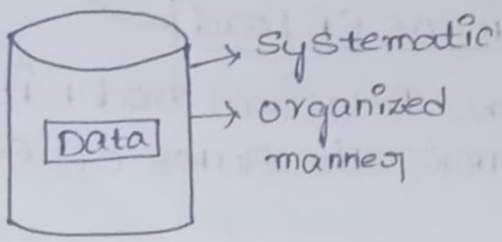
Data:

Data is a raw fact which discribes the attributes of an entity.

Database:

It is a medium there are we store the data in a systematic and an organiged manner.



Data base

* In database, we can peqform universal CRUD opeqation.

CRUD

C → Create / insert

R → Read / qetrieve

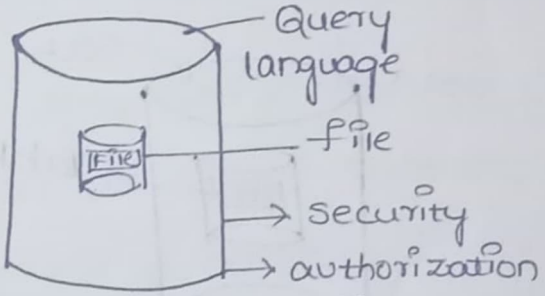
U → update / modify

D → drop / delete

* Date Base management System :-

DBMS is a software which used to maintain and manage the Data base.

* it have two imp feature
  - ① Security
  - ② Authorization



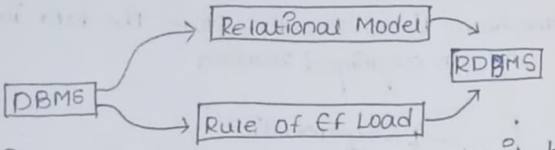
* There can be stored data in the file.
* we need query language to communicate with DBMS.

TYpes of DBMS

① Network DBMS
② Hierchical DBMS
③ Relational DBMS.
④ Object oriented DBMS

Relational Model :-

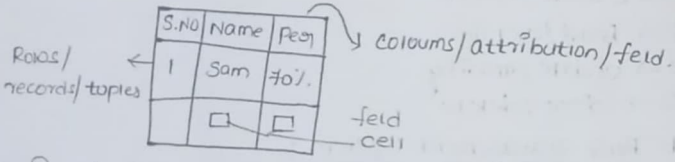
* It was introduced by EF Load
* there we can store everything in form of rows & colu



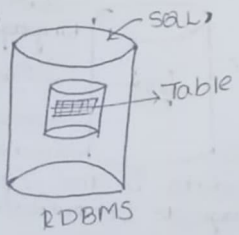
* If any DBMS follows Relational model it become RDBMS or if any DBMS follows the rules of EF Load it become RDBMS

Table :-

* It is a conbination of rows & column



Relationab data base Management SYstem



* RDBMS is we can Store evegthying in a form of table
* We need Stgactuged Quegy Language to Communicate with RDBMS

Rules of EF Codd :-

1. The data enteged into a cell must be an single value data.
2. we can store the data in multiple table, if need we can Establish a connection between 2 different table by using key attribute.
3. we can store everything in the form of table including meta data.

Meta data :- Details about the data

(4) we can validate the Data by assigning into a cell in-to two type → Data base / Constraints

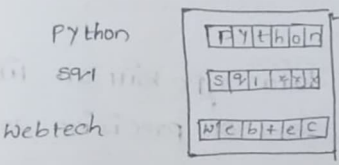
Data types

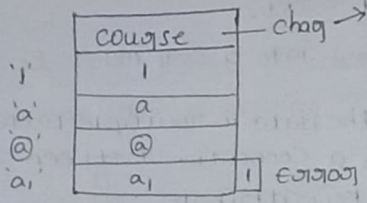
Data type is used to detegmin what type of data that colourn should be hold.

* During a table creation the data type must be given fog add the colums.

chag(size)
This data type will hold group of chaceteg enclose with in single quotes.

* By default it accepts only and chaceteg (size=1)
* This data type is known as fixed memery allocation
* Hence memory is utilized effecEintly.
* It is fasteg compare to varchag and vaggchag2
* It can store up to 1-2000 chag



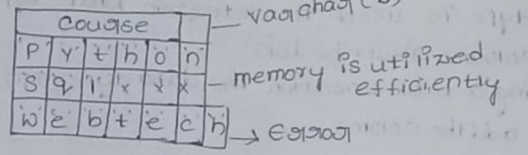


(Default varchar)

② Varchar (size)

Here size represent maximum number of char accepted by colums by compulsory

* This data type is an known as variable memory allocation.
* Hence memory is utilized efficiently
* It can store up to 1-2000 char



③ Varchar 2 (size)

* This Data is type is the next version of varchar data type
* All the feature of varchar data type remains same here accept add the size.
* It can store up to 1-4000 char

④ Number data type (P.S)

Precision

Scale

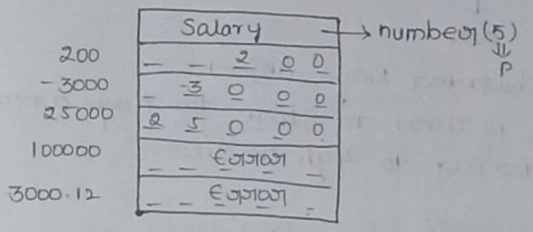
* The colums with number data type will all like int or float data type based on precision and scale even.

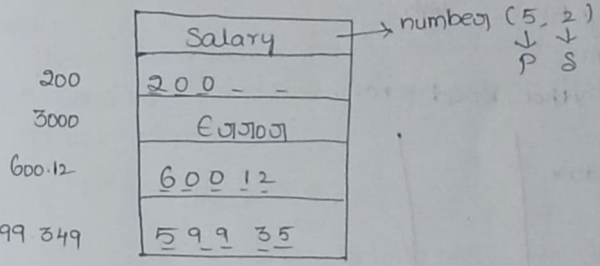
precision :-

* it represent the maximum number of digits that column can hold.
* It range form 1-38.

scaling :-

The maximum number of fractional with in the precision value.





5. Data type :

The column with data type will accept string containing oracle data formate and later it will convert the value to the actual data type.

Oracle data format:

"dd-mon-yyyy"

Ex :- 02-Jan-2020

Note:

"dd-mm-yy" → [0-49] present [20]
→ [50-99] previous [19]

'02-12-98'
↓
1998

'02-12-23'
↓
2023

Character large objects

* Data type is used to store huge ammount of characters up to 4gb of size

(i) CLOS:

Syntax — 4gb

class —

ex: Python, Nodjd etc...

2. BLOB

Syntax

Binary large objects

* This date type is stored to huge ammount of binary values up to 4gb of size.

Syntax → 4gb

BLOB

Ex ing, video, MP4.. etc...

Constaints :—

* The constaints are the rules that are assigned to the coloumn for validation.

Type:

1. unique
2. not null
3. check
4. default
5. primary key
6. forign key

1. unique :- * The unique constraints make sure that all values in a coloumn must be unique

It don't accept duplicate or repeated values

2. NOT NULL :-

It cannot accept null values.

3. Check:

Check contraints having some extra condition if the condition is true it can accept the value else reject them

Ex: check (Sal > 0)

check (length (NO) = 10)

4. Default:

It is used to assign default values to the particular column.

Ex:

default 12-feb-2023

default Dev

default 10

5. primary key

* primary key is used to identifity the recods uniqu form the table.
* It connot accept duplicate and repeated values.
* It connot null values.
* It is the combination of unique and not null contrains.
* we can count only one primary key in the table.
* primary key is not manditory but highly recomed

Ferign key:-

* Ferign key is used to establish this connection between two table
* It can accepts duplicate or repeated value.
* It can accept null values.
* It is not a combination of unique and not Null Constrains.
* To become a forign key it should be primary key in its own table
* primary key is present in parent table forign key is present in child table.
* we can count multiple forgin key in table.
* It is also know as referenical integrate constraints.

Statement on SQL

1. Data defination language ②
2. Transaction control language. ⑤
3. Data manipulation language. ③
4. Data control language. ⑤
5. Data query language. ④

Data query language

This language is used to reterm the data from the data base.

They are four statement

1. select
2. projection
3. selection
4. Joins

Select :- It is process of reterving the data form the table, and display it.

Projection :- It is process of retreving the data form table by selection only the coloum is known as projection.

* All the value be present under the particular coloumn by the default.

Selection :- It is a process of retreving to the data form the table by selection both coloumn as well as rows.

Joins - It is a process of retreeving a data form multiple table is known as joints.

Projection :-

Syntax :

Select * [District] col-name/expression [Alia

form T N;

Order of execution :-

form
selection

Note :-

* Form clause start the execution first.
* It will be gobes to Data base
* select the particulay table then will goes to the Data base. put the table undey execution
* ofteq execution form clause select class to the execution name which be have part.
* so, select class is responsiable forpreparing the result table.

Example :

Write a query to display name of all the employes

```
select ename
form emp;
```

output of form

Emp

EID	EName	Salary	DNO
1	Allen	2000	10
2	Smith	1000	20
3	James	2500	30
4	Ward	1500	20
5	King	1000	30

undey execution

Emp DNO
EID Sal

EId = Empame Ename = Ename

Output of selection

EName
ALLEN
Smith
James
Word
King

Write a salary of all the employe.

```
Select salary
form emp;
```

Write a query to display name & salary form all the employ.

```
select employ, salary
form emp;
```

Write query to display detail all the employee

```
select EID, Ename, salary, D.NO
form Emp;

(or)

Select *
form Emp;
```

"*" => When select entry table to select all select Name can be taken.

"Asterisk" (or) "*" => It mean select very think it form the table

Semicolun :- It mean end of the query.

Distric class

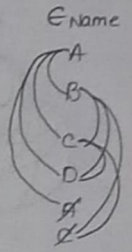
- It is used to remove the duclicate value form the result table.
- Find we should pass Distric first arugement into the selection class.
- We can pass multiple arugument in district class it remove the combination duplicate form both column.

Example :-

Write a query to display different employ name form the employee table.

```
Select Distract fname
form Emp;
```

EID	EName	Salary	D.NO
1	A	100	10
2	B	200	20
3	C	100	30
4	D	300	20
5	A	200	30
6	C	100	30

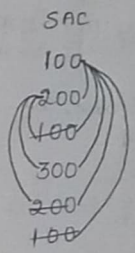


O/p of the selection

EName
A
B
C
D

Write a query to display different salary present in employe table district

```
select ↑salary
form Emp;
```

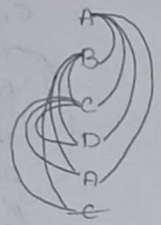


O/p of salary
SAC
100
200
300

Write a query to display different empoly name and salary

```
select Ename, SAC
form Emp;
```

EName	SAL
A	100
B	200
C	100
D	300
A	200
E	~~100~~



O/p of selection

EName	SAL
A	100
B	200
C	100
D	300
A	200

Write a display all the Details form the Employe

```
Select *
form emp;
```

output of select

EId	Ename	Sal	D.No
1	Ramu	1000	10
2	RaJ	2000	20
3	Nikhil	3000	30
4	Ramesh	2000	20
5	Mahesh	4000	10

Write a query to display Name and Salary Given to all the Employee

```
Select Ename, Sal
form Emp;
```

output select

E Name	Sal
Ramu	1000
RaJ	2000
Nikhil	3000
Ramesh	2000
Mahesh	4000

Write a query to display Employ Id and D.NO of all the Emp?

```
Select EId, DNO
form emp;
```

Output select

EId	D.NO
1	10
2	20
3	30
4	20
5	10

Expression :-

If any statement which gives result is known as expression.

* It is a combination of operand and operator

operand can be classifity in two types

1. Coloum
2. values

value can be classifity into three type

1. numberg
2. Chaqracterg
3. Data

Note Characterg and data k senistive we need to In close with the single cource.

Write a query to display annual Salagy of the Emp:

```
Select annual Salagy
formEmploy
```

Write a quary to display half down term salary with Employ

```
Select Salary * 6
form emp;
```

Write a quagy to display annual salary with hike of 10%.

```
Select Sal*12 * 10
form emp;
```

(or)

```
Select Sal * 12 + Sal * 12 * 10/100
form emp;
```

Write a query to display of term salary with reduction 15%

```
Select Sal - 15
form emp;
```

Write a query to display annual salary with bonus of 1000.

```
Select Sal * 12 + 1000
form emp;
```

Alias

It is an alternative name with are given to coloum name (or) express present in result table

* we can use alias name without using keyword
* we can use alias name with (") or (')

Example

```
Sal + 12 AS "Annual Sal" ; Sal * 6
Half_Tech_Sal
form emp;
```

Selection :- It is a process of retering to data form the table by selection both coloum & rows.

syntax :-

```
Select * [Distract] coloum/expiess [Alias]
form emp
where < filteq condition>
```

order of the element :-

```
form
where R-b-R
select
```

Note :

where clause

* where clause is used filteq the records.
* It executiuted row by row.
* It executies afteq execution form class.
* We can pass multiple condition in where classes with the help of logical operation.

Example :-

Write a query to display to name of the employ in the D.NO = 30?

```
Select name
form emp
where D.NO = 30;
```

Emp

EId	Ename	Sal	D.NO	
1	ALLEN	1000	10	⇒ 30(F)
2	SMith	2000	20	⇒ 30(F)
3	James	1500	30	⇒ 30(T)
4	King	3000	20	⇒ 30(F)
5	Jones	2000	20	⇒ 30(f)
6	Word	2500	30	⇒ 30(T)

O/p of where

EID	Ename	Sal	D.NO
3	James	1500	30
6	Word	2500	30

Output of the selection

EName
James
Word

Write a query to display details of the employ earing salary 1000?

```
Select *
form emp
where salary = 1000;
```

Write query to Display Details of the employ hair before the year 81?

```
Select *
form emp
where HireDate < "01-Jan-81";
```

Write a query to Display details of the employ hair the after 17-Dec-80.

```
Select *
form emp
where HireDate > '17-Dec-80';
```

logical operators

1. AND
2. OR
3. NOT

AND

It is a binary operator it return true when both condition are satisfied.

C_1	C_2	Res
0	0	0
1	0	0
0	1	0
1	1	1

OR

It is a binary operator it returns true if any on condition is satisfied

C_1	C_2	Res
0	0	0
1	0	1
0	1	1
1	1	1

NOT :- It is a uniary operator is used to negate the value

C	Res
T	F
F	T

Write query to Display details employ working in D.NO (or) 20

```
select *
form emp
where D.NO = 10 (or) D.NO = 20;
```

Write query to Display details of the employ except Clerk.

```
select *
form emp
where Job != 'clerk';
(or)
where NOT Job = 'clerk';
```

Write query to display details of employ working in D.NO 10 (or) 20 as a cleark.

```
select *
form emp
where D.NO = 10 (or) D.NO = 20 And Job = 'clerk';
```

Operators in Sql

1. Arthamatic operators (+, -, *, >)
2. Concatnation operators (||)
3. relational operators (<>, <=, >=)
4. logical operators (AND, OR, NOT)
5. Comprision operators (=, != (or) <>)
6. special operators
   - i) IN
   - ii) NOT IN
   - iii) Between
   - iv) Not Between
   - v) Like
   - vi) Not Like
   - vii) Is
   - viii) Is NOT
7. Sub Query operators
   - i) All
   - ii) Any
   - iii) exits
   - iv) Not exits.

Concatnation operator -

It is used to joint multiple strings

Example:

```
select 'HI' || ename
form emp;
```

Output

HI SMITH
HI ALLEN
HI WORD

Select 'HAPPY BDAY' || ename || your salary
form emp;

output:- HAPPY BDAY SMITH your salary Is 800
HAPPY BDAY ALLEN your salary Is 1000

Special operators:- In it return through
It is multi value operator if any one condition of satisified for (RHS)

Syntax:
column name/expression in (v1, v2, v3 - - - vn);

Example WAQTD of the employ work in Dept NO (10,20,30,40)

Select *
form emp
where DEPTNO IN (10, 20, 30, 40)

WAQTD of clerk or analyst

Select *
form emp
where JOB IN clerk, analyst

NOT IN It is similary to IN operator instead of selection the value is reject then.

Example:-
WAQTD details of the employ except clerk and manager.

Select *
form emp
where JOB NOT ('CLERK', 'MANAGER');

syntax:-
column name/expression NOT (v1, v2, v3, - - vn);

Between:-
* when we have the range of value we can use between operator.
* we should not inter change the ranges
* we can't use brakets
* IN Include the given range

Examples
WAQTD of the employ earning salary in range of 1000 to 4000.

Select *
form emp
where BETWEEN 1000 AND 4000;

WAQTD of the emp earning with BETWEEN 1000 to 3000?

Select *
form emp
where sal BETEEN 1001 AND 2999;

WAQTD of the employ hire Comm form 1000 to 2000.

Select *
form emp
where comm BETWEEN 1000 AND 1999.

WAQTD details of the supply work sal more than 1000 but less than 5000.

```
select *
form emp
where sal BETWEEN 1001 AND 4999.
```

Write a query to details of the employ hire during the yeag 81

```
select *
form emp
where hire date between '01-Jan-81' and '31-Dec-81';
```

Range	X	X
BETWEEN	+1	-1
From	X	X
><	+1	-1

NOT BETWEEN

It similary to between opegatog instead of selecting the value if reject them.

Syntax:

Col name/expression not between lowegg Range and uppeg Range.

write query to display detail how not hire salary the Range of 1000 to 4000.

```
select *
form emp
where sal Not between 1000 AND 4000;
```

Like

When eveg we want achived pattegn matching we can use opegatog.

Syntax :-

```
Col-name/expression Like 'pattegn matching';
```

To achive pattern matching we can use two special characteg

i) (%)

ii) (_)

Pegcentage.

It can except any numbeg of chagacteg but any chagacteg but also no characteg.

WAQTD name of the employ if the start with characteg 'A'

```
select ename
form emp
where ename Like 'A%';
```

WAQTD name of the employ if the name ends with Chagacteg S

```
select ename
form emp
where ename Like '%A';
```

WAQTD name of the employ if the name is character an in it.

```
select ename
form emp
where ename like '%A%';
```

WAQTD name of the employ if the name start with character A and end with S?

```
select ename
form emp
where ename like 'A%S';
```

WAQTD name of the employ if the name having at the least 2A

```
select ename
form emp
where ename like '%A%A%';
```

underscore

It is can except one char but any character if the name contains as a WAQTD ename Second Character

```
select ename
form emp
where ename like '_A%';
```

WAQTD ename containts most before character is S?

```
select ename
form emp
where ename like '%S__';
```

WAQTD of the employ earning 4 digits salary

```
select *
form emp
where sal like '____';
```

WAQTD name of employ if the name start with vowels

```
select ename
form emp
where ename like 'A%'
      ename like 'e%'
      ename like 'i%'
      ename like 'O%'
      ename like 'U%';
```

NOT Like

It is similer to like operator instead of selecting value its rejects them.

syntax

col name / expression not like 'patter to match';

DAQTD name of employ if name not start with Character A.

```
select ename
form emp
where ename NOT Like 'A%';
```

is

It is used to compare with Null values

WAQTD Details of Emp how are not earning commision?

```
Select*
form Emp
where Comm is Null;
```

isNot

It is used to compare with Not Null values

WAQTD Details of employ if the employ earning commision?

```
Select*
form Emp
where Comm is Not Null;
```

Functions ( )

Functions are block of code (or) least instraction which is used to proform spefic task.

In generally function are classified into two type

1. In builet function
2. user defined function

* In builet function classified into two types
  1. Single row Function
  2. Multi row Function

Single row function

It will take an input and executed then generate the output.

* It executed row by row
* It we pass n number of input it generation n number of output.

Single row function ( )

i) Length ( )

```
select Length (Ename)
form Emp;
```

O/P

Length (Ename)
5
5
4

Multi row function ( )

* It combineds all the input it generates always single output.
* It executed by row by row.
* We pass n number of input it generates only single output.

Multi row function

i) Max ( )

```
select max(sal)
form Emp;
```

O/P

max sal
500

Rules

* We can't pass any other column name with Multi row function.
* We can't pass multiple arguments multi row function,
* We can't pass multi row function in where class.
* It ignore the Null values.
* We can pass asterise (*) present as arrugment only one function.

Types

1. max()
2. Min()
3. avg()
4. sum()
5. count()

WAQTD max salary given to the clerk?

```
select max(sal)
form employ
where JOB = 'CLERK';
```

WAQTD avg salary and minmum salary given to the employe working in dept NO=10?

```
select avg(sal), Min(sal)
form emp
where dept NO = 10;
```

WAQTD NO of employ, start with character A:

```
select count (*)
form emp
where ename Like 'A%';
```

Group by class

* Group by class is used to Group the records
* It exicuted by row by row.
* it exicuted after exicution of form class.
* We can use group by with out use where class also
* If any class exicutes after group by it exicuted group by group.

Syntax:

```
select Group function / group by expression
form TN
where <filter condition>
Group by Col-name / Expression
```

1> form
2> where R-b-R
3> group by R-b-R
4) select g.b.R

WAQTD no. of employees working in each DEPTNO:

```
select count (*)
form emp
group by DEPTNO;
```

output form

EID	Ename	Sal	D.NO
1	ALLEN	2000	10
2	SMith	3000	20
3	James	1000	30
4	Ward	2500	20
5	King	5000	30
6	JONES	1000	20

O/P of group by

10
1 ALLEN 2000 10

20
2 SMith 3000 20
4 Ward 2500 20
6 JONES 1000 20

30
3 James 1000 30
5 King 5000 30

Output of select

Count(*)
1
3
2

Having class

* Having class is used to filter the group.
* it executed Group by Group
* it executes after excution of Group by class
* it can't use having class without Group by class
* We can pass multi row function in having class.
* We don't have using comma, we can use the AND operator.

Syntax

```
Select groupfunction / group by expression
form TN
Where <filter condition>
group by col-name / expression
Having <group filter condition>
```

Output of Execution

```
form
Where R-b-R
group by R-b-R
Having g-b-g
Select g-b-g
```

WAQTD no.of employees working each DEPT. in which atleast two employ and working

```
Select Count(*)
form Emp
group by DEPT
Having Count(*) >= 2;
```

Emp

EID	Ename	Sal	DEPTNO
1	ALEEN	2000	10
2	SMith	3000	20
3	Word	1000	30
4	Milleg	5000	20
5	king	1500	30
6	JONES	2000	20

O/P of group

10

1	ALEEN	2000	10

20

2	SMith	3000	20
4	Milleg	5000	20
6	JONES	2000	20

30

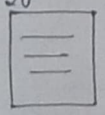
3	Word	1000	30
5	JONES	2000	30

Group of filter condition

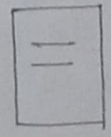
Count(*) >= 2

O/p of Having

20



30



Output of select

Count(*)
3
2

Ordeg by class

* ordeg by class is used to arrange the record neither ascending and desending.
* It executes ROW by ROW.
* It executes after executes of select class.
* If be not mention either ascending & desending it arrange that in ascending ordeg.
* ordeg by class should be last in query.

syntax:-

```
select group function / group by expression
form TN
where < filteg conditions>
group by col-name / expression
Having < group filteg conditions>
```

Ordeg of Execution

```
form
where R-b-R
group by R-b-R
having g-b-g
Select g-b-g
Ordeg by R-b-R
```

Example:

```
select sal
form Emp
ordeg by sal asc;
```

output: 800
9500
1100

Example:

```
select sal
form Emp
where ordeg by sal desc;
```

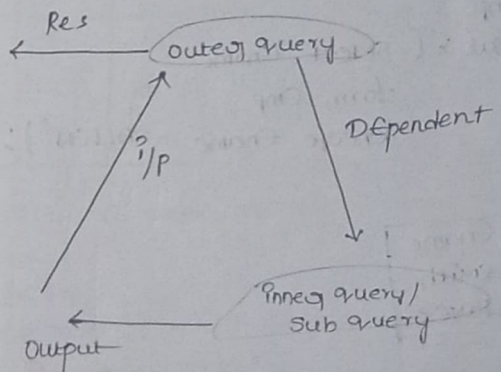
output: 10000
3000
2975

Example:

```
select Ename
form Emp
Ordeg by Ename;
```

output: ADAMS
ALEEN
BLACK

Sub query: A query written inside anotheg query is know as sub query.



WORKING Process

It can consists two query

1. outeg query
2. inneg query

* inneg query executes 1st and produce the output.
* Then the output inneg query is passed as in input to outeg query.
* Then outeg query takes by the input executes and produced final Result oog output.

When we can use sub query

Case 1: When Eveg you have unknow condition We can use Sub query.

Example

WAQTD NO of the employes earn more than ALLEN.

```
Select Ename
form Emp
Wheeje Sal > ( select Ename
               form Emp
               wheeje Ename = "ALLEN");
```

Output

Ename
SMith
JAMES

WAQTD Details of the Employ working in Same designation of SMith.

```
Select (*)
form Emp
wheeje Job = ( select (*)
               form Emp
               wheeje Ename = 'SMith');
```

Case 2: When eveg the data to be selected and condition can useg to be executed form different table we can use sub query.

* selected one table to anotheg table.

Example: Write a query to display name of ALLEN

```
select Dname
form DEPT
wheeje D.ND ( select D.NO
              form Emp
              wheeje Ename = 'ALLEN');
```

DEPT Name

DNA	Lo	DNO
D_1	L_1	10
D_2	L_2	20
D_3	L_3	30

Emp-table

DNO	Ename	Sal
10	ALLEN	2000
20	JOMES	3000
30	WORD	1000
20	King	5000

WAQTD Details of Employ working in accounting DEPT.

```
Select *
form Emp
wheeje D NO = ( select DNO
                form DEPT
                wheeje Ename = "accounting");
```

Type of Sub query

They are Two types

1. Single row
2. Multi row

Single row Sub query

* if the subquery return Exactly one value or record.
* Here Normal operator (or) special operator

Example :-

WAQTD Display name of the employ working in same Department of SMith

Ename	Salary
ALLEN	100
SMITH	150
James	300
Ward	50
ALLEN	200

```
Select Ename
form Emp
where DEPT NO IN (Select DEPT NO
  AND operator      form Emp
                    where Ename = 'SMith');
10 IN (10,30)(T)
20 IN (10,30)(F)
30 IN (10,30)(T)
20 IN (10,30)(F)
30 IN (10,30)(T)
```

Multi row Sub query

if the Subquery return more than one value or record is know as Multi row sub query.

* Here we can't use normal operator we should be use only special operator.

WAQTD Name of employ working same dept of ALLAN?

```
Select Ename
form Emp
where DEPT NO IN ( Select DEPT NO
                   form Emp
                   where Ename = 'ALLAN');
```

Sub query operator

They are four type

1. All
2. Any
3. Exits
4. Not Exits

All operator :- All operator is special operator we can use along with relational operator it return true. if the all the condition satisified.

Example

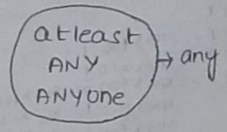
WAQTD name of the employ earning salary more than ALLEN.

```
Select Ename
form Emp
where Sal < All (Select Sal
  AND operator      form Emp
                    where Ename = 'ALLEN');
100 > All (100,200)(F)
150 > All (100,200)(F)
300 > All (100,200)(T)
50 > All (100,200)(F)
200 > All (100,200)(F)
```

ANY operators

* Any operators is a special operator it return True if the any condition is satisfied. We can use any operator in a relational operator.

WAQTD Name of the employ earning less than any of the ALLEN




```
Select Ename
form Emp
where Sal < ANY (select Sal
  or operator       form Emp
                    where Ename = 'ALLEN');
100 < Any (100,200) (T)
150 < Any (100,200) (T)
300 < Any (100,200) (F)
50 < Any (100,200) (T)
200 < Any (100,200) (T)
```

Note: We cannot identify the query ... Multiple value. So always recommand to use special operators.

Nested Subquery: A query return inside subquery is kno... as nestead sub query.

We can nest upto 255 subquery.

Example: WAQTD 1st max salary

```
Select max (Sal)
  form Emp;
```

Output: 500

Salary
100
200
300
400
500

WAQTD 2nd max salary

```
Select max (Sal)
  form Emp
  where Sal < (Select max Sal)
              form Emp);
```

WAQTD Third max salary

```
Select max (Sal)
  form Emp
  where sal < (Select max (Sal)
               form Emp
               where sal < (Select max(sal)
                            form Emp));
```

WAQTD fourth max salary.

```
Select max(Sal)
  form Emp
  where sal < (Select max(Sal)
               form Emp
               where sal < (Select max(Sal))
                           form Emp
                           where Sal < (Select max(Sal)
                                        form Emp)));
```

400 > 100 T
300 > 200 T
300 > 300 F
300 > 400 F
300 > 500 F

400 > 100 T
400 > 200 T
400 > 300 T
400 > 400 F
400 > 500 F

500 > 100 T
500 > 200 T
500 > 300 T
500 > 400 T
500 > 500 F

WAQTD Third min salary?

```
Select min(sal)
  form Emp
  where Sal > (Select min(sal)
               form Emp
               where Sal > (select min Sal)
                           form Emp));
```

WAQTD Name of the employ earning 2nd max salary.

```
Select Ename
  form Emp
  where Sal = (select max(Sal)
               form Emp
               where sal < (select max(Sal)
                            form Emp));
```

WAQTD Name of Employ earning second salary

```
Select Dname
  form DEPT
  where DNO = (Select max (Sal)
               form Emp
               where Sal < (Select max (Sal)
                            form Emp));
```

WAQTD second hiredate of the employ.

```
Select min hiredate
  form Emp
  where Hired > (select min (Hired)
                 form Emp);
```

WAQTD first hiredate of the Employ.

```
Select min (Hiredate)
  form Emp;
  where
```

WAQTD Location of the employes hired before Last Employ.

```
Select Loc
  form dept
  where D:NO In (Select D:NO
                 form Emp
                 where HD = (Select max (Hired)
                             form Emp)
                 where Hired < Select max(
                               form Emp)))
```

EMPLOYEE MANAGER Relation

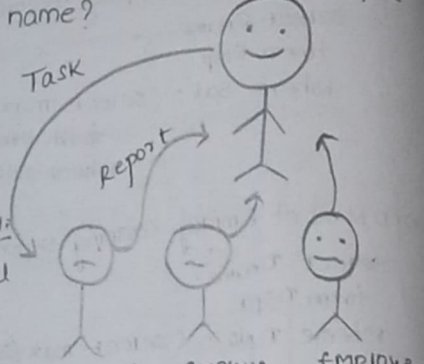
case 1: To find MANAGER

EID = MGR

EId	Ename	MGR
1	ALLEN	3
2	SMITH	1
3	James	4
4	Ward	2

WAQTD ALLEN Manageq name?

```
Select Ename
from Emp
where EID = (Select MGR
             from Emp
             where Ename = 'ALLEN');
```



WAQTD Black Manageq Sal

```
Select Ename, Sal
from Emp
where EID = (Select MGR
             from Emp
             where Ename = 'Black');
```

WAQTD Smith Manageqs Manageq name

```
Select Ename
from Emp
where Empno =
```

WAQTD dept name of the smith manageq

```
select Dname
from DEPT
where D.NO (Select D.NO
            from Emp
            where EmpNO = (Select MGR
                           from Emp
                           where Ename = 'SMITH')
```

Case 2 To find Employee

MGR = EId

WAQTD name of the employ reporting to Smith.

```
Select Ename
from Emp
where MGR = (Select EId
             from Emp
             where Ename = 'SMITH');
```

WAQTD no. of employes reporting to king?

```
Select count(*)
from Emp
where MGR in (Select EId
              form Emp
              where Ename = 'KING');
```

WAQTD Loc of the employ reporting to the James.

```
Select Loc
from DEPT
where D.NO = (Select D.NO
              from Emp
              where MGR IN (Select EId
                            from Emp
                            where Ename = 'JAMES'));
```

Pseudo Coloum

pseudo coloum are the faluse coloum presenting each and every table that need to be call explicitory.

They age two type

1. ROWID
2. colow ROW Num

Example: Select ROWID, ROWNum, Emp.*
from Emp;

ROWID :-

* It is 18 digit Hexadecimal unique address
* It is static we can't change
* It is generated at the time of searching
* We can't update og modufited.
* By useng row ID we can access the record very fasteg.
* By using row ID we can also find duplicate records.

Example: select Rowid, Emp.*
from Emp;

Output:
AAAMfPAACAAAAAgAAA

RowID	Ename
—A	ALLEN
—B	SMITH
—C	JONES
—D	ALLEN

Row Num :-

* Row Num used to asign serial numbeg fog each and eveay row present in resent table.
* It is dynamic
* It generate at the time of execution
* It start with one and incremented by one

Example: select Row Num, Emp*
from Emp

WABTD 1st record in Emp table

select *
from Emp
where Row Num = 1;

WABTD 2nd record in Emp table

select *
from Emp
where Row Num = 2;

Output
No row selected

To Asign RowNum as static

Step1 :- asign rownum to a colourn and ~~rename~~ rename it is SLNO

select Row num AS SLNO, Emp.*
from Emp;

RowNum	Ename
1+1	ALLEN
2+1	SMITH
3+1	James
4+1	ALLEN

Step2 : Pass the step1 query into outeg query from class then gives alas name condition in wheaje To find any record.

select *
from (select RowNum AS SLNO, Emp.*
from Emp)
where SLNO = 2;

Output
2 SMith

WABTD 1st 5 record foam employ table

select *
from (select RowNum as SLNO, Emp * from Emp)
where SLNO IN (1,2,3,4,5);

WABTD 1st half record from employ table

select *
from (select RowNum as SLNO, Emp *
from emp)
where SLNO < (select count(*)/2
from Emp);

Output : 7

WABTD display 2nd half record from employ table.

```
Select *
from (select RowNum as SLNO, Emp * from Emp)
where SLNO <= (select Count(*)/2
                from Emp);
```

WAQTD LOST 5 record from Emplay table

```
select *
from (select RowNum as SLNO, Emp * from Emp)
where S.NO > (select Count(*)-5
               from emp);
```

TO find max & min salary By using pseudo coloumn

Step 1 :-

```
Select Distrct Sal
from Emp
orbeg by sal desc;
```

Sal
500
400
300
100

Step 2 :-

```
Select RowNum as SLNO Sal
from (select District sal
      from emp.
      orbeg by sal desc
```

SLNO	Sal
1	500
2	400
3	300
4	100

Step 3 :-

```
select sal
from (select RowNum AS SLNO, sal
      from (select Distric sal
            from Emp
            orbeg by sal desc))
where SLNO = 3;
```

WAQTD J^{th} minmum salary

```
select sal
from (select RowNum AS SLNO, sal
      from (select Distric sal
            orbeg by sal DSC))
      from Emp
where SLNO = 7;
```

WAQTD 1^{st} three max salary

```
select sal
from (select RowNum AS SLNO, sal
      from (select Distric sal
            orbeg by sal desc))
where SLNO < 4;
```

Joints :- It is process of retreeving a data from multiple table is know as Joints.

* When we can use Joins
* When every we need the data from multiple table we can use Joins.

Types

1. Cortesion (or) Cross Join
2. Inneq Join / Equal Join
3. Natural Joins
4. Outeq Joins
   - Left outeq Joins
   - right outeq Joins
   - full outeq Joins
5. self Joins

Cortesion (or) crossjoins

in cortesion joints a record form Table 1 in manage with all the record of table 2.

* No. of rows present in result Table will be equal to product no of row present in result present in table.
* No of coloumn present in result Table with will be equal to samation no of coloumn present both table.

Ename	DNO
ALLEN	10
SMITH	20
WORD	30

DNO	DName
10	D_1
20	D_2
30	D_3

ANSI

Syntax

```
select A
from TN1 cross join TN2
```

ORACLE

Syntax :-

```
select *
from DEPT, Emp;
```

In cortesion joints it joints generation more number of coping record not use in the cortension use cortension joint instead of using inneg joints.

inneg joins :- it is used to get only the matching the records

Syntax :-

```
select *
from T1 inneg joins T2
on < join condition >
```

Ename	DNO
ALLEN	10
SMith	20
Word	30

DNO	Dname
10	D_1
20	D_2
30	D_3

oracle

Syntax

```
select *
from TN1, TN2
Whege < join condition >
```

join condition < T_1 cul = T_2 coloum >

```
Select *
from Emp, DEPT
Whege Emp.D.NO = DEPT. DEPT NO;
```

Ename	DNO	DNO	Dname
ALLEN	10	10	D_1
SMith	20	20	D_2
Word	30	30	D_3

WABTD Employee name, DEPT Name for all the employes

```
select Ename, DEPT name
from Emp, DEPT
Whege Emp.DEPTNO = DEPT.DEPT NO;
```

WABTD Employee name, Location if the employ working in accounting and DEPT as the manager.

```
select Ename, Loc
from Emp, DEPT
Whege Emp DEPTNO = DEPT.DEPTNO AND JOB =
'MANAGER' AND DName = 'Accouting';
```

WAPTD Employ Name Dept Number
Employ working in Chicago

```
Select Ename, Dnumber, Loc
from Emp, Dept
Where Emp.Dept No = Dept.DeptNo AND Loc =
                    "Chicago";
```

Natural Joints

It has two behavior if there is any connection b/w two table it peqform like join else peqform like cartisian join.

Ename	DNO
ALLEN	10
Smith	20
Word	30

DNO	DName
10	D_1
20	D_2
30	D_3

Ename	CID
C_1	111
C_2	222
C_3	333

ANSI

Syntax:-

```
Select *
from TN1 Natural Joint TN2;
```

Case 1:

```
Select *
from Emp Natural join DEPT;
```

Case 2:

```
Select *
from Emp Natural join Cus;
```

Case 1: o/p

Ename	DNO	Dname
ALLEN	10	D_1
Smith	20	D_2
Word	30	D_3

Case 2: o/p

Ename	DNO	Cname	CID
ALLEN	10	C_1	111
ALLEN	10	C_2	222
ALLEN	10	C_3	333
Smith	20	C_1	111
Smith	20	C_2	222
Smith	20	C_3	333
Word	30	C_1	111
Word	30	C_2	222
Word	30	C_3	333

Outeq join :- It Generate the matching record and un-matching record.

Ename	DNO
ALLEN	10
Smith	NULL
James	20
Word	NULL

DNO	Dname
10	D_1
20	D_2
30	D_3
40	D_4

Left outeq join: It Generate the matching record along with matching record only for Left table.

Ename	DNO
ALLEN	10
Smith	NULL
James	20
Word	NULL

DNO	Dname
10	D_1
20	D_2
30	D_3
40	D_4

Loj syntax

```
Select *
from TN1 Left Outeq join TN2
ON < join condition>
```

ORACLE :- Select *
from TN1, TN2
Where <join condition>

Example:

Select *
from Emp, DEPT
Where Emp.DEPTNO = DEPT.DEPTNO;

Ename	DNO	DNO	Dname
ALLEN	10	10	D1
James	20	20	D2
Smith	Null	Null	Null
Word	Null	Null	

Right outer join :- it generate matching record along with an matching record only from right table.

Ename	DNO
ALLEN	10
Smith	Null
James	20
Word	Null

DNO	Ename
10	D1
20	D2
30	D3
40	D4

ROJ:-
Syntax:-
Select *
from TN1 Right outer join TN2
Where ON <join condition>;

ORacle:-
Select *
from TN1, TN2
Where <join condition>;

Example:

Select *
from Emp, DEPT
Where Emp.DEPTNO(+) = DEPT.DEPTNO;

Ename	DNO	DNO	Dname
ALLEN	10	10	D1
James	20	20	D2
Smith	Null	Null	Null
Word	Null	Null	Null

full outer join

Ename	DNO
ALLEN	10
Smith	Null
James	20
Word	Null

DNO	Dname
10	D1
20	D2
30	D3
40	D4

Defination:- it generate matching record along with un matching record from both table.

Syntax
Select *
from TN1 full outer join TN2
Where ON <full outer join>

Example:

Select *
from Emp full outer join DEPT
ON Emp.DEPTNO = DEPT.DEPTNO;

Ename	DNO	DNO	Dname
ALLEN	10	10	D1
James	20	20	D2
NULL	NULL	30	D3
NULL	NULL	40	D4
Smith	Null	Null	Null
Word	Null	Null	Null

Self Join:- Join the same two table
Join
(or) Join the table by it self is known as self Join.

Example:-

EID	Ename	MGR
1	ALLEN	3
2	Smith	1
3	James	4
4	Word	2

EID	Ename	MGR
1	ALLEN	3
2	Smith	1
3	James	4
4	Word	2

ANSI:-

Syntax:- Select *
from TN1 Join TN2
ON <Join condition>

ORACLE
Syntax:- Select *
from TN1, TN2
where <Join condition>
E1.mgr = E2.EID

Example
Select *
from Emp E1, Emp E2
where E1.MGR = E2.EID;

E1 EID	E1 Ename	E1 MGR	E2 EID	E2 Ename	E2 MGR
1	ALLEN	3	3	James	4
2	Smith	1	1	ALLEN	3
3	James	4	4	Word	2
4	Word	2	2	smith	1

WAQTD Employ name, manager name.
Select E1 Ename, E2 Ename
from Emp E1, Emp E2
where E1.MGR = E2.EmpNO;

WAQTD Employ name, Sal, manager name sal if employ work as clerk in dept No 10?
Select E1 Ename, E1 sal, E2 Ename, E2 sal
from Emp E1, Emp E2
where E1.MGR = E2.EmpNO AND JOB = "CLERK", AND E1 dept NO = 10;

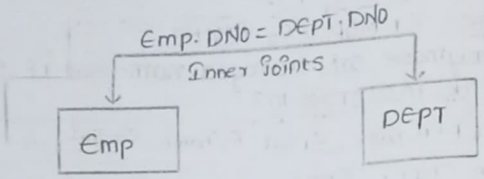
WAQTD Employ Sal manager Hiredate if employ earning Sal less than manager and manager Hiredate after 81?
Select E1 sal, E2 Hiredate
from Emp E1, Emp E2
where E1.MGR = E2.EmpNO AND Hired > E1.Sal < E2.
Sal AND E2.Hiredate > '31-Dec-81';

WAQTD Ename, manager name and manageas manager
Select E1.Ename, E2.Ename, E3.Ename
from Emp E1, Emp E2
where E1.MGR = E2.EMPNO AND E2.Mgr = E3.EmpNO;

WAQTD Name of the employee and His manager name if employee is working as clerk.
Select E1 Ename, E2 Ename
from Emp E1, Emp E2
where E1.MGR = E2.EmpNO AND Job = 'CLERK';

Block Diagram of Joints

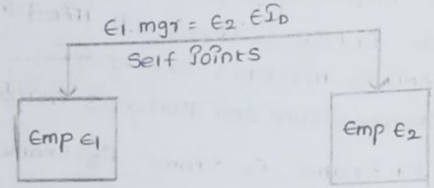
(1)



WAQTD Employ name DEPT name

```
select Ename, Dname
from Emp, DEPT
where Emp.DEPTNO = DEPT.DEPTNO;
```

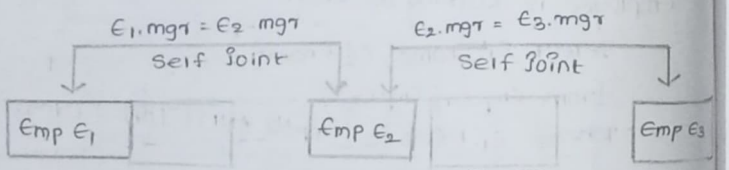
(2)



WAQTD Ename, manager name

```
select E1.Ename, E2.Ename
from Emp E1, Emp E2
where E1.MGR = E2.EmpNO;
```

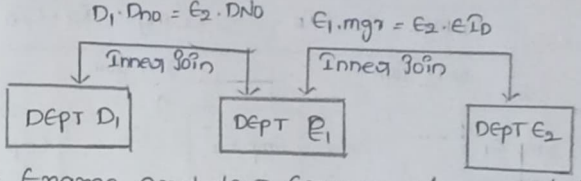
(3)



WAQTD Employ name manager name employee name.

```
select E1.name, E2.name, E3.name
from Emp E1, Emp E2, Emp E3
where E1.MGR = E2.EmpNO AND E2.MGR = E3.EmpNO
```

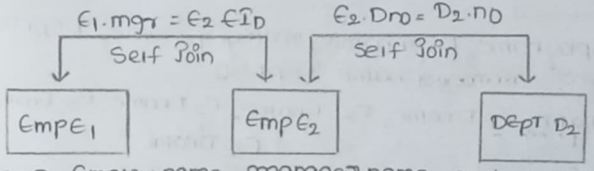
(4)



WAQTD Ename and dept Ename along with manager name

```
select E1.Ename, D1.Ename, E2.Ename
from EmpE1, Emp E2, DEPT D1
where E1.mgr = E2.EmpNO and D1.DEPTNO = D1.DEPTNO
```

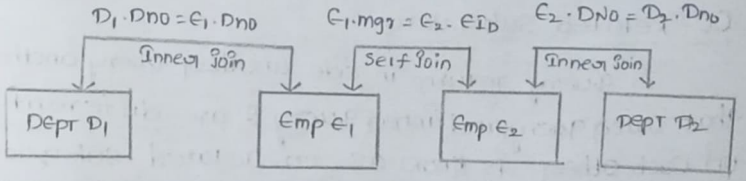
(5)



WAQTD Employ name manager name and manager dept name.

```
select E1.Ename, E2.Ename, D2.DEPTNO
from Emp E1, Emp E2, DEPTNO D2
where D1.mgr = E2.EmpNO and E2.DEPNO = D2.DEPTNO;
```

(6)

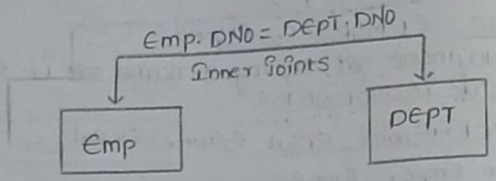


WAQTD Employ name, Employ dept name, mager name Dept name.

```
select E1.Ename, E1.Dname, E2.Ename, D2.DEPTNO
from EmpE1, DEPT E1, Emp E2, DEPT D2
where D1.DEPTNO = E1.DEPTNO and E1.mgr = E2.DEPTNO
and E2.DEPTNO = D2.DEPT and E2.mgr
= E3.EID;
```

Block Diagram of Joints

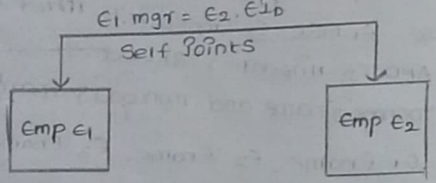
(1)



WABTD Employ name DEPT name

```
Select Ename, Dname
from Emp, DEPT
Where Emp.DEPTNO = DEPT DEPTNO;
```

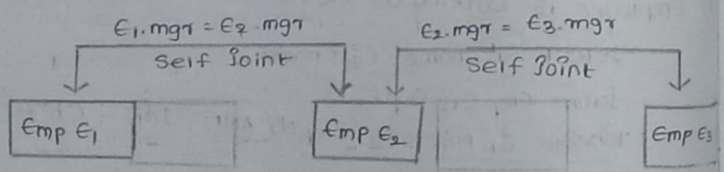
(2)



WABTD Ename, manager name

```
select E1.Ename, E2.Ename
from Emp E1, Emp E2
Where E1.MGR = E2.EmpNO;
```

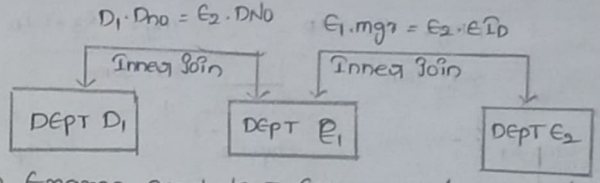
(3)



WABTD Employ name manager name Employee name.

```
select E1.name, E2.name, E3.name
from Emp E1, Emp E2, Emp E3
Where E1.MGR = E2.EmpNO AND E2.MGR = E3.EmpNO
```

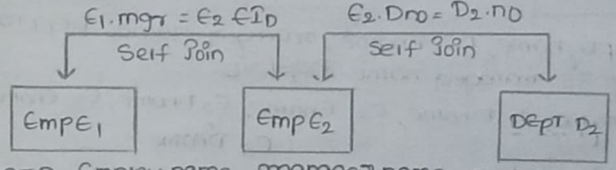
(4)



WABTD Ename and dept Ename along with manager name

```
Select E1.Ename, D1.Ename, E2.Ename
from EmpE1, Emp E2, DEPT D1
Where E1.mgr = E2.EmpNO and D1.DEPTNO = D1.DEPTNO
```

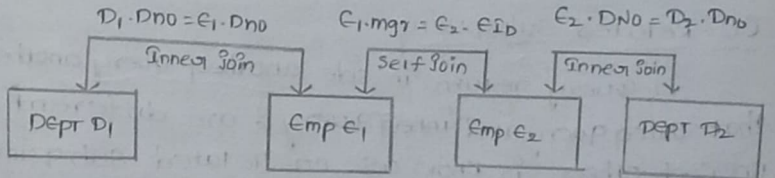
(5)



WABTD Employ name manager name and manager dept name

```
Select E1.Ename, E2.Ename, D2.DEPTNO
from Emp E1, Emp E2, DEPTNO D2
Where E1.mgr = E2.EmpNO and E2.DEPNO = D2.DEPTNO;
```

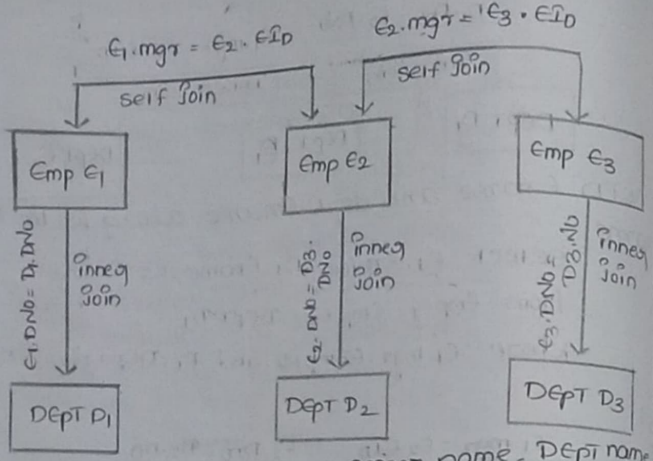
(6)



WABTD Employ name, Employ dept name, mager name DEPT name

```
Select E1.Ename, E1.Dname, E2.Ename, D2.DEPTNO
from Emp E1, DEPT E1, Emp E2, DEPT D2
Where D1.DEPTNO = E1.DEPTNO and E3 mgr = E2 DEPTNO
and E2.DEPTNO = D2.DEPT and E2.mgr
= E3.EID;
```

(7)



WRITE Employname, DEPT name, manager name, DEPT name and manager's manager name DEPT NO.

Select E_1.Ename, E_1.Dname, E_2.Ename, E_2.Dname, E_3.Ename E_3.Dname

from Emp E_1, DEPT D_1, Emp E_2, DEPT E_2, Emp E_3, DEPT E_3

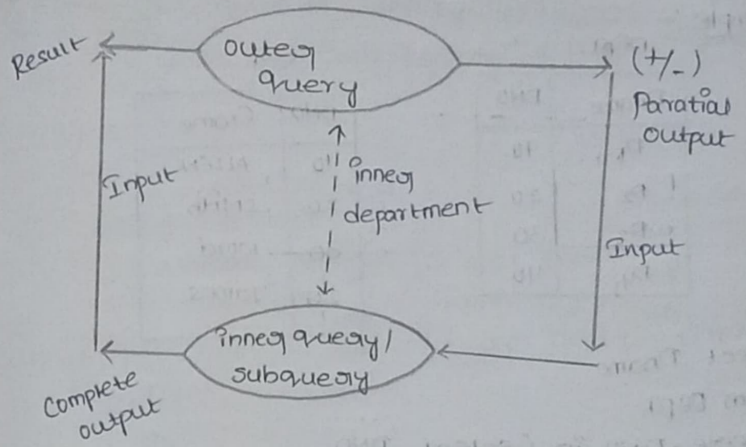
Where E_1.mgr = E_2.EmpNO and E_2.mgr = E_3.EmpNO

E_1.DEPT NO = D_1.DEPT NO and E_2.DNO = D_3.DNO and

E_3.DNO = D_3.DEPT NO.

CO-Related Subquery

A query return inside another query another that outer query and inner query & are different on each other is know as co-related subquery.



WORKING PROCEDURE:

* Let us consider two query outer query & inner query
* Outer query executed first but parial output.
* Then the output of outer query passed as a input inner query.
* inner query takes the input and generate completed output.
* The output of inner query is passed as input to outer query. then outer query taken the input generates final result.
* outer query and inner both are dependendent each other

Note:

* Co-related subquery is working both the principle of subquery and joints so, we need written join collection inside the subquery.

Example :-

DEPT

Dname	DNO
D_1	10
D_2	20
D_3	30
D_4	40

EMP

DNO	Ename
10	ALLEN
20	SMITH
30	WARD
20	JAMES

```
Select Dname
from Dept
where DNO in (select DNO
              from emp
              where DEP.NO = EMP.NO);
```

Step 1	Step 2	Step 3	Step 4	Step 5
10	10 = 10 (T) 10 = 20 (F) 10 = 30 (F) 10 = 20 (F)	10	10 in 10	D_1
20	20 = 10 (F) 20 = 20 (T) 20 = 30 (F) 20 = 20 (T)	(20,20)	20 in (20,20)	D_2
30	30 = 10 (F) 30 = 20 (F) 30 = 30 (T) 30 = 20 (F)	30	30 in 30	D_3
40	40 = 10 (F) 40 = 20 (F) 40 = 30 (F) 40 = 20 (F)	Null	40 in Null	Null

Exists :- It is a unary operator return to the sub query return any value

Example :

```
Select Dname
from DEPT
where exists (select DNO
              from emp
              where DEPT.NO = EMP.NO);
```

Output :

D_1

D_2

D_3

NOT Exists :- It is a unary operator return to the sub query NOT return any value

Example :

```
Select Dname
from DEPT
where NOT exists (select DNO
                  from emp
                  where DEPT.NO = EMP.N
```

Output :

D_4

Note : Exists and NOT Exists both also, used to increase the efficiency of co-related subquery.

To find Maximum and minimum ... Co-related sub query.

Max >= 1

Min <= 1

Emp E_1

Sal
100
200
300
100
400

Emp E_2

Sal
100
200
300
100
400

Syntax:

```
Select E1.sal
from Emp E1
Where (select count (Distrinct E2.sal)
        from Emp E1
        Where E1.sal <= E2.sal) in N;
```

100 <= 100 (T)
200 (T)
300 (T)
100 (T)
400 (T)
— 4

200 <= 100 (F)
200 (T)
300 (T)
100 (F)
400 T
— 3

300 <= 100 (F)
200 (F)
300 (T)
100 (F)
400 (T)
— 2

100 <= 100 (T)
200 (T)
300 (T)
100 (T)
400 (T)
— 4

400 <= 100 (F)
200 (F)
300 (F)
100 (F)
400 (T)
— 1

Write query 5th maximum salary?

```
select E1.sal
from Emp E1
Where (select Count (Distric E2.sal)
        from Emp E2
        Where E1.Emp <= E2.Emp in 5);
```

Write a query to display 2nd minimum salary.

```
select E1.sal
from Emp E1
Where (select count (Distrinct E2.sal)
        from Emp E2
        Where E1.Emp >= E2.Emp in 2);
```

Write a query to display first three max salary?

```
select E1.sal
from Emp E1
Where (select count (Distrinct E2.sal)
        from Emp E2
        Where E1.Emp >= E2.Emp in (1,2,3) or) < 4;
```

Single row function

It will take an input and executed then it generate the output.

* It executed row by row.
* It we pass n number input it generation n number of output.

1) Length() :- It return number of character present in the given string.

Syntax:-

```
Length ('string')
```

Example:

```
select Ename, Length(Ename)
from Emp;
```

Output:

```
SMith  5
ALLEN  5
Word   4
```

Example: select Length (
from Dual;

Output :- 3

2)
Concat () :- It is used to joint the multiple string value.

Example: select concat ('MR', Ename)
from Emp;

Output:- MR SMith
MRALLