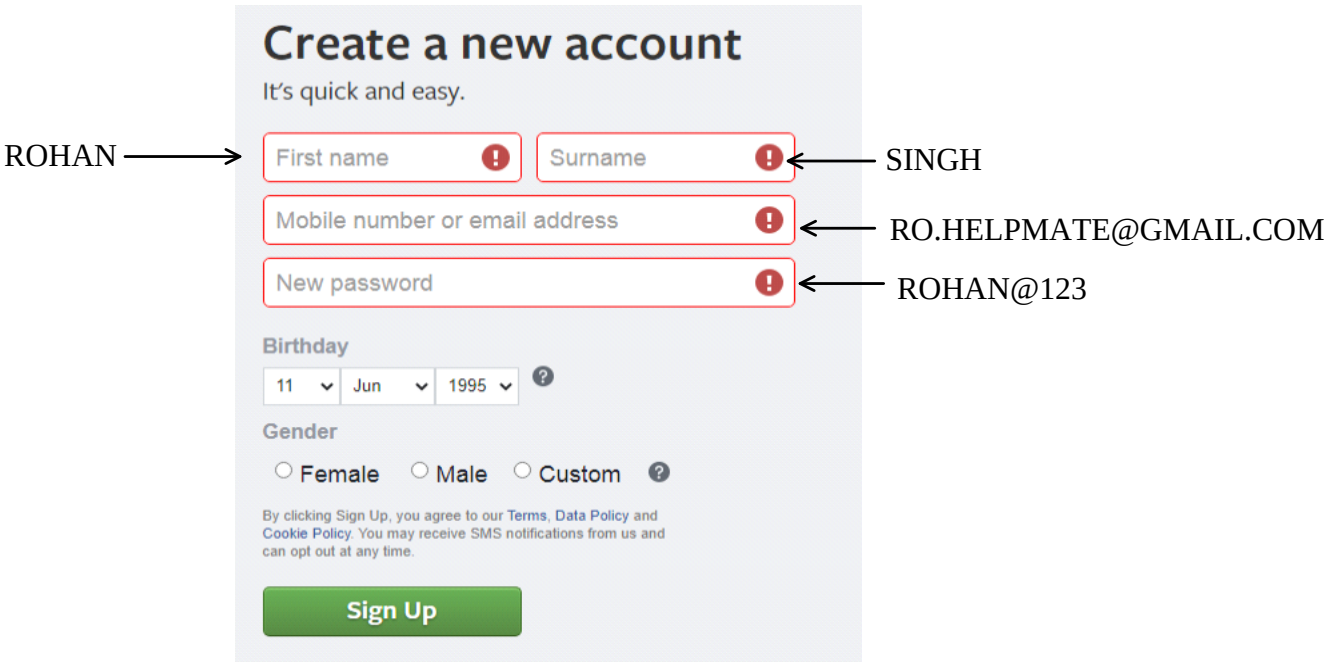


Basics of Database

Friday, June 12, 2020 9:38 AM

What is DATA ?

DATA is a raw fact which describes the attributes of an entity .

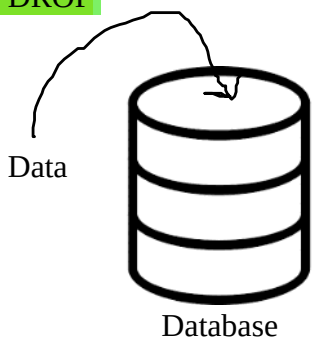


Database :

"It is a place or a medium in which we store the data in a systematic and organized manner " .

The basic operations that can be performed on a Database are :

i. CREATE / INSERT
ii. READ / RETRIEVE
iii. UPDATE / MODIFY
iv. DELETE / DROP



These operations are referred as **"CRUD"** Operations .
Examples : contacts , Facebook db , oracle db , drop box , google .

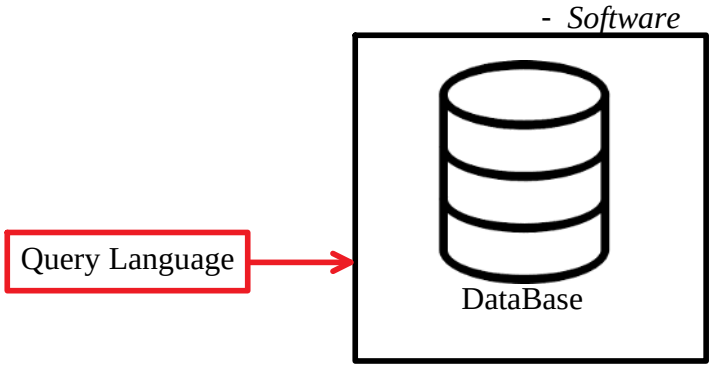
Database Management System (DBMS) :

" DBMS is a software which is used to maintain and manage the database " .

- The two important features provided by DBMS are
  - ◇ Security.
  - ◇ Authorization.

- Software

- ◇ Security.
- ◇ Authorization.

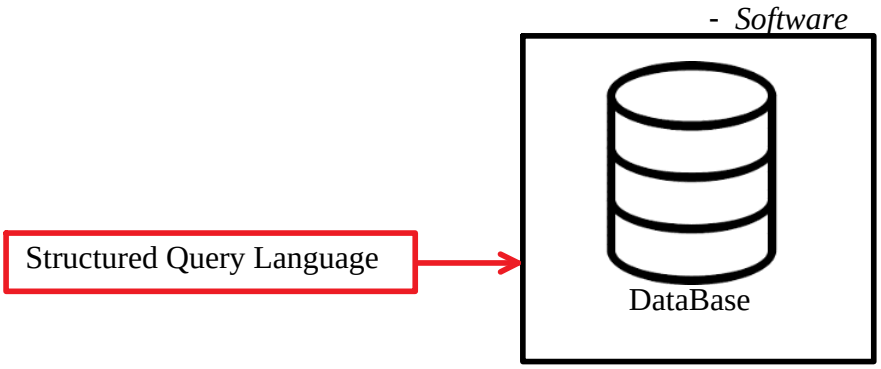


Database Management System (DBMS)

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

Relational Database Management System (RDBMS) :

" RDBMS is a type of DBMS Software which stores the data in the form of Tables " (rows and column) .



Relational Database Management System (RDBMS)

- ➢ We use SQL to communicate or interact with RDBMS .

Example : Oracle RDBMS , IBM DB2 , Microsoft Azure etc. …

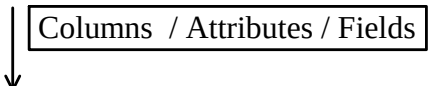
Relational Model :

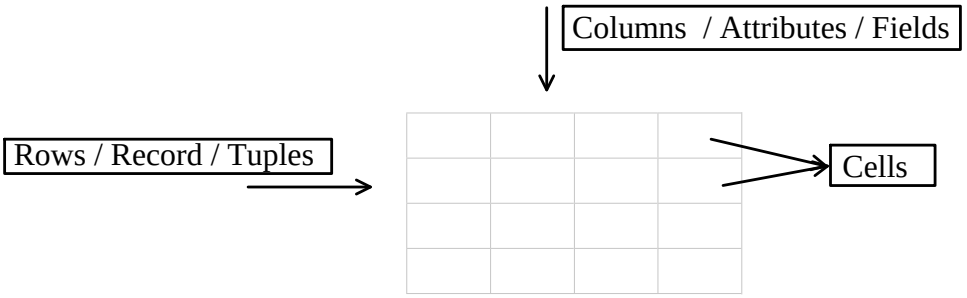
Relational Model was designed by "**Edgar F CODD**"

- ➢ The data must be stored in the form of rows and columns (table) .

- ➢ Any DBMS that follows Relational Model becomes RDBMS .
- ➢ Any DBMS which follows rules of EF CODD becomes RDBMS .

TABLE :





Example :

Students

SID	SNAME	BRANCH	PER
1	DINGA	ECE	60
2	DINGI	CSE	90
3	MANGA	ME	53

What is a Table ?

" Table is a logical organization of data which consist of rows and columns ".

Columns :

- Columns are also known as Attributes or Fields
- A column is used to represent property of all the entities .

SNAME
DINGA
DINGI
MANGA

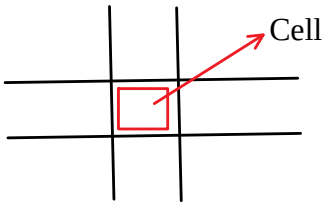
Row :

- Row is also known as Record or Tuples
- A row is used to represent properties of an individual entity .

3	MANGA	ME	53

Cell :

- Cell is the smallest unit of the table in which we store the data .
- The intersection of rows and columns generate cells .



DAY 3

Monday, 15 June 2020 9:29 AM

RULES OF E.F CODD :

- The data entered into a cell must be a single valued data (ATOMIC).

Example : customer

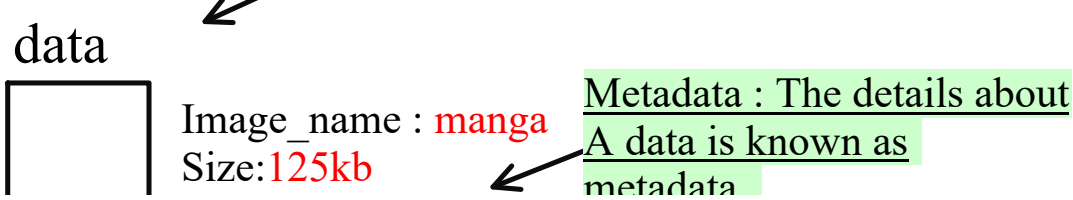
CID	CNAME	PHONE_NO
1	DINGA	1001
2	DINGI	2001 , 2002
3	MANGA	3001

CID	CNAME	PHONE_NO	ALTERNATE_NO
1	DINGA	1001	
2	DINGI	2001	2002
3	MANGA	3001	

- In RDBMS we store everything in the form of TABLES , Including METADATA .

Example :

PHOTO	CID	CNAME	PHONE_NO
	1	DINGA	1001
	2	DINGI	2002
	3	MANGA	3001



Format :jpeg
Resolution:400x600

Meta table- autogenerated

Image_name	size	format	resolution
manga	125	jpeg	400x600

- According to EF CODD we can store the data in Multiple Tables . If necessary we can establish a connection between The tables using Key Attributes .

- The data entering the table can be verified in 2 steps
  - By assigning Datatypes .
  - By assigning Constraints.

  Datatypes are mandatory whereas constraints are Optional .

Note : SQL is *not a case sensitive language .*

DATATYPES :

It is used to specify or determine the type / kind of data that Will be stored in a Particular memory location .

Data types in SQL

1. CHAR
2. VARCHAR / VARCHAR2
3. NUMBER
4. DATE
5. LARGE OBJECTS
   i. CHARACTER LARGE OBJECT
   ii. BINARY LARGE OBJECT

1. **CHAR** : 'A-Z' , 'a-z' , '0-9' , Special characters (!, $, @ etc.)

- Characters must always be enclosed within single quotes ' '.
- Characters are case sensitive

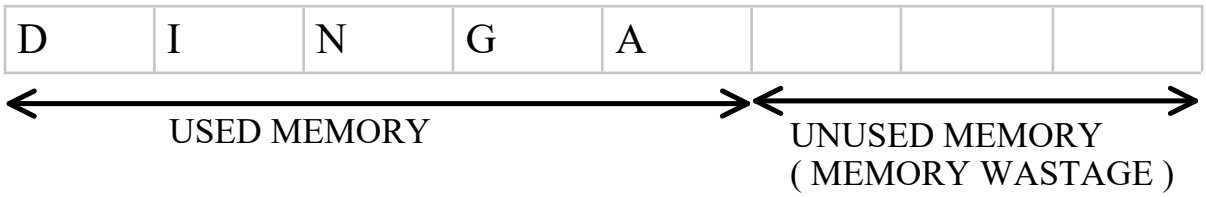
- Characters are case sensitive .
- When ever we use character datatype we must mention Size .

SYNTAX: CHAR (SIZE)

SIZE : It determines number of characters that can be stored .
The maximum number of characters that can be stored is 2000 ch.

- It follows fixed length memory allocation .

Example : Char (8)



2. VARCHAR : 'A-Z' , 'a-z' , '0-9' , Special characters (!, $, @ etc.)

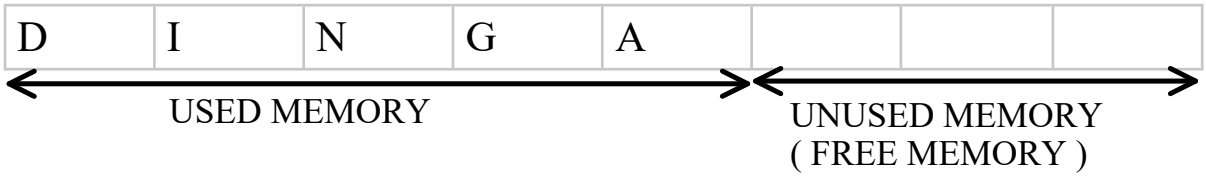
- Characters must always be enclosed within single quotes ' '.
- Characters are case sensitive .
- When ever we use character datatype we must mention Size .

SYNTAX: VARCHAR (SIZE) / VARCHAR2(SIZE)

SIZE : It determines number of characters that can be stored .
The maximum number of characters that can be stored is 2000 ch.
In Varchar2 we can store 4000ch .

- It follows Variable length memory allocation .

Example : varchar (8)



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

SYNTAX: NUMBER (Precision [, Scale])

SYNTAX: NUMBER (Precision [, Scale])

Precision : It is used to determine number of digits to store integer Values .

- The range of Precision is 1 to 38 .

Example :	Number (4)	+/- 9999
Example :	Number (6)	+/- 999999
Example :	Number (2 , 0)	+/- 99

Scale : It is used to determine number if digits to store decimal Value within the precision .

- The range of scale is -84 to 127 .
- Scale is not mandatory , The default value of scale is 0 .

Example :	Number (4 , 2)	+/- 99.99
Example :	Number (7 , 2)	+/- 99999.99
Example :	Number (3 , 3)	+/- .999
Example :	Number (2 , 0)	+/- 99
Example :	Number (3 , 5)	+/- .00999
Example :	Number (4 , 8)	+/- .00009999

Number(5)

72827.5 - 72828 can be stored by rounding off

4. **DATE :** It is used to store date in a specific format .

Format : 'DD-MON-YY'. OR 'DD-MON-YYYY'
Example : '15-JUN-20' , '15-AUG-1947'

SYNTAX: DATE

5. **LARGE OBJECTS**

i. **Character large object :** it is used to store large amount Of characters up to 4gb of size .

ii. **Binary Large object :** it is used to store binary values of Images , mp3 , mp4 , documents etc. up to 4gb of Size .

Example : STUDENT

COL_NAME	**SID**	**SNAME**	**PER**	**DOB**
DATATYPES	Char(4)	Varchar(8)	Number(4,2)	Date
	QSP1	DINGA	72.11	14-JUN-96
	QSP2	DINDI	80.23	12-AUG-97
	QSP3	MANGA	99.2	01-JAN-97

Assignment :

1. DIFFERENTIATE BETWEEN CHAR AND VARCHAR .

MAIL_ID : ro.helpmate@gmail.com

Subject : Assignment DAY3

Name :
Phone NO :
Batch Code : QCDM32

DAY 4

Tuesday, 16 June 2020 9:35 AM

CONSTRAINTS :

"It is a rule given to a column to validate the data " .

Types of constraints :

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

UNIQUE : "it is a constraint which is assigned to a column which Should not accept duplicated or repeated values ".

NOT NULL : "it is a constraint which is assigned to a column which Should not accept Null ".

CHECK :"It is a constraint that provides extra validation with a condition , IF the condition is satisfied then the value is accepted else rejected ".

Example :	CHECK (SAL > 0)
	CHECK (length(Phone_No) = 10)
	CHECK (age > 18)
	CHECK (DOB < SYSDATE)

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 Null
- PK cannot accept Duplicate or Repeated Values
- PK is always a combination of Unique & Not Null Constraint.
- It is not mandatory but highly recommended .

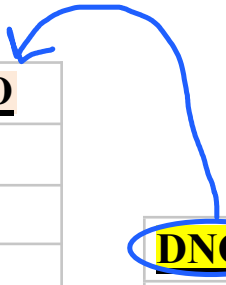
FOREIGN KEY :

"It is a constraint which is used to establish a connection Between the tables " .

Emp

EID	ENAME	SAL	DNO
1	A	100	20
2	B	200	Null
3	C	400	10

Child Table



Dept

DNO	DNAME	LOC
10	D1	L1
20	D2	L2

Parent Table

Characteristics of Foreign Key

- We can have Multiple FK in a table .
- FK can accept Null
- FK can accept Duplicate or Repeated Values
- FK is not a combination of Unique & Not Null Constraint.
- FK is present in child table but actually belongs to Parent Table.
- FK is also known as *Referential Integrity Constraint* .
- For an attribute to become Foreign key it must be a Primary Key it its table .

Example : STUDENT

PRIMARY KEY	Primary Key			
CHECK			Check (per > 0)	
NOT NULL	Not null	Not null		Not null
UNIQUE	Unique			
COL_NAME	**SID**	**SNAME**	**PER**	**DOB**
DATATYPES	Char(4)	Varchar(8)	Number(4,2)	Date
	QSP1	DINGA	72.11	14-JUN-96
	QSP2	DINDI	Null	12-AUG-97
	QSP3	DINGA	99.2	01-JAN-97

Example :

Emp

EID	ENAME	SAL	DNO	CONTACT
1	A	100	20	100
2	B	200	Null	101
3	C	400	10	100

Child Table

Customer

CID	CNAME	Contact
111	X	100
222	Z	101

Parent Table

Dept

DNO	DNAME	LOC
10	D1	L1
20	D2	L2

Parent Table

NOTE :

NULL

"It is keyword which is used to describe *Nothing* or *Empty* cell".

Characteristics of null :

- It doesn't represent 0 (zero) or space ' ' .
- It doesn't occupy any memory .
- In RDBMS we cannot equate NULL .
- Any Arithmetic Operations performed on a Null will always result

  In null itself .

  Ex: Null + 2000 = Null .

  Null * 2 = Null .

ASSIGNMENT :

- Differentiate between Primary Key and Foreign Key

- Differentiate between Primary Key and Foreign Key .

PRIMARY KEY	FOREIGN KEY

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

ORACLE standards .

SOFTWARE	Oracle 10g	SQL*Plus	goo.gl/6UTQvc

DOWNLOAD
EXTRACT
INSTALL

Password : TIGER

ro.helpmate@gmail.com

DAY 5

Wednesday, 17 June 2020 9:36 AM

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 :

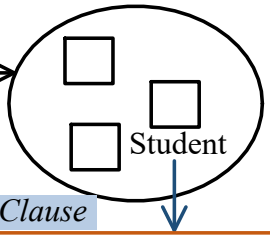
```
SELECT * / [DISTINCT] Column_Name / Expression [ALIAS]
FROM Table_Name ;
```

ORDER OF EXECUTION

1. FROM
2. SELECT

Example : Write a query to display name and per of all the students .

Database



2- SELECT SNAME , PER
1- 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	PER
A	60
B	75
C	50
D	80
C	75
E	95

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 .

Questions :

1. WAQTD name and branch of all the students .

   SELECT SNAME , BRANCH
   FROM STUDENT ;

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

   SELECT SNAME , BRANCH , PERCENTAGE
   FROM STUDENT ;

   Invalid identifier

   SELECT SNAME , BRANCH , PER
   FROM STUDENT ;

3. WAQTD details of all the students from students table .

   SELECT *
   FROM STUDENTS ;

   The table doesn't exist

   SELECT *
   FROM STUDENT ;

4. WAQTD name and id's of all the students .

   SELECT SNAME , SID
   FROM STUDENT ;

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

   SELECT SNAME , SID , PER , BRANCH
   FROM STUDENT ;

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

DEPT :

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

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 :

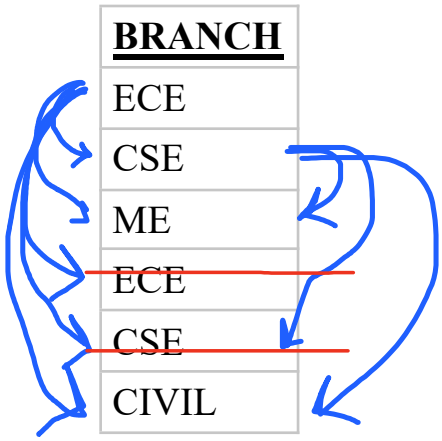
" Distinct clause is used to remove the duplicate records present in the Result table " .

- 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 .

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 DISTINCT BRANCH FROM STUDENT ;



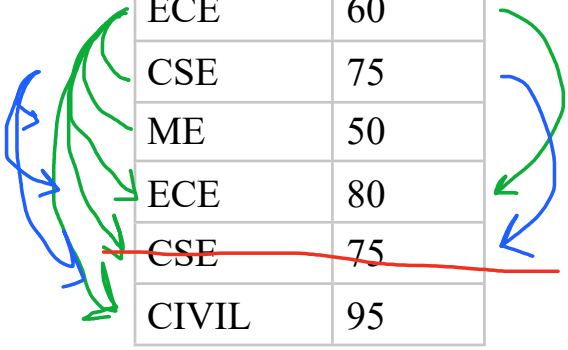
BRANCH
ECE
CSE
ME
CIVIL

- SELECT DISTINCT SNAME FROM STUDENT ;

SNAME
A
B
C
D
E

- SELECT DISTICT BRANCH , PER FROM STUDENT ;

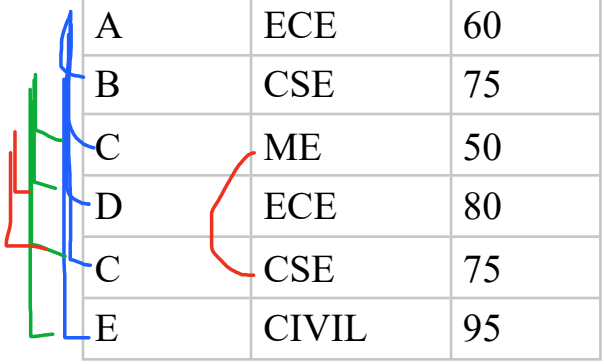
BRANCH	PER
ECE	60
CSE	75
ME	50
ECE	80
CSE	75
CIVIL	95



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

- SELECT SNAME , BRANCH , PER FROM STUDENT ;

SNAME	BRANCH	PER
A	ECE	60
B	CSE	75
C	ME	50
D	ECE	80
C	CSE	75
E	CIVIL	95



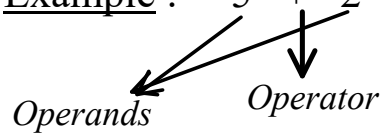
DAY 6

Thursday, 18 June 2020 9:38 AM

EXPRESSION :

Any statement which gives a result is known as expression .
Expression is always a combination of Operator and Operands .

Example : 5 + 2



Operands can be of two types :

- Column_Name → SAL * 12
- Literal

Literals can be of 3 types :

1. *Number Literal*
2. *Character Literal*
3. *Date Literal*

NOTE : Character and Date Literals are case sensitive and must be Enclosed within single quotes .

Example : NAME : 'Rohan Singh'
PHONE_NO : 9876543210
DOB : '14-MAY-1995'

Example : EMP

EID	ENAME	SAL
1	SMITH	200
2	ADAMS	100
3	MILLER	200

- WAQTD name of the employee along with their salary

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

- WAQTD name of the employee along with their Annual salary .

```
SELECT ENAME , SAL * 12
FROM EMP ;
```

- WAQTD name of the employee along with their Half term salary .

```
SELECT ENAME , SAL * 6
FROM EMP ;
```

- WAQTD all the details of employee along with their Half term salary.

**Asterisk has to be used alone .*

```
SELECT  * , SAL*6
FROM EMP ;
```

```
SELECT  EID , ENAME , SAL , SAL*6
FROM EMP ;
```

```
SELECT EMP.* , SAL*6
FROM EMP ;
```

- WAQTD name of the employee along with 10% hike in the salary .

How to find percentage :

SAL + SAL * a / 100

SAL * 1.a

```
SELECT ENAME , SAL+SAL*10/100
FROM EMP ;
```

```
SELECT ENAME , SAL*1.1
FROM EMP ;
```

- WAQTD name of the employee along with 10% deduction in the salary .

```
SELECT ENAME , SAL - SAL * 10 /100
```

SELECT ENAME , SAL - SAL * 10 /100
FROM EMP ;

SELECT ENAME , SAL * 0.9
FROM EMP ;

- WAQTD name of the employee along with 32% deduction in the salary .

SELECT ENAME , SAL - SAL * 32 /100
FROM EMP ;

ENAME	SAL-SAL*32/100
SMITH	136
ADAMS	68
MILLER	136

ALIAS :

"It is an Alternate name given to a column or an expression in the Result table ".

- Alias names must always be a single string which is Separated by underscore or enclosed within double quotes "

Example :	Annual_Salary
	"Annual Salary"

- We can assign alias name with or without using 'AS' keyword .

Example : EMP

EID	ENAME	SAL
1	SMITH	200
2	ADAMS	100
3	MILLER	200

- WAQTD name and annual salary for all the employees .

SELECT ENAME , SAL*12
FROM EMP ;

FROM EMP ;

ENAME	SAL*12
SMITH	2,400
ADAMS	1,200
MILLER	2,400

SELECT ENAME , SAL*12 AS Annual_Salary
FROM EMP ;

ENAME	Annual Salary
SMITH	2,400
ADAMS	1,200
MILLER	2,400

SELECT ENAME , SAL*12 Annual_Salary
FROM EMP ;

- SELECT EID A , ENAME B , SAL C
FROM EMP ;

A	B	C
1	SMITH	200
2	ADAMS	100
3	MILLER	200

- WAQTD name of the employee along with 32% deduction in the salary .

SELECT ENAME , SAL - SAL * 32 /100 "Sal After Deduction"
FROM EMP ;

ENAME	Sal After Deduction
SMITH	136
ADAMS	68

MILLER	136

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> ;
```

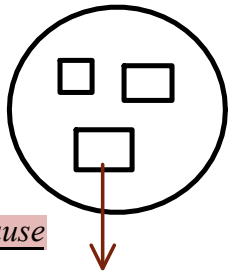
ORDER OF EXECUTION

1. FROM
2. WHERE
3. SELECT

WHERE Clause :

"Where clause is used to filter the records "

Example :



3- SELECT ENAME
1- FROM EMP
2- WHERE DNO = 20 ;

Output of FROM Clause

EMP

EID	ENAME	SAL	DNO
1	SMITH	100	10
2	ALLEN	250	20
3	BLAKE	300	30
4	MILLER	400	10
5	JONES	250	20

Filter Condition
DNO = 20

1	SMITH	100	10
2	ALLEN	250	20
3	BLAKE	300	30
4	MILLER	400	10
5	JONES	250	20

Output of WHERE Clause

EID	ENAME	SAL	DNO
2	ALLEN	250	20
5	JONES	250	20

Output of SELECT Clause

ENAME
ALLEN
JONES

NOTE :

1. WHERE Clause executes after From clause
2. WHERE clause executes row by row .
3. We can write filter condition in where clause , we can also write Multiple filter condition in where clause with the help of Logical Operators .

Questions :

- WAQTD names of the employees if they are working in dept 20 .

```
SELECT ENAME
FROM EMP
WHERE DNO = 20 ;
```

- WAQTD names of the employees earning more than 250 .

```
SELECT ENAME
FROM EMP
WHERE SAL > 250 ;
```

- WAQTD name and salary of the employees earning exactly 250 rupees.

```
SELECT ENAME
FROM EMP
WHERE SAL = 250 ;
```

- WAQTD all the details of the employee who's name is 'SMITH' .

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

- WAQTD all the details of the employee was hired on '11-DEC-1982' .

```
SELECT *
FROM EMP
WHERE HIREDATE ='11-DEC-1982' ;
```

ASSIGNMENT ON EXPRESSION & ALIAS

1.WAQTD NAME OF THE EMPLOYEE ALONG WITH THEIR ANNUAL SALARY.

```
Select ENAME , SAL*12 annual_salary
From emp  ;
```

2.WAQTD ENAME AND JOB FOR ALL THE EMPLOYEE WITH THEIR HALF TERM SALARY.

```
Select ENAME , JOB , SAL*6 half_term_salary
From emp  ;
```

3.WAQTD ALL THE DETAILS OF THE EMPLOYEES ALONG WITH AN ANNUALBONUS OF 2000.

```
Select EMP.* , SAL*12+2000 Bonus
From emp  ;
```

From emp ;

4.WAQTD NAME SALARY AND SALARY WITH A HIKE OF 10%.

*Select ENAME , SAL+ SAL*10/100 hike*
From emp ;

5.WAQTD NAME AND SALARY WITH DEDUCTION OF 25%.

*Select ENAME , SAL- SAL*25/100 deduction*
From emp ;

6.WAQTD NAME AND SALARY WITH MONTHLY HIKE OF 50.

Select ENAME , SAL+ 50 hike
From emp ;

7.WAQTD NAME AND ANNUAL SALARY WITH DEDUCTION OF 10%.

*Select ENAME , SAL*12- SAL*12*10/100 deduction*
From emp ;

8.WAQTD TOTAL SALARY GIVEN TO EACH EMPLOYEE (SAL+COMM).

Select SAL+COMM total_salary
From emp ;

9.WAQTD DETAILS OF ALL THE EMPLOYEES ALONG WITH ANNUAL SALARY.

Select EMP. , SAL*12 annual_salary*
From emp ;

10.WAQTD NAME AND DESIGNATION ALONG WITH 100 PENALTY IN SALARY.

Select ENAME , JOB , SAL-100 penalty
From emp ;

ASSIGNMENT ON WHERE Clause .

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

*Select SAL*12*
From emp
Where ENAME ='SMITH' ;

2.WAQTD NAME OF THE EMPLOYEES WORKING AS CLERK

2.WAQTD NAME OF THE EMPLOYEES WORKING AS CLERK

```
Select ENAME
From emp
Where  JOB ='CLERK' ;
```

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

```
Select SAL
From emp
Where  JOB ='SALESMAN'  ;
```

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

```
Select *
From emp
Where  SAL > 2000  ;
```

5.WAQTD DETAILS OF THE EMP WHOS NAME IS JONES

```
Select *
From emp
Where  ENAME ='JONES'   ;
```

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

```
Select *
From emp
Where  HIREDATE > '01-JAN-81' ;
```

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

```
Select ENAME , SAL , SAL * 12
From emp
Where  SAL *12 > 12000 ;
```

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

```
Select EMPNO
From emp
Where  DEPTNO = 30 ;
```

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

```
Select ename , hiredate
```

From emp
Where hiredate < '01-JAN-1981' ;

10.WAQTD DETAILS OF THE EMPLOYEES WORKING AS MANAGER

*Select **
From emp
Where JOB ='MANAGER';

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

Select ENAME , SAL
From emp
Where comm = 1400 ;

12.WAQTD DETAILS OF EMPLOYEES HAVING COMMISSION MORE THAN SALARY

*Select **
From emp
Where comm > sal ;

13.WAQTD EMPNO OF EMPLOYEES HIRED BEFORE THE YEAR 87

Select empno , hiredate
From emp
Where hiredate < '01-JAN-1987' ;

14.WAQTD DETAILS OF EMPLOYEES WORKING AS AN N ANALYST

*Select **
From emp
Where JOB ='ANALYST' ;

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

*Select **
From emp
Where SAL > 2000 ;

DAY 7

Friday, 19 June 2020 9:26 AM

OPERATORS IN SQL

1. ARITHEMATIC OPERATORS :- (+ , - , * , /)
2. CONCATINATION 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

CONCATWNATION OPERATOR:

"It is used to join the given strings "
The symbol used for concatenation is ||

Example :

EID	ENAME	SAL	DNO
1	SMITH	100	10
2	ALLEN	250	20
3	BLAKE	300	30
4	MILLER	400	10
5	JONES	250	20

- SELECT ENAME
  FROM EMP
  WHERE DEPTNO = 20 ;

ENAME
ALLEN
JONES

- SELECT 'Hello '|| ENAME
  FROM EMP
  WHERE DEPTNO = 20 ;

ENAME
Hello ALLEN
Hello JONES

- SELECT ENAME || SAL
  FROM EMP
  WHERE DEPTNO = 20 ;

ENAME||SAL
ALLEN250
JONES250

- SELECT ENAME || ' ' || SAL
  FROM EMP
  WHERE DEPTNO = 20 ;

ENAME||' ' ||SAL
ALLEN 250
JONES 250

LOGICAL OPERATORS :

1. AND
2. OR
3. NOT

- Logical Op such as AND , OR must be used between The conditions .
- AND Op is used whenever we have satisfy all the conditions.
- OR Op is used when we've to satisfy any one of the conditions.
- NOT Op is a unary operator [*it can have only 1 Operand*]
  It is used to negate the Output . [*inverse*]

1. WAQTD names of the employees working in dept 10

   SELECT ENAME
   FROM EMP
   WHERE DEPTNO = 10 ;

2. WAQTD named of the employees working in dept 10 and earning More than 150 rupees .

   SELECT ENAME

FROM EMP
WHERE DEPTNO = 10 AND SAL > 150 ;

3. WAQTD name and sal and deptno of the employee of the emp Is working as CLERK and earning more than 2300 rupees .

SELECT ENAME , SAL , DEPTNO
FROM EMP
WHERE JOB = 'CLERK' AND SAL > 2300 ;

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

4. WAQTD name and hiredate of the employee working as a 'CLERK' Or as a MANAGER .

SELECT ENAME , HIREDATE
FROM EMP
WHERE JOB = 'CLERK' OR JOB='MANAGER' ;

5. WAQTD all the details of the employees working in dept 10 or 20 or as a salesman .

*SELECT **
FROM EMP
WHERE DEPTNO = 10 OR DEPTNO = 20 OR JOB='SALESMAN';

6. WAQTD name and hiredate of the employees if the employees Are hired after 1981 and before 1987 .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE > '31-DEC-1981' AND
                HIREDATE < '01-JAN-1987'
```

7. WAQTD name sal and annual salary of all the employees who are Working in dept 20 and has annual salary greater than 10000 but less Than 20000 .

```
SELECT ENAME , SAL , SAL*12
FROM EMP
WHERE DEPTNO = 20 AND SAL*12 > 10000 AND SAL*
12<20000 ;
```

```
SELECT ENAME , SAL , SAL*12  A_S
FROM EMP
WHERE DEPTNO = 20 AND A_S > 10000 AND A_S <20000 ;
```

WRONG (*Alias names cannot be used before assigning)*

8. WAQTD name , sal and deptno of the employees earning 2000 In dept 20 or 30 .

```
SELECT ENAME , SAL , DEPTNO
FROM EMP
WHERE SAL = 2000 AND ( DEPTNO = 20 OR DEPTNO = 30 ) ;
```

EID	ENAME	SAL	DNO
1	SMITH	2000	30
2	ALLEN	2000	20
3	BLAKE	2000	10
4	MILLER	400	10
5	JONES	250	20

SAL = 2000	*AND*	*DEPTNO = 20*	*OR*	*DEPTNO = 30*
2000 = 2000	*And*	*30 = 20*	*Or*	*30 = 30*
T		*F*	*OR*	*T*
T		*T*		
TRUE				

SAL = 2000	*AND*	*DEPTNO = 20*	*OR*	*DEPTNO = 30*
2000 = 2000	*And*	*20 = 20*	*Or*	*20 = 30*
T		*T*	*OR*	*F*
T		*T*		
TRUE				

SAL = 2000	*AND*	*DEPTNO = 20*	*OR*	*DEPTNO = 30*
2000 = 2000	*And*	*10 = 20*	*Or*	*10 = 30*
T		*F*	*OR*	*F*
T		*F*		
FALSE				

9. WAQTD names of the employees working in dept 10
As a clerk or a manager .

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

10. WAQTD details of the employees working as 'SALESMAN' or
' ANALYST' in dept 10 or 30 .

```
SELECT *
FROM EMP
WHERE ( JOB ='SALESMAN' OR JOB='ANALYST' ) AND
 ( DEPTNO = 10 OR DEPTNO = 30 ) ;
```

11. WAQTD details of all the employees except the employee working as 'Salesman' .

```
SELECT *
FROM EMP
WHERE JOB != 'SALESMAN' ;

SELECT *
FROM EMP
WHERE NOT JOB ='SALESMAN' ;
```

12. WAQTD details of all the employees except the employees Working in dept 10 or 20 .

```
SELECT *
FROM EMP
WHERE DEPTNO != 10 OR DEPTNO !=20  ;

SELECT *
FROM EMP
WHERE NOT ( DEPTNO = 10 OR DEPTNO = 20 ) ;
```

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

Oracle 10 G. SQL*Plus : goo.gl/6UTQvc

While installing if you get any error
Snapshot the error to my account : ro_sql_helpmate (INSTA).

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 IN (10,20);
```

WHERE JOB = 'CLERK' AND DEPTNO IN (10,20);

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

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

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

```
SELECT ENAME
FROM EMP
WHERE DEPTNO IN (10, 20, 30, 40);
```

14. WAQTD DETAILS OF EMPLOYEES WITH EMPNO 7902, 7839

```
SELECT *
FROM EMP
WHERE EMPNO IN (7902, 7839);
```

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

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

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

```
SELECT NAME
FROM EMP
WHERE HIREDATE BETWEEN '01-JAN-82' AND '31-DEC-86' ;
```

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

```
SELECT *
FROM EMP
WHERE SAL BETWEEN 1250 AND 3000;
```

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

```
SELECT ENAME
FROM EMP
WHERE HIREDATE > '31-DEC-81' AND DEPTNO IN (10, 30);
```

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

```
SELECT ENAME, SAL *12 “ANNUAL SALARY”
FROM EMP
```

```
WHERE JOB IN ('MANAGER', 'CLERK') AND DEPTNO IN (10, 30);
```

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

```
SELECT EMP.*, SAL*12 "ANNUAL SALARY"
FROM EMP
WHERE SAL*12>15000 AND SAL BETWEEN 1000 AND 4000;
```

DAY 8

Monday, 22 June 2020 9:29 AM

SPECIAL OPERATORS

1. **IN :** It is a multi valued operator which can accept multiple Values at the RHS .
   - IN Operator returns true if any of the condition is satisfied At RHS .

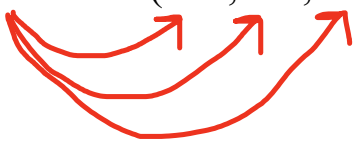
SYNTAX : Column_Name / Exp IN (v1 , v2 , … vn) ;

Example :

- WAQTD names of the employees working in dept 10 or 20 or 30.

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

```
SELECT ENAME
FROM EMP
WHERE DEPTNO IN ( 10 , 20 , 30 ) ;
```



DEPTNO	IN	(10 , 20 , 30)
30	IN	(10 , 20 , 30)
30	=	10 F
30	=	20 F
30	=	30 TRUE

- WAQTD details of the employees working as clerk or salesman .

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

WHERE JOB IN (CLERK , SALESMAN) ;

- WAQTD details of the employee whose name is WARD .

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

SELECT *
FROM EMP
WHERE ENAME IN 'WARD' ;
```

2. **NOT IN :** "It is similar to IN operator , instead of selecting the values It rejects the values ".

SYNTAX : Column_Name / Exp NOT IN (v1 , v2 ,… Vn);

Example :

- WAQTD all the details of the employees except the employees Who works as analyst or salesman .

```
SELECT *
FROM EMP
WHERE JOB NOT IN ('SALESMAN' ,'ANALYST' );
```

- WAQTD name , deptno and salary of all the employees except the Employees who work in dept 10 or 40

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

3. **BETWEEN :** "It is used whenever we have range of values ". [start value and end value] .

SYNTAX: Column_Name BETWEEN Lower_Range AND Higher_Range

- We cannot interchange the range .
- It works including the range specified .

Example :

- WAQTD name and salary of the employees if their range of salary Is 1000 to 3000 .

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

- WAQTD name and hiredate of the employees if the employees Were hired during 1982 .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE BETWEEN '01-JAN-82' AND '31-DEC-82' ;
```

4. **NOT BETWEEN :** "It is Opposite of between operator "

SYNTAX: Column_Name NOT BETWEEN L_R AND H_R;

Example :

- WAQTD name and hiredate of all the employees except the employee
  Hired in 2020 .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE NOT BETWEEN '01-JAN-2020' AND 31-
DEC-2020 '
```

- WAQTD name and salary given to the employees who's salary Is not the range 1000 to 3000 .

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

5. **IS : "** IS operator is used to compare with Null .

SYNATX : Column_Name IS NULL ;

Example :
EMP

Ename	Sal	Comm
A	100	20
B	200	Null
C	Null	10

- WAQTD the name of the employee who doesn't get salary .

```
SELECT ENAME
FROM EMP
WHERE SAL = NULL ;     --No ( null cannot be equated ).

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

- WAQTD all the details of the employee who doesn't earn commission .

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

6. **IS NOT :**"IS NOT Operator is used to compare with NOT NULL ".

SYNTAX: Column_Name IS NOT NULL ;

Example :

- WAQTD name of the employee who get commission .

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

- WAQTD name of the employee who gets salary as well as commission .

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

- WAQTD the emp name who doesn't get salary but gets commission .

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

7. **LIKE :** "It is used to perform pattern matching ".

To achieve pattern matching we use Special Characters

1. Percentile (%)
2. Underscore (_)

SYNTAX : Column_Name LIKE 'pattern_to_match' ;

Example :

- WAQTD salary of smith .

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

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

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

Ename
ALLEN
BLAKE
CLARK
JONES
SMITH
MILLER
SCOTT
SAM

SMITH
SCOTT
SAM

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

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

JONES

- WAQTD name of the employee if the employee has 'S' in the name .

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

- WAQTD name of the employee of emp has character 'A' as the Second character in the name .

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

- WAQTD name of the employee if the emp has exactly 4 character Name .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '____' ;
```

- WAQTD name of the employee who has 6 character in the name And 3rd character is 'A' .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '__A___' ;
```

- DISPLAY NAME WHICH HAS 8 CHARACTERS IN WHICH 3rd character is A , 6th character is 'B' and last character is 'C' .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '__A__B_C' ;
```

8. NOT LIKE : "It is similar to like Operator instead of selecting the Values it rejects ".

SYNTAX: Column_Name NOT LIKE 'pattern_to_match' ;

Example :

- WATQD details of all the employees except the employee who's name

name
Starts with ' Z ' .

```
SELECT *
FROM EMP
WHERE ENAME NOT LIKE 'Z%' ;
```

- WAQTD name of the employee who was hired on December .

```
SELECT ENAME
FROM EMP
WHERE HIREDATE LIKE '%DEC%' ;
```

- WAQTD name and salary of the employees where the last two digits Of the employee's salary is 50 .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL LIKE '%50' ;
```

- WAQTD name and hiredate of the employee if the employee's name Starts with 'A' and was hired in 1982 .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE ENAME LIKE 'A%' AND HIREDATE LIKE '%1982' ;
```

ASSIGNMENT ON SPECIAL 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

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
21) LIST THE EMPLOYEES WHOSE SALARY NOT IN THE RANGE OF 1000 TO 2000 AND WORKING IN DEPT 10,20 OR 30 EXCEPT ALL SALESMEN
22) LIST THE DEPARTMENT NAMES WHICH ARE HAVING LETTER 'O' IN THEIR LOCATIONS AS WELL AS THEIR DEPARTMENT NAMES
23) DISPLAY ALL THE EMPLOYEES WHOSE JOB HAS STRING 'MAN' IN IT.
24)LIST THE EMPLOYEES WHO ARE HIRED AFTER 82 AND BEFORE 87.
25)WAQTD ALL THE DETAILS OF EMPLOYEES HIRED IN NOVEMBER AND DECEMBER.
26)LIST ALL THE EMPLOYEE NAMES AND COMISSION FOR THOSE EMPLOYEES WHO EARN COMISSION MORE THAN THEIR SALARY
27)WAOTD NAME AND DESIGNATION FOR ALL THE

EMPLOYEES HAVING REPORTING MANAGERS AND ALSO THRIE NAMES STARTING WITH ‘S’

28)WAQTD NAME AND SALARY OF ALL THE EMPLOYEES IF THEIR ANNUAL SALARY ENDS WITH ‘0’ .

29)WAQTD NAME OF THE EMPLOYEE HAVING ATLEAST 2L’s IN HIS NAME .

30)WAQTD NAME OF THE EMPLOYEES WHOS NAME STARTS WITH A ‘VOWEL’

DAY 9

Tuesday, 23 June 2020 9:24 AM

FUNCTIONS

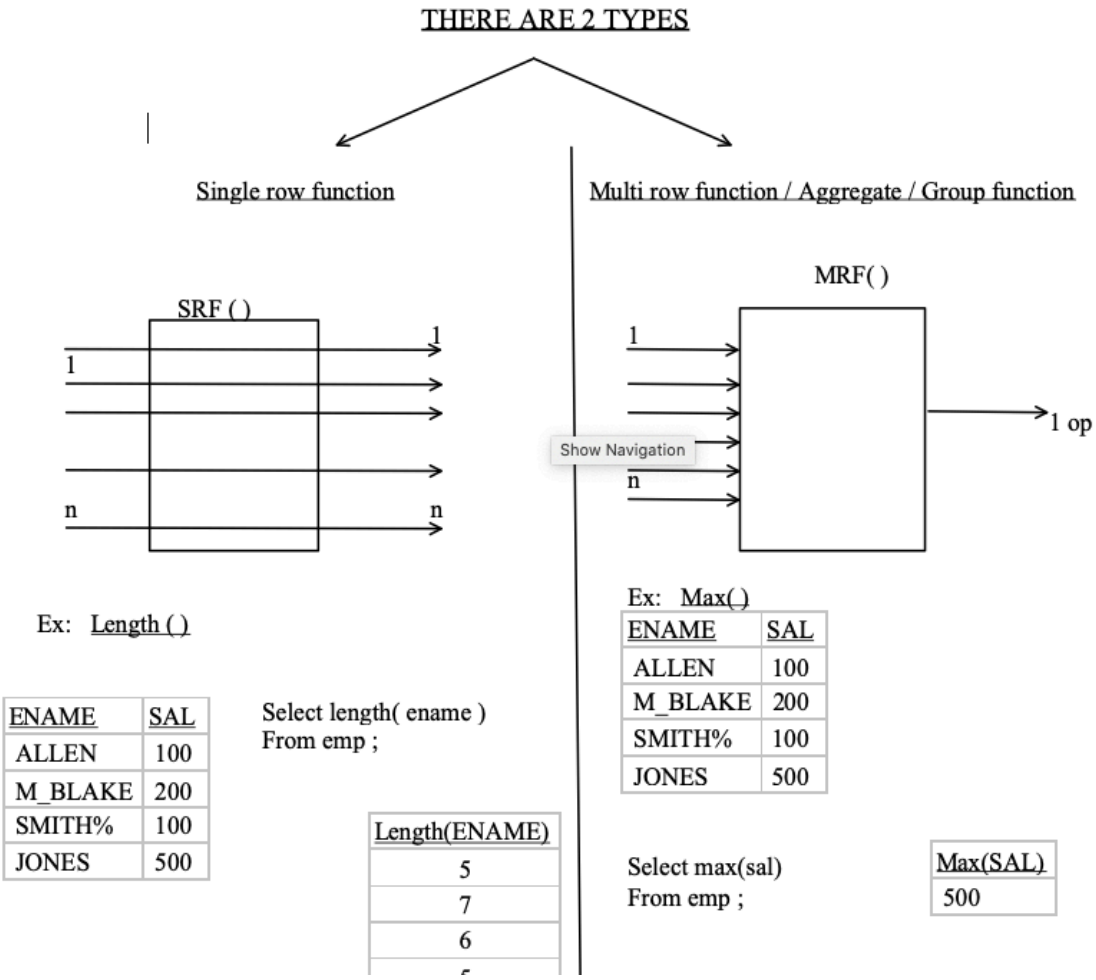
Are a block of code or list of instructions which are used to perform a specific task .

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.



ENAME	SAL
ALLEN	100
M_BLAKE	200
SMITH%	100
JONES	500

Length(ENAME)
5
7
6

ENAME	SAL
ALLEN	100
M_BLAKE	200
SMITH%	100
JONES	500

Max(SAL)
500

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

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( ENAME )
   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 HIREDATE > '31-DEC-1982' AND HIREDATE < '01-JAN-1985' AND DEPTNO IN ( 10 , 30 ) ;
   ```

   ```
   SELECT COUNT(*)
   FROM EMP
   ```

WHERE HIREDATE BETWEEN '01-JAN1983' AND '31-DEC-1984' 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 ;
```

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

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

ESCAPE CHARACTER

" It is used to remove the special behavior given to a Special character (% , _) and to treat it as a normal character "

ENAME
SMITH
ALLEN
DON_Z
ABC100%

- WAQTD the names that have an '_' present .

SELECT ENAME

ENAME
SMITH
ALLEN

```
FROM EMP
WHERE ENAME LIKE '%_%' ;
```

SC

DON_Z
ABC100%

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '%!_%'  ESCAPE '!' ;
```

NC

DON_Z

- WAQTD name which has '%' in it .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '%$%%' escape  '$' ;
```

ABC100%

SC
EC
NC

NOTE :

- Escape character must be defined .
- Escape character must be used before the special character which has to treated as a Normal Character .
- The recommended characters for Escape is : [! , / , \ , $] .

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)
```

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

DAY 10

Wednesday, 24 June 2020 9:39 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]

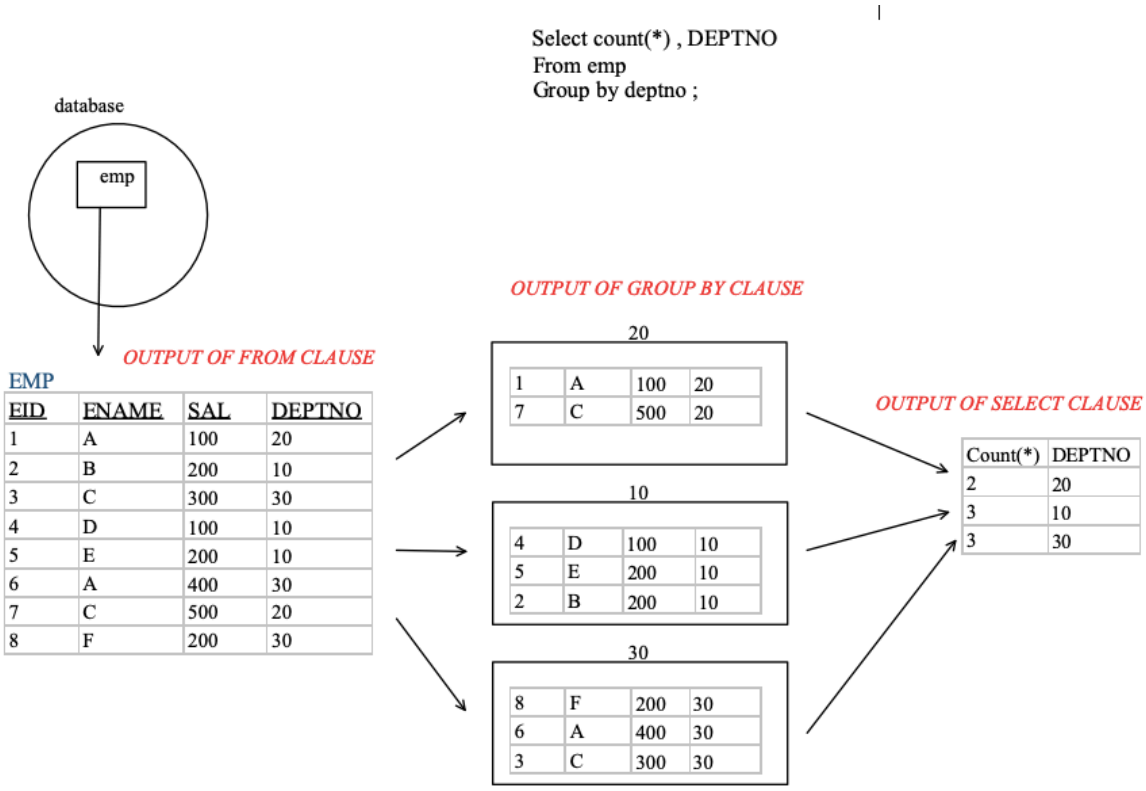
EMP

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 DEPTNO , 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(*)

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(*)
From emp
Where comm is not null
Group by deptno ;

Or

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:

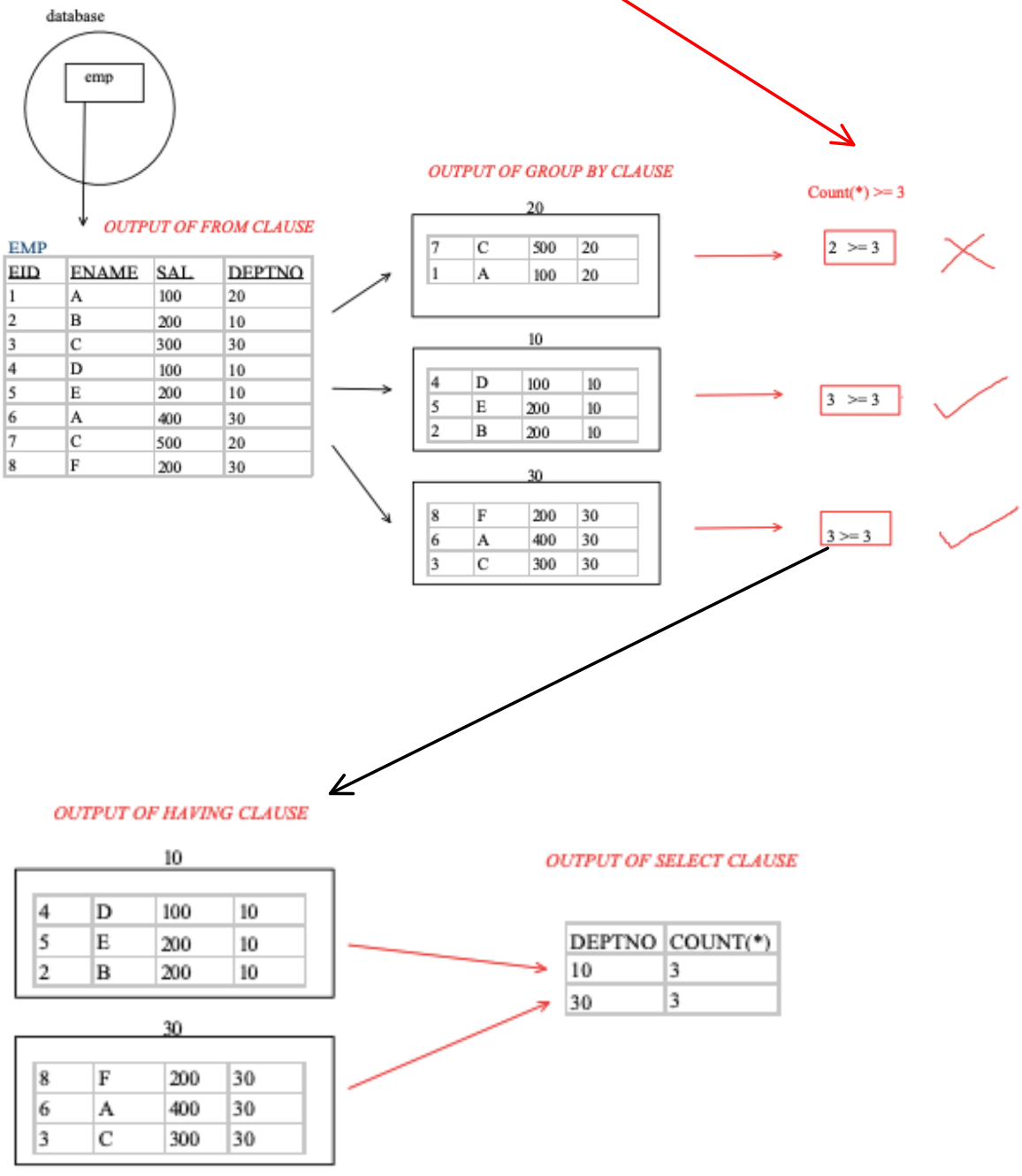
1-FROM
2-WHERE(if used) *[ROW-BY-ROW]*
3-GROUP BY(if used) *[ROW-BY-ROW]*
4-HAVING (if used) *[GROUP-BY-GROUP]*
5-SELECT *[GROUP-BY-GROUP]*

Example :

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

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(*)
```

```
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
```

```
Group by job
Having sum(sal) >3800 ;
```

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;
```

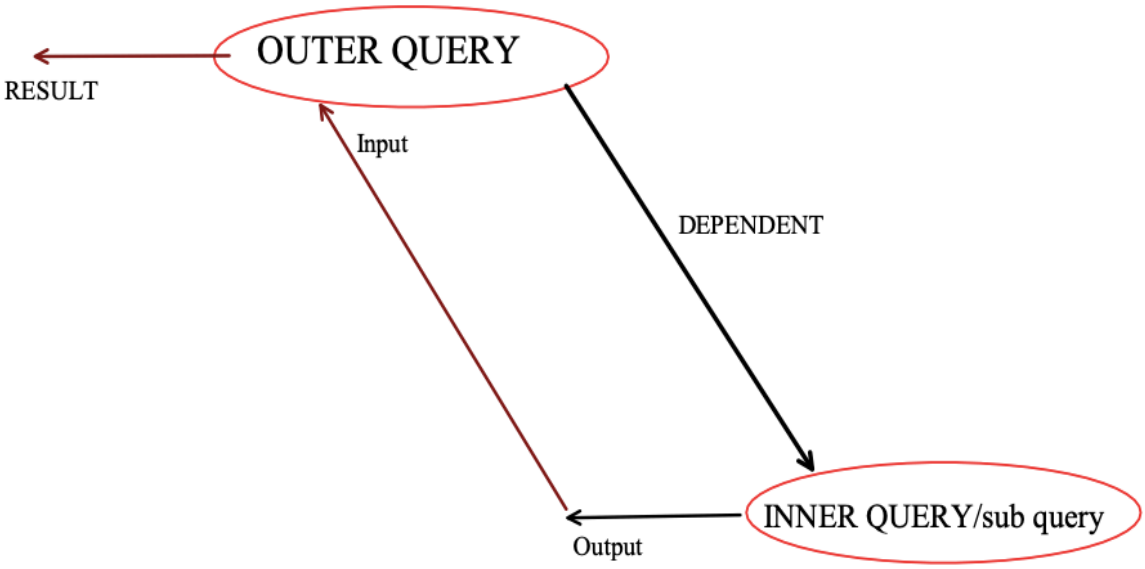
DAY 11

Thursday, 25 June 2020 9:27 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                1500
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' ) ;
```

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

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' ) ;
```

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

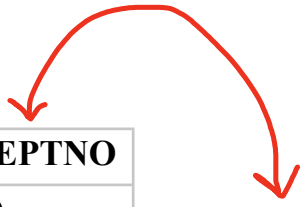
EMPLOYEES ARE HIRED BEFORE MILLER
10.WAQTD ENAME AND EMPNO OF THE EMPLOYEES IF EMPLOYEES ARE EARNING MORE THAN ALLEN
11.WAQTD ENAME AND SALARY OF ALL THE EMPLOYEES WHO ARE EARNING MORE THAN MILLER BUT LESS THAN ALLEN
12.WAQTD ALL THE DETAILS OF THE EMPLOYEES WORKING IN DEPT 20 AND WORKING IN THE SAME 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

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



DEPT

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

5	SMITH	2500	10

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' ) ;
```

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

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

SUB-QUERY

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' );
```

21.WAQTD DNAME OF THE EMPLOYEES WHOS NAME IS SMITH

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' );
```

DAY 12

Friday, 26 June 2020 9:44 AM

- WAQTD name of the employee working in New York .

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

- WAQTD names of the employees working in the same designation Jones .

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

- WAQTD names of the employees working in New York and in the same designation as jones .

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

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

TYPES OF SUB QUERY :

We have 2 types of sub queries .

1. Single Row Subquery
2. Multi Row Subquery

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

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

Example :

- WAQTD dname of Allen

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

20

WHERE ENAME ='ALLEN') ;

- WAQTD dname of Allen and Clark

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

20
30

1. Single Row Sub Query :

If a sub query returns exactly 1 record / 1 value
We call it as Single Row Sub Query .

- If a sub query returns single value then we can use both Normal Op as well as Special Op .

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

20

= or IN

2. Multi Row Sub Query :

If a sub query returns more than 1 record / 1 value
We call it as Multi Row Sub Query .

- If a sub query returns more than 1 value then we cannot use Normal Op we *must only use Special Op* .

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

20
30

IN

IN

- WAQTD name and salary of the employee earning more than Allen and Blake .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL  >   ( SELECT SAL
                 FROM EMP
                 WHERE ENAME IN ('ALLEN','BLAKE' ) );
```

We cannot use (> , < , >= , <=) we have to use Sub query Op (ALL & ANY)

Note : IN & NOT IN can be used as a substitute for = & != Operator only not for Relational Op (> , < , >= , <=)

Example :

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL  > ALL  ( SELECT SAL
                    FROM EMP
                    WHERE ENAME IN ('ALLEN','BLAKE' ) )
```

1000
2000

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

SAL
1000
2000

> ALL

1000

3000
1500
2500

1000 > 1000	False
1000 > 2000	False

2000> 1000	True
2000>2000	False

3000> 1000	True
3000 > 2000	True

ALL Operator :

All operator is a special operator which has to be used along with relational Operator which will compare the value present at the LHS with all the values of RHS

- *ALL op **returns true** if all the values at the RHS have satisfied the Condition .*

ANY Operator :

Any operator is a special operator which has to be used along with relational Operator which will compare the value present at the LHS with all the values of RHS

- *ANY op **returns true** if one of the values at the RHS have satisfied the Condition .*

Example :

- WAQTD name and salary of the employee earning more than Allen or Blake .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL    >  ANY ( SELECT SAL
                      FROM EMP
```

FROM EMP
WHERE ENAME IN ('ALLEN','BLAKE'))

SAL
1000
2000
3000
1500
2500

> ANY

1000
2000

1000 > 1000	False
1000 > 2000	False

2000 > 1000	True
2000 > 2000	False

3000> 1000	True
3000 > 2000	True

1500> 1000	True
1500 > 2000	False

- WAQTD name and salary of the employees earning more than The employees of dept 20 .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL  > ALL ( SELECT SAL
                   FROM EMP
                   WHERE DEPTNO = 20 ) ;
```

- WAQTD name and salary of the employees if the employees Were hired before a clerk .

SELECT ENAME , SAL

```
SELECT ENAME , SAL
FROM EMP
WHERE HIREDATE < ANY ( SELECT HIREDATE
                       FROM EMP
                       WHERE JOB ='CLERK' ) ;
```

- WAQTD name and hiredate of the employee if all the employees were hired after all the managers .

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE > ALL( SELECT HIREDATE
                      FROM EMP
                      WHERE JOB ='MANAGER' ) ;
```

ASSIGNMENT ON ALL & ANY

41.WAQTD NAME OF THE EMPLOYEES EARNING SALARY MORE THAN THE SALESMAN
42.WAQTD DETAILS OF THE EMPLOYEES HIRED AFTER ALL THE CLERKS
43.WAQTD NAME AND SALARY FOR ALL THE EMPLOYEES IF THEY ARE EARNING LESS THAN ATLEST A MANAGER
44.WAQTD NAME AND HIREDATE OF EMPLOYEES HIRED BEFORE ALL THE MANAGERS
45.WAQTD NAMES OF THE EMPLOYEES HIRED AFTER ALL THE MANAGERS AND EARNING SALARY MORE THAN ALL THE CLERKS
46.WAQTD DETAILS OF THE EMPLOYEES WORKING AS CLERK AND HIRED BEFORE ATLEST A SALESMAN
47.WAQTD DETAILS OF EMPLOYEES WORKING IN ACCOUNTING OR SALES DEPT
48.WAQTD DEPARTMENT NAMES OF THE EMPOYEES WITH NAME SMITH , KING AND MILLER
49.WAQTD DETAILS OF EMPLOYEES WORKING NEWYORK OR CHICAGO
50.WAQTD EMP NAMES IF EMPLOYEES ARE HIRED AFTER ALL THE EMPLOYEES OF DEPT 10

MAX & MIN

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

- WAQTD maximum salary given to an employee .

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

- WAQTD name of the employee getting maximum salary .

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

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

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

- WAQTD name and hiredate of the employee who was hired at the last .

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

2 days time .

All assignments on SUB QUERY must me Mailed

AND must be practiced .

Flow :

- What is sub query ?
- Show the execution ! Draw it .
- When ?
  - Case 1
  - Case 2
- Types
  - Single row
  - Multi row
    - ALL
    - ANY

ANSWERS ON CASE 1 & 2

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
```

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');

ANSWERS ON ALL & ANY :

51.WAQTD NAME OF THE EMPLOYEES EARNING SALARY MORE THAN
THE SALESMAN

SELECT ENAME
FROM EMP
WHERE SAL> ALL (SELECT SAL
FROM EMP
WHERE JOB ='SALESMAN') ;

52.WAQTD DETAILS OF THE EMPLOYEES HIRED AFTER ALL THE CLERKS

*SELECT **
FROM EMP
WHERE HIREDATE > ALL (SELECT HIREDATE
FROM EMP
WHERE JOB ='CLERK') ;

53.WAQTD NAME AND SALARY FOR ALL THE EMPLOYEES IF THEY ARE

EARNING LESS THAN ATLEST A MANAGER

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

54.WAQTD NAME AND HIREDATE OF EMPLOYEES HIRED BEFORE ALL THE
MANAGERS

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE < ALL ( SELECT HIREDATE
FROM EMP
WHERE JOB ='MANAGERS' ) ;
```

55.WAQTD NAMES OF THE EMPLOYEES HIRED AFTER ALL THE MANAGERS
AND EARNING SALARY MORE THAN ALL THE CLERKS

```
SELECT ENAME
FROM EMP
WHERE HIREDATE > ALL ( SELECT HIREDATE
FROM EMP
WHERE JOB ='MANAGER' ) AND SAL> ALL ( SELECT SAL
FROM EMP
WHERE JOB ='CLERK' ) ;
```

56.WAQTD DETAILS OF THE EMPLOYEES WORKING AS CLERK AND HIRED
BEFORE ATLEST A SALESMAN

```
SELECT *
FROM EMP
WHERE JOB ='CLERK' AND HIREDATE  < ANY( SELECT
HIREDATE
FROM EMP
WHERE JOB ='SALESMAN' ) ;
```

57.WAQTD DETAILS OF EMPLOYEES WORKING IN ACCOUNTING OR SALES DEPT

```
SELECT *
FROM EMP
WHERE DEPTNO IN( SELECT DEPTNO
```

```
FROM DEPT
WHERE  DNAME IN( 'ACCOUNTING','SALES')  );
```

58.WAQTD DEPARTMENT NAMES OF THE EMPOYEES WITH NAME
SMITH , KING AND MILLER

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

59.WAQTD DETAILS OF EMPLOYEES WORKING NEWYORK OR CHICAGO

```
SELECT *
FROM EMP
WHERE DEPTNO IN( SELECT DEPTNO
FROM DEPT
WHERE LOC IN ('NEW YORK','CHICAGO')  );
```

60.WAQTD EMP NAMES IF EMPLOYEES ARE HIRED AFTER ALL THE EMPLOYEES
OF DEPT 10

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

DAY 13

Monday, 29 June 2020 9:34 AM

NESTED SUB QUERY :

" A sub query written inside another sub query is known as Nested sub query ".

- **We can nest about 255 sub queries .**

EMP

ENAME	SAL
A	100
B	200
C	300
D	100
E	200
F	400
G	500

1. WAQTD the maximum salary given to the employees .

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

2. WAQTD second maximum salary .

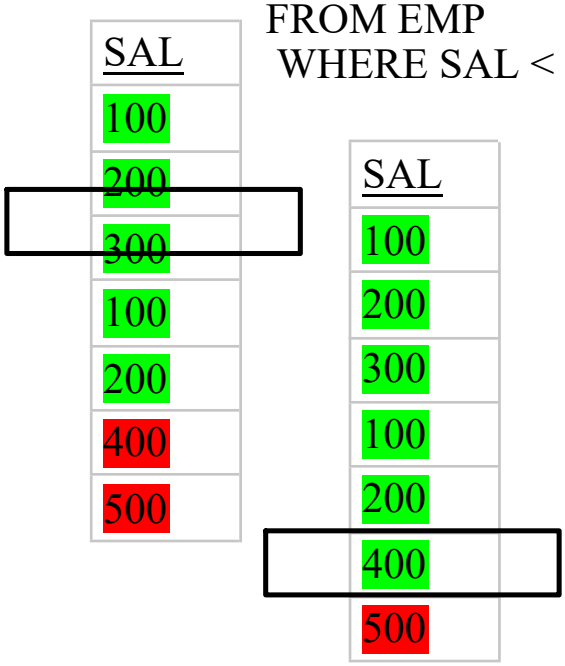
```
SELECT MAX( SAL )
FROM EMP                    500
WHERE SAL < ( SELECT MAX( SAL )
                FROM EMP );
```

SAL
100
200
300
100

3. WAQTD Third maximum salary .

300

```
SELECT MAX( SAL )
FROM EMP                                    400
WHERE SAL < ( SELECT MAX( SAL )
              FROM EMP                                          500
               WHERE SAL < ( SELECT MAX( SAL)
                             FROM EMP ) );
```



4. WAQTD 3 rd minimum salary .

MAX ( )	<
MIN ( )	>

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

5. WAQTD name of the employee getting 2nd Max salary .

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

```
WHERE SAL < ( SELECT MAX(SAL)
                FROM EMP ) ) ;
```

6. WAQTD Dname of the employee getting 3 min 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
                 WHERE SAL > ( SELECT MIN ( SAL )
                               FROM EMP ))));
```

7. WAQTD Dname of the employee getting maximum salary .

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

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

EMPLOYEE - MANAGER RELATION .

EID	ENAME	MGR
1	ALLEN	3
2	BLAKE	1
3	JONES	2

CASE 1 :

TO FIND THE Manager / Reporting Manager .

Example :

1. WAQTD the manager name of Allen .

JONES

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

3

1
2
3

2. WAQTD the name of JONES' s manager .

BLAKE

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

2

1
2
3

3. WAQTD Dname of Blake's Manager .

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

4. WAQTD Allen's Manager's manager name .

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

CASE 2 :

TO FIND THE EMPLOYEE / REPORTERS

1. WAQTD names of the employees reporting to Blake .

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

2. WAQTD salary of the employee reporting to Allen .

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

WHERE ENAME = 'ALLEN');

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

Click the picture of the Receipt and what's app it to 9845687781

Name : Rohan Singh
Phone : 9876543210
Batch : QCDM32
FEE : paid (R) / not paid / partially paid (R)

61.WAQTD 2ND MINIMUM SALARY

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

62.WAQTD 5TH MAXIMUM SALARY

```
SELECT MAX(SAL)
FROM EMP
WHERE SAL < ( SELECT MAX(SAL)
FROM EMP
WHERE SAL < ( SELECTMAX(SAL)
FROM EMP
```

WHERE SAL < (SELECTMAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP)));

63.WAQTD NAME OF THE EMPLOYEE EARNING 3RD MAXIMUM SALARY

SELECT ENAME
FROM EMP
WHERE SAL = (SELECTMAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP)));

64.WAQTD EMPNO OF THE EMPLOYEE EARNING 2D MAXIMUM SALARY

SELECT EMPNO
FROM EMP
WHERE SAL = (SELECTMAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP));

65.WAQTD DEPARTMENT NAME OF AN EMPLOYEE GETTING 4TH MAX SAL

SELECT DNAME
FROM DEPT
WHERE DEPTNO = (SELECT DEPTNO
FROM EMP
WHERE SAL =(SELECT MAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP
WHERE SAL < (SELECTMAX(SAL)
FROM EMP)))));

66.WAOTD DETAILS OF THE EMPLOYEE WHO WAS HIRED 2nd

```
SELECT *
FROM EMP
WHERE HIREDATE = (SELECT MIN(HIREDATE)
FROM EMP
WHERE HIREDATE > ( SELECT MIN(HIREDATE)
FROM EMP ));
```

67.WAQTD NAME OF THE EMPLOYEE HIRED BEFORE THE LAST EMPLOYEE

```
SELECT ENAME
FROM EMP
WHERE HIREDATE < ( SELECT MAX(HIREDATE)
FROM EMP );
```

68.WAQTD LOC OF THE EMPLOYEE WHO WAS HIRED FIRST

```
SELECT LOC
FROM DEPT
WHERE DEPTNO =  (SELECT DEPTNO
FROM EMP
WHERE HIREDATE = (SELECT MIN(HIREDATE)
FROM EMP ));
```

69.WAQTD DETAILS OF THE EMPLOYEE EARNING 7TH MINIMUM SALARY

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

70.WAQTD DNAME OF EMPLOYEE GETTING 2ND MAXIMUM SALARY

SALARY

```
SELECT DNAME
FROM DEPT
WHERE DEPTNO = (SELECT DEPTNO
FROM EMP
WHERE SAL =( SELECT MAX(SAL)
FROM EMP
WHERE SAL < ( SELECTMAX(SAL)
FROM EMP ));
```

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
```

```
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 14

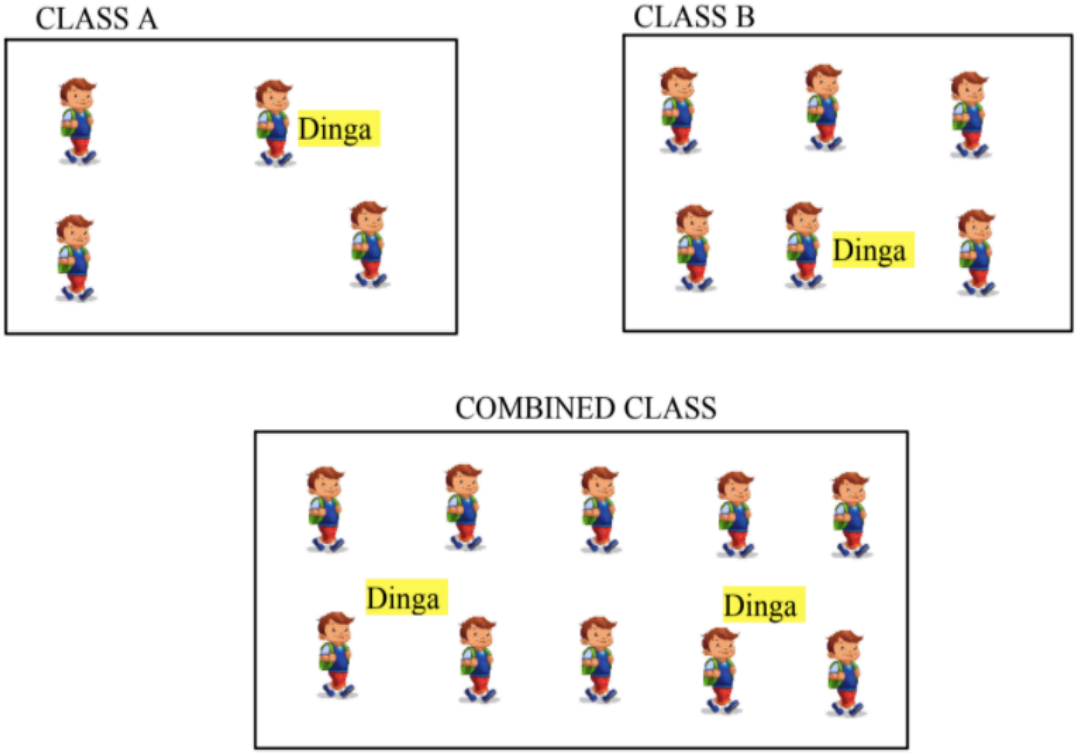
Tuesday, 30 June 2020 9:15 AM

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

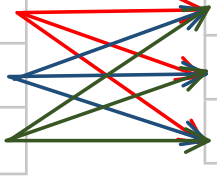
5. NATURAL JOIN .

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
C	10



DEPT

DNAME	DEPTNO
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 column 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 " .

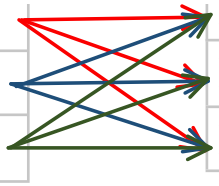
EMP

ENAME	DEPTNO

DEPT

DNAME	DEPTNO

A	20
B	30
C	10



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> ;

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

2. **Oracle**

```
SELECT Column_Name
```

```
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' ;

SELECT ENAME , LOC
FROM EMP INNER JOIN DEPT
ON EMP.DEPTNO = DEPT.DEPTNO
WHERE 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 JOB ='CLERK' AND EMP.DEPTNO = 20 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 , LOC
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.

3. 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

Dept

DNAME	DEPTNO

A	20
B	30
C	10

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

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
FROM EMP , DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO
AND JOB ='SALESMAN' ;
```

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
```

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 15

Wednesday, 1 July 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
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
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 ;
```

1. **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

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	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	MGR
1	ALLEN	3
2	SMITH	1
3	MILLER	2

EMP_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
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.EID ;
```

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.EID ;
```

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

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 ;
```

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

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

QUESTIONS: (TWO TABLES)

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.WAOTD 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 ;
```

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' ;
```

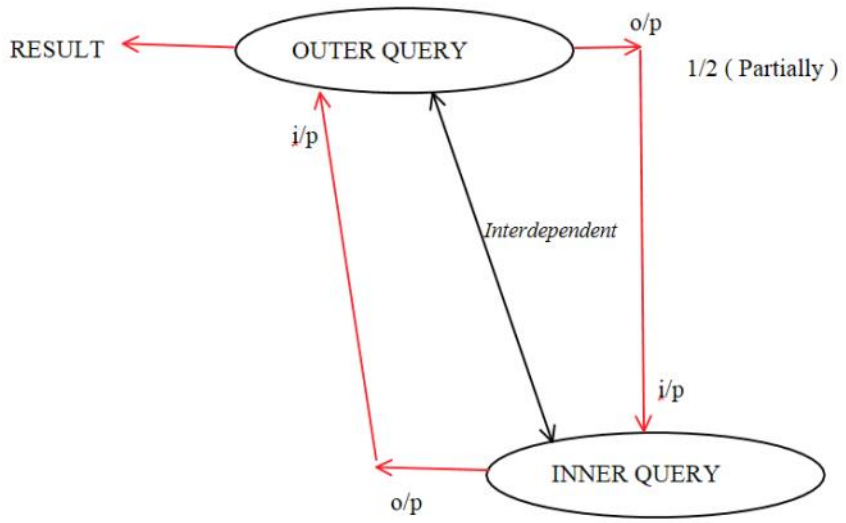
DAY 16

Friday, July 3, 2020

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 .

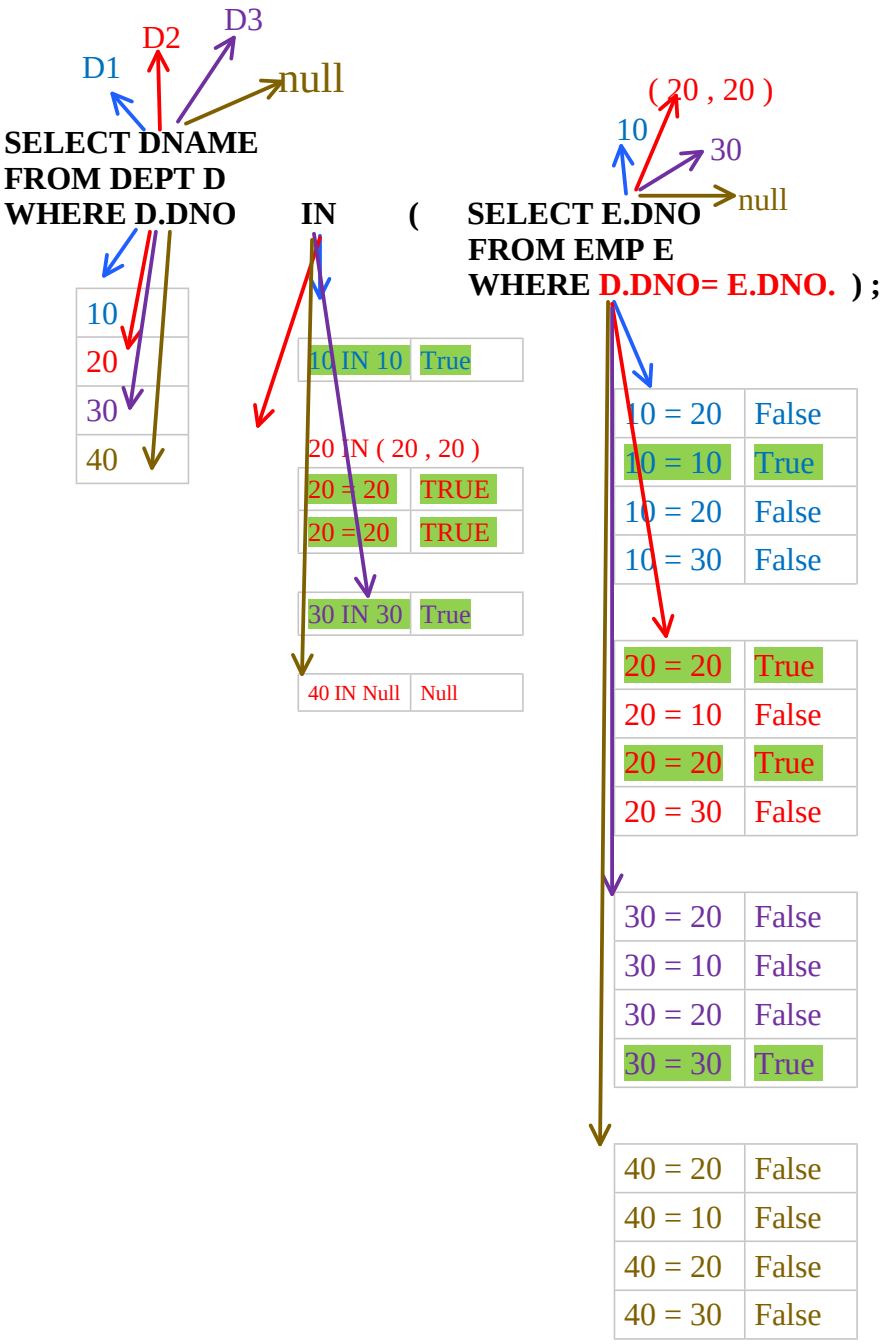
Example :

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 .

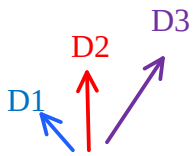
```
SELECT DNAME
FROM DEPT D
WHERE D.DEPTNO IN ( SELECT E.DEPTNO
                    FROM EMP E
                    WHERE D.DEPTNO = E.DEPTNO ) ;
```

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

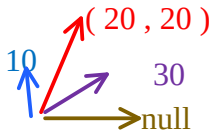
```
SELECT DNAME
FROM DEPT D
WHERE D.DEPTNO NOT IN ( SELECT E.DEPTNO
                        FROM EMP E
                        WHERE D.DEPTNO = E.DEPTNO ) ;
```

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. **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

5000	5000

2nd , 4th , 5th , 7th MAX salary

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

To find MINUMUM salary :

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

TEST PAPER :

INCLUDES [GROUP , SUB QUERY , JOINS]

1.WAQTD NAME OF THE EMPLOYEES WHO ARE HAVING ATLEAST 3 REPORTINGS .
2.WAQTD NAME OF THE DEPARTMENT IN WHICH THERE ARE ATLEAST 3 EMPLOYEES WHOS NAME HAS A CHAR 'A' IN IT .
3.WAQTD NUMBER OF EMPLOYEES REPORTING TO KING.
4.WAQTD NAME OF THE DEPARTMENT IN WHICH SMITH'S MANAGER IS WORKING .
5.WAQTD NUMBER OF EMPLOYEES ARE REPORTING TO BLAKES MANAGER
6.WAQTD NAME OF THE EMPLOYEE ALONG WITH HIS MANAGER ONLY IF MANAGER WORKING IN SALES DEPARTMENT
7.WAQTD DNAME AND AVERAGE SALARY FOR EACH DEPARTMENT EXCLUDING DEPT 20
8.DISPLAY NAME OF THE EMPLOYEES REPORTING TO CLERK'S MANAGER'S MANAGER .
9.WAQTD NAME OF EMP , HIS DNAME , HIS MANAGER NAME , HIS DNAME ONLY IF EMPLOYEES MANAGER'S MANAGER WORKS IN ACCOUNTING DEPARTMENT
10.WAQTD DEPARTMENT NAME IN WHICH THERE ARE NO EMPLOYEES
11.WAQDT NAME OF EMPLOYEE AND HIS MANAGER WITH HIS MANAGER ALONG WITH THEIR
DEPARTMENT NAMES IF EMPLOYEE GETS LESS SALARY THAN MANAGER'S MANAGER .
12.WAQTD NAMES OF EMPLOYEES INDIRECTLY REPORTING TO

KING .
13.WAQTD NAME OF THE EMPLOYEE GETTING 8TH MIN SALARY
14.WAQTD NAME OF THE EMPLOYEE WHO WAS HIRED AFTER 6 EMPLOYEES
15.WAQTD DNAME , ENAME OF THE EMPLOYEES REPORTING TO MARTIN . ONLY IF EMPLOYEE WORKS IN ACCOUNTING OR SALES DEPT .

DAY 17

Monday, 6 July 2020 9:21 AM

ESCAPE CHARACTER

Escape character is used to remove the special behaviors of the Special characters (**% , _**) and treat them as Normal Characters .

- Escape character must be defined
- Escape character must be used before the SC which has to be Treated as a NC .
- The recommended character for EC are [**! , $,** / , \ … etc.].

SYNTAX: Column_Name LIKE 'pattern' **ESCAPE** 'char'

Example :

ENAME
SMITH
ALLEN_DON
MILLER
JAMES_BOND
HE_CALLS_ME_CUTE
WIN%

ENAME LIKE '%_%'

↓

Special Char
With spcl behavior
It can accept any char only once

ENAME LIKE '%!_%' ESCAPE '!'

↓

Normal Char

ENAME LIKE '%!_!_%' ESCAPE '!'

- WAQTD names of the employees having '_' at the first place .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '!_%' ESCAPE '!' ;
```

- WAQTD name of the employee having '%' in the name .

```
SELECT ENAME
FROM EMP
```

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '%!%%' ESCAPE '!' ;
```

- WAQTD name of the employee having 8 characters in the name And 3 and 7th character is '_' .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '__!____!__' ESCAPE '!' ;
```

_	_	!_	_	_	_	!_	_

- WAQTD name of the employee whose name has 3 and 7th character is '_' .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '__!____!_%' ESCAPE '!' ;
```

_	_	!_	_	_	_	!_	%

- WAQTD name of the employee having '$' in the name .

```
SELECT ENAME
FROM EMP
WHERE ENAME LIKE '%$%' ;
```

ORDER BY Clause

Order by clause is used to arrange the records either in Ascending Order Or in descending order .

- By default Order By clause sorts the records in ASC order .
- Order By Clause executes at the last .
- Order by clause executes after select clause .

SYNTAX:

```
SELECT  group_by_expression / group_function
 FROM table_name
[WHERE <filter_condition>]
[GROUP BY column_name/expression]
[HAVING  <group_filter_condition>]
 ORDER BY Col_name/expression [ASC]/DESC ;


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]
6-ORDER BY              [ROW-BY-ROW]
```

ASCII : AMERICAN STANDARD CODE FOR INFORMATION INTERCHAGE

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

- **Every key on the keyboard has an ASCII value .**

DAY 18 / 14

Tuesday, 7 July 2020 8:18 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')

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])

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**' , '**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 / 15

Sunday, May 17, 2020 4:26 PM

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

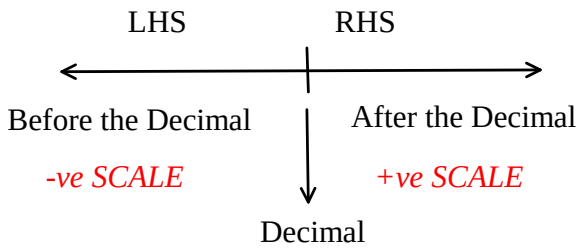
```
SELECT *
FROM EMP
WHERE MOD( EID , 2 ) = 1 ;
```

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)

5.6	6	Rounding off to units place

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

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
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
(123.6712638723 , 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) **SYSDATE = 08-JUL-2020**
2 FROM DUAL ;

LAST_DAY

31-JUL-20

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
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'

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 / 16

Thursday, July 9, 2020 8:45 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			
Foreign Key				

Primary Key	Primary Key			
Foreign Key				

Not Mandatory

Syntax to create a table :

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

Example :

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

NOTE :

To Describe the table:

```
Syntax:    DESC Table_Name ;
```

Example 2:

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

Product

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

Syntax to create a table :

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

Example :

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

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

Syntax: RENAME Table_Name TO New_Name ;

Example :

```
RENAME Customer TO Cust ;
```

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

- **TO ADD A COLUMN :**

Syntax:
```
ALTER  TABLE  Table_Name
ADD Column_Name Datatype Constraint_type ;
```

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

- **TO DROP A COLUMN :**

Syntax:
```
ALTER  TABLE  Table_Name
DROP COLUMN Column_Name ;
```

Example :
```
ALTER TABLE Cust
DROP COLUMN MAIL_ID ;
```

- **TO RENAME A COLUMN :**

Syntax: ALTER TABLE Table_Name
RENAME COLUMN Column_Name TO new_Column_Name ;

Example : ALTER TABLE Cust
RENAME COLUMN CNO TO PHONE_NO ;

- **TO MODIFY THE DATATYPE :**

Syntax: ALTER TABLE Table_Name
MODIFY COLUMN_NAME New_Datatype;

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

- **TO MODIFY NOT NULL CONSTRAINTS :**

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

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

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

Syntax: TRUNCATE TABLE Table_Name ;

Cust

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

Example : TRUNCATE TABLE Cust ;

Cust

Cid	Cname	Phone_no	Address

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

Syntax: DROP TABLE Table_Name ;

Example :

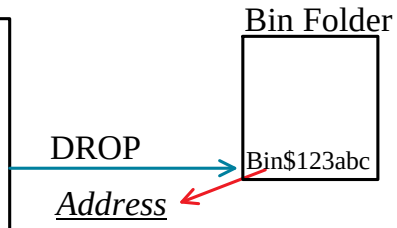
DATABASE

Example :

DATABASE

Cust

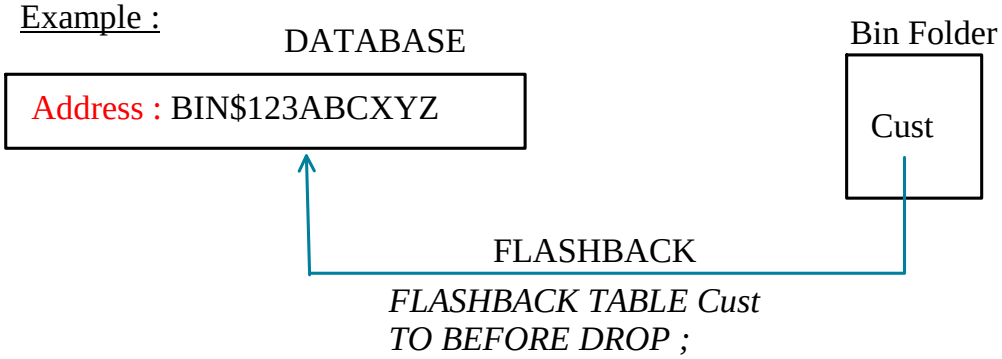
Cid	Cname	Phone_no	Address
1	A	1234567890	BANGALORE
2	B	1234567899	MYSORE
3	C	1234567880	MANGALORE



TO RECOVER THE TABLE :

Syntax: FLASHBACK TABLE Table_Name
TO BEFORE DROP ;

Example :

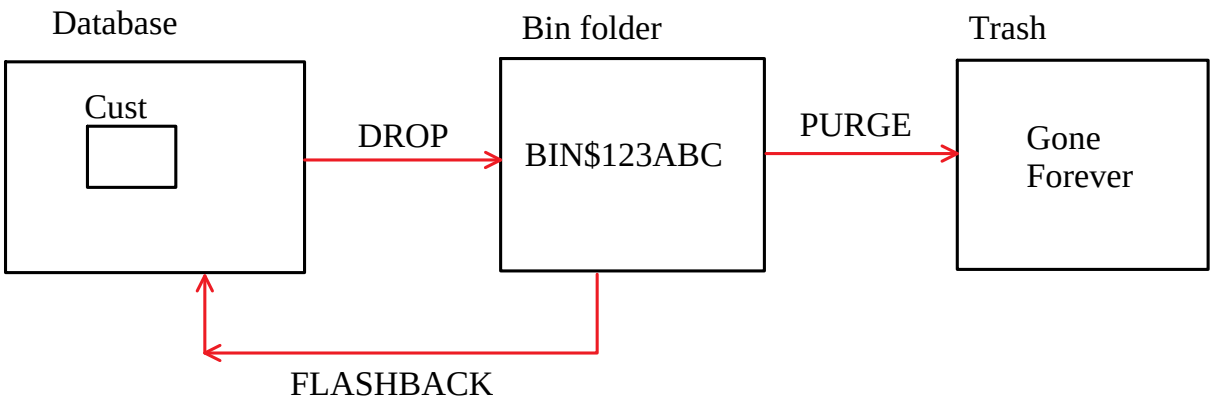


FLASHBACK TABLE Cust
TO BEFORE DROP ;

TO DELETE THE TABLE FROM BIN FOLDER :

Syntax: PURGE TABLE Table_Name ;

Example : PURGE TABLE Cust ;



NOTE : DDL STATEMENTS ARE AUTO-COMMIT STATEMENTS

DAY 21 / DAY 17

Friday, July 10, 2020 9:36 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 ……) ;

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
1	ABHI	1234567890	BANGALORE
2	ABDUL	1234567891	MANGALORE

Ex: INSERT INTO CUSTOMER VALUES(1 , 'ABHI' , 1234567890 , 'BANGALORE');
INSERT INTO CUSTOMER VALUES(2 , 'ABDUL' , 1234567891 , 'MANGALORE');

PID	PNAME	PRICE	CID
NUMBER(2)	VARCHAR(10)	NUMBER(6,2)	NUMBER(3)
10	IPHONE	10000	2
20	ZENPHONE	4000	1
30	ONEPLUS	NULL	NULL

Ex: INSERT INTO PRODUCT VALUES(10 , 'IPHONE' , 10000 , 2);
INSERT INTO PRODUCT VALUES(20 , 'ZENPHONE' , 4000 , 1);
INSERT INTO PRODUCT VALUES(20 , 5000 , 'EARPHONES', NULL); // ERROR
INSERT INRO PRODUCT VALUES(30 , 'ONEPLUS8' , NULL , NULL) ;

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	1234567891	MANGALORE

Change cno of abdul to 9876543210

EX:

UPDATE CUSTOMER
SET CNO = 9876543210
WHERE CNAME ='ABDUL ' ;

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

I have to remove a customer ABHI

Ex:
DELETE FROM CUSTOMER
WHERE CNAME ='ABHI' ;

CID	CNAME	CNO	ADDDRESS
NUMBER(2)	VARCHAR(10)	NUMBER(10)	VARCHAR(20)
2	ABDUL	1234567891	MANGALORE

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

   UPDATE EMP
   SET SAL = SAL*2
   WHERE JOB ='MANAGER' ;

2. WAQT change the name of SMITH to SMIITH .

   UPDATE EMP
   SET ENAME ='SMIITH'
   WHERE ENAME ='SMITH' ;

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

   UPDATE EMP
   SET JOB ='PRESIDENT'
   WHERE ENAME ='KING' ;

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

```
UPDATE EMP
SET ENAME ='ALLEN MORGAN'
WHERE ENAME ='ALLEN' ;
```

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

```
UPDATE EMP
SET SAL = SAL+SAL*10/100
WHERE SAL < 2000 AND JOB ='SALESMAN' ;
```

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

```
DELETE FROM CUSTOMER
WHERE COMM IS NULL ;
```

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

```
DELETE FROM EMP
WHERE HIREDATE < '01-JAN-1987' AND DEPTNO = 20 ;
```

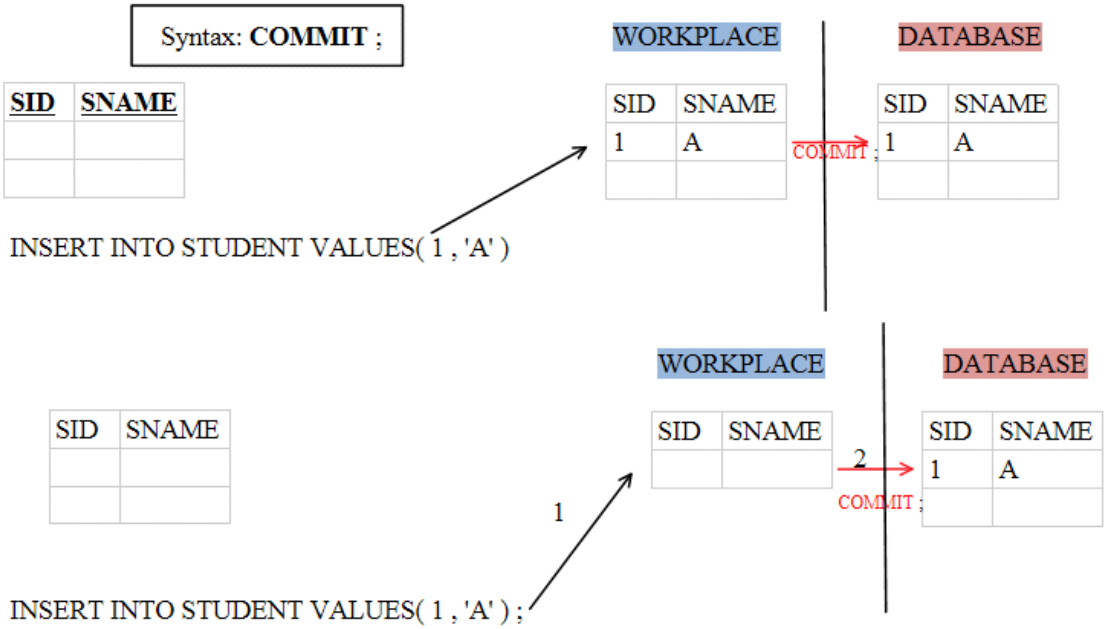
8. Differentiate between TRUNCATE and DELETE statements .

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 .

TRANSACTION CONTRL LANGUAGE (TCL)

1. **COMMIT :** It is used to save all the transactions into the database.

Syntax: **COMMIT** ;

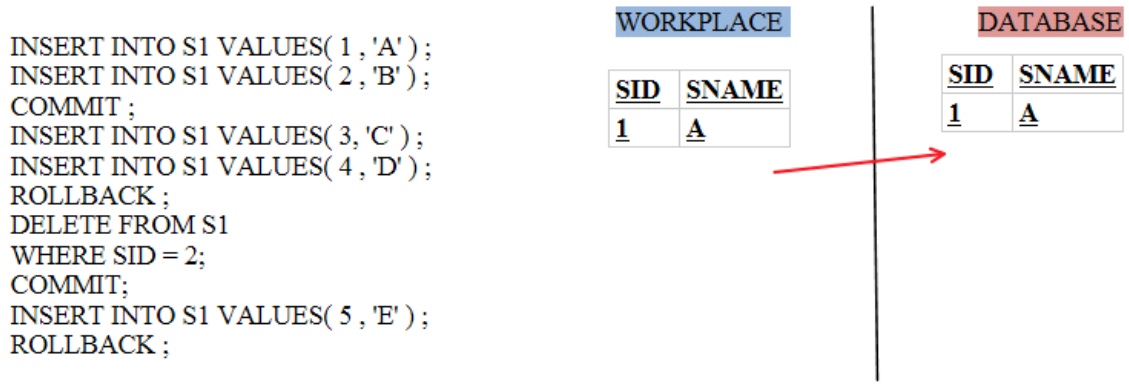


NOTE : Transactions : The operations performed by using DML commands on the table is known as Transaction .

2. **ROLLBACK :** It is used to get back the *latest saved transactions* .

Syntax: ROLLBACK ;

ROLLBACK TO Savepoint_Name ;



3. **SAVEPOINT :** It is used to Mark the transaction and it is used with rollback

Syntax: SAVEPOINT Savepoint_Name ;

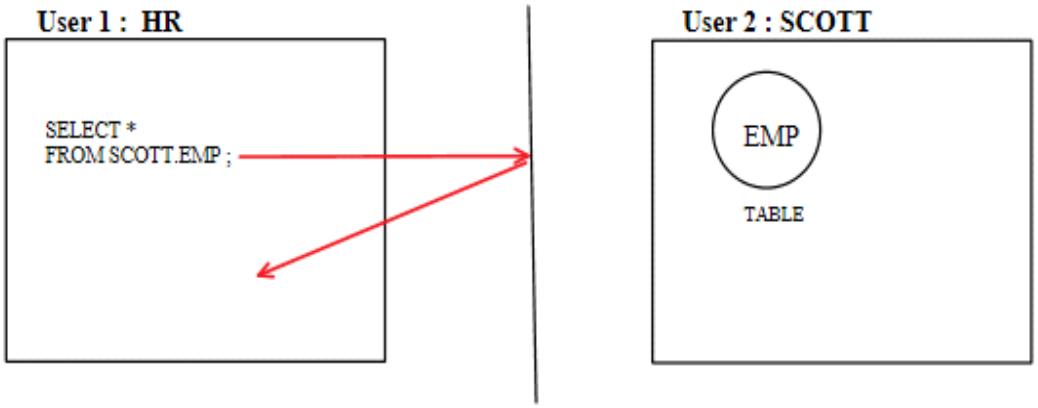
DATA CONTROL LANGUAGE

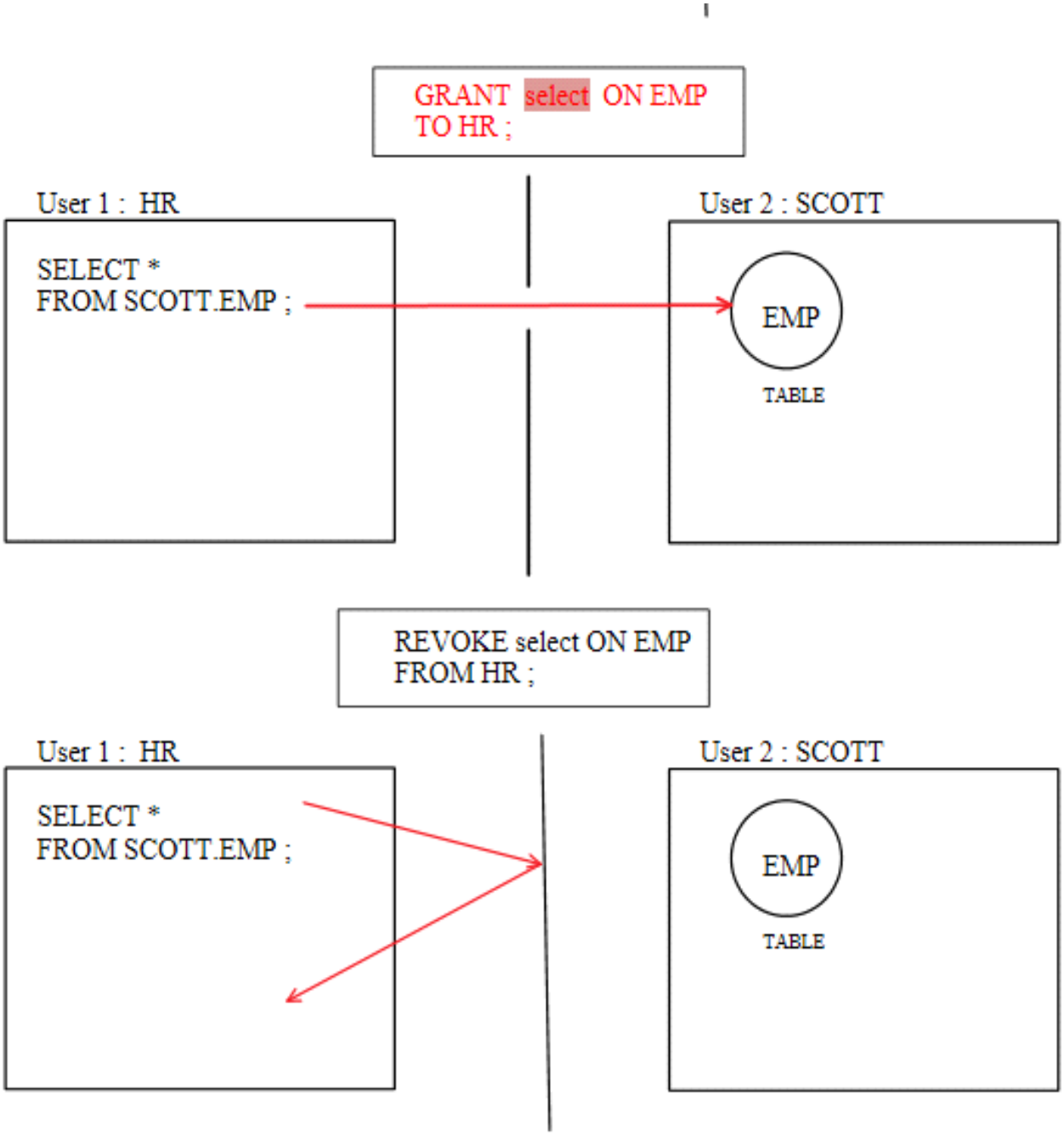
IS used to control the data flow between the users .

1. GRANT
2. REVOKE

1.GRANT : IT is used to give permission to a user .

```
Syntax: GRANT   sql_stmt  ON Table_Name
          TO  User_Name ;
```




GRANT select ON EMP
TO HR ;
User 1 : HR
SELECT *
FROM SCOTT.EMP ;
User 2 : SCOTT
EMP
TABLE
REVOKE select ON EMP
FROM HR ;
User 1 : HR
SELECT *
FROM SCOTT.EMP ;
User 2 : SCOTT
EMP
TABLE

2.REVOKE : IT is used to take back the permission from a user .

Syntax: REVOKE sql_stmt ON Table_Name
FROM User_Name ;

EX: REVOKE select ON EMP
FROM HR ;

EXAMPLE :

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

Grant succeeded.

SQL> CONNECT
Enter user-name: HR
Enter password: *****
Connected.
SQL> SHOW USER
USER is "HR"
SQL> DELETE FROM SCOTT.EMP
  2  WHERE ENAME LIKE 'A%';

2 rows deleted.

SQL> SELECT *
  2  FROM SCOTT.EMP ;
FROM SCOTT.EMP
         *
ERROR at line 2:
ORA-01031: insufficient privileges
```

DAY 22 / DAY 18

Saturday, 11 July 2020 11:59 AM

ATTRIBURTES / PROPERTIES

1. KEY ATTRIBUTE / CANDIDATE KEY : "Any attribute with which we can identify a record uniquely from the table is known as key attribute ".
   Example : SID , MAIL_ID , PHONE_NO

2. NON KEY ATTRIBUTE : "All the other attributes except key attribute is known as NON Key attribute ".
   Example : SNAME , AGE , ADDRESS,PERCENTAGE , DOB , BRANCH , COLLEGE_NAME .

3. PRIME KEY ATTRIBUTE :" Among the key attributes an attribute is chosen to be the main attribute to identify a record uniquely from the table is known as Prime Key Attribute ".
   Example : SID

4. NON PRIME KEY ATTRIBUTE : " All the key attributes except Prime Key Attribute is known as Non prime key attribute " .
   Example : MAIK_ID , PHONE_NO

5. COMPOSITE ATTRIBUTE :" It is a combination of 2 or more non key attributes by which we can identify a record uniquely from the table ". [it is used only when there is no key attribute in the table]
   Example : (SNAME , DOB , ADDRESS)

6. SUPER KEY ATTRIBUTE : "It is a Set of all the key attributes "
   Example : { SID , PHONE_NO , MAIL_ID }

7. FOREIGN KEY ATTRIBUTE : "It is an attribute which behaves as an attribute of another table to represent the relation ship " .

Example : Entity : *STUDENT*

Properties : SID , SNAME , AGE , ADDRESS , MAIL_ID PERCENTAGE , DOB , BRANCH , COLLEGE_NAME , PHONE_NO .

FUNCTIONAL DEPENDENCIES :

" A relation exists such that an attribute determines another attribute uniquely , is known as functional dependency ""

Example : let us consider a relation with 2 attributes X and Y , in which X determines Y (Y is dependent on X).

It is represented as R = { X , Y }

Functional dependency : $X \longrightarrow Y$

X - determinant , Y - dependent

Types of Functional Dependency :

1. TOTAL FUNCTIONAL DEPENDENCY
2. PARTIAL FUNCTIONAL DEPENDENCY
3. TRANSITIVE FUNCTIONAL DEPENDENCY

1. Total Functional Dependency : " If all the attributes of a relation are determined a key attribute we call it as Total Functional Dependency ".

Example : Let us consider a Relation R1 with four attributes A , B , C and D . In which A is Key attribute .

R1 = { A , B , C , D } A* - key attribute

$A \longrightarrow B$
$A \longrightarrow C$
$A \longrightarrow D$

. $A \longrightarrow (B , C , D)$

2. Partial Functional Dependency :
A relation is said to have PFD if
   - It consist of composite key attribute
   - There exists a dependency such that an attribute determines another attribute which is a part of Composite key attribute.

Example : Let us consider a Relation R2 with four attributes A , B , C and D . In which (A , B) is composite key attribute .

R2 = { A , B , C , D } (A , B)" - Composite key attribute

$(A , B) \longrightarrow (C , D)$
$B \longrightarrow D$

3 Transitive Functional Dependency : " There exists a dependency such that

3. Transitive Functional Dependency : " There exists a dependency such that an attribute is determined by another attribute which is in turn determined by Key attribute is known as Transitive Functional Dependency "

Example : Let us consider a Relation R3 with four attributes A , B , C and D . In which A is key attribute .

R3 = { A , B , C , D } A*- key attribute

$$A \rightarrow B$$
$$\boxed{\begin{array}{l} A \rightarrow D \\ D \rightarrow C \end{array}}$$

Redundancy : " The repetition of Unwanted data is known as Redundancy "

Anomaly : " The side effects that occur due to DML operations is known as Anomaly "

Wherever there is Redundancy there is Anomaly !!!

TFD	PFD	TrFD
Redundancy not present	Are present	Are present
No anomalies	Are present	Are present

NORMALIZATION :

" It is a process of converting a Large table into Smaller tables in order to remove redundancies and Anomalies by identifying their Functional Dependencies is known as Normalization " .

Or

"it is a process of decomposing a table into its Normal Form is known as Normalization "

NORMAL FORM :

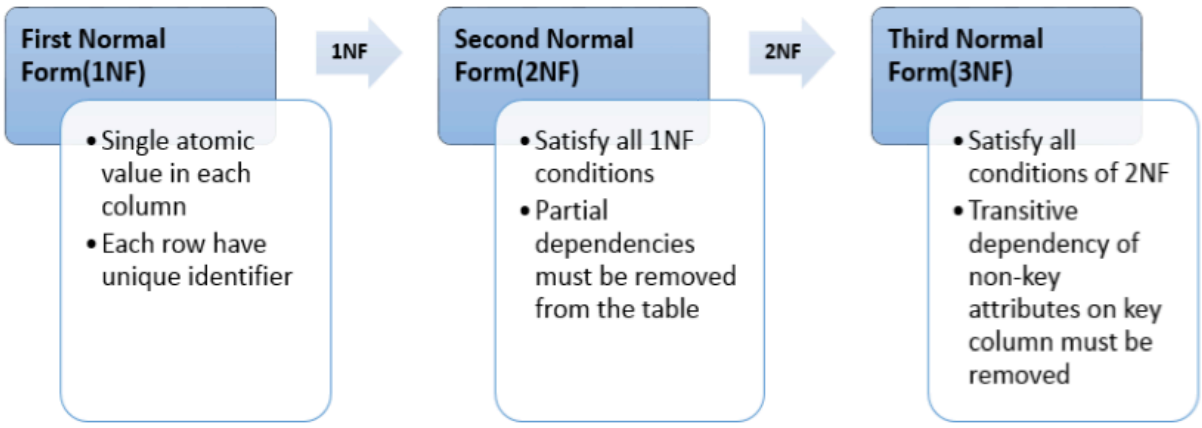
Any table which doesn't have redundancies and anomalies is said to be in NORMAL FORM "

Levels of Normal Form :

1. FIRST NORMAL FROM (1NF)
2. SENCOND MORMAL FROM (2NF)
3. THIRD NORMAL FROM (3NF)

NOTE : if we reduce any given tablet 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	COURSE1	COURSE2	COURSE3
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
Ename - no
Sal - no
Deptno - dname , loc
Dname - no
Loc - no

:- (eid , deptno) - { ename , sal , dname , loc }

(eid , deptno) is a composite key attribute and hence there exists Partial functional dependency .

We have to separate the attributes !!

R1 = { eid , ename , sal }
Eid - ename
Sal

R2 = { deptno , dname , loc }
Deptno - dname
Loc

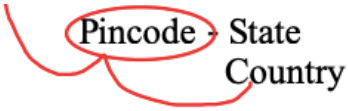
eid	ename	sal	Deptno
1	A	100	10
2	B	120	20
3	C	320	10
4	D	251	10

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 , Pincode , state , country)

EID - ename
Sal
Comm



(EID , Pincode) - { ename , sal , comm , state , country }

R1 = { eid , ename , sal , comm }.
R2 = { pincode , state , country }.

CONTACT :

Notes : bit.ly/roSQLQCDM32
Reference : goo.gl/hVjjxE
Mail id : ro.helpmate@gmail.com
Instagram : ro_sql_helpmate / rohan_singh_ro

Any further details contact your respective branches !!

Here's details of Basvanagudi branch .

Counselors -	**Contact Number**
1	9845687781
2	9686700900

Hr.'s-	**Exp / Freshers**	**Contact Number**
1	Experienced	9663011671
2	Freshers	9686700800
3	Freshers (What's app)	7337885026

Thank You All , LOVE YOU GUYS :)

ALL THE VERY BEST :)

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;

21) LIST THE EMPLOYEES WHOSE SALARY NOT IN THE RANGE OF 1000 TO 2000 AND WORKING IN DEPT 10,20 OR 30 EXCEPT ALL SALESMEN

SELECT ENAME

FROM EMP WHERE

SAL NOT BETWEEN 1000 AND 2000 AND DEPTNO IN(10,20) OR DEPTNO IN 30

AND JOB NOT IN 'SALESMAN';

22) LIST THE DEPARTMENT NAMES WHICH ARE HAVING LETTER ‘O’ IN THEIR LOCATIONS AS WELL AS THEIR DEPARTMENT NAMES

SELECT DNAME

FROM DEPT

WHERE LOC LIKE'%O%' AND DNAME LIKE'%O%';

23) DISPLAY ALL THE EMPLOYEES WHOSE JOB HAS STRING ‘MAN’ IN IT.

```
SELECT ENAME

FROM EMP

WHERE JOB LIKE '%MAN%';
```

24)LIST THE EMPLOYEES WHO ARE HIRED AFTER 82 AND BEFORE 87.

```
SELECT ENAME

FROM EMP

WHERE HIREDATE BETWEEN '01-JAN-83' AND '31-DEC-86';
```

25)WAQTD ALL THE DETAILS OF EMPLOYEES HIRED IN NOVEMBER AND DECEMBER.

```
SELECT *

 FROM EMP

WHERE HIREDATE LIKE '%NOV%' OR HIREDATE LIKE '%DEC%';
```

26)LIST ALL THE EMPLOYEE NAMES AND COMISSION FOR THOSE EMPLOYEES WHO EARN COMISSION MORE THAN THEIR SALARY

```
SELECT ENAME,COMM

FROM EMP

WHERE COMM > SAL;
```

27)WAQTD NAME AND DESIGNATION FOR ALL THE EMPLOYEES HAVING REPORTING MANAGERS AND ALSO THRIE NAMES STARTING WITH ‘S’

```
SELECT ENAME,JOB

FROM EMP

WHERE MGR IS NOT NULL AND ENAME LIKE 'S%';
```

28)WAQTD NAME AND SALARY OF ALL THE EMPLOYEES IF THEIR ANNUAL SALARY ENDS WITH ‘0’ .

```
SELECT ENAME,SAL

FROM EMP

WHERE SAL*12 LIKE '%0';
```

29)WAQTD NAME OF THE EMPLOYEE HAVING ATLEAST 2L’s IN HIS NAME .

```
SELECT ENAME

 FROM EMP

WHERE ENAME LIKE '%L%L%';
```

30)WAQTD NAME OF THE EMPLOYEES WHOS NAME STARTS WITH A ‘VOWEL’

SELECT ENAME

FROM EMP

WHERE ENAME LIKE 'A%' OR ENAME LIKE 'E%' OR ENAME LIKE 'I%' OR ENAME LIKE 'O%' OR ENAME LIKE 'U%';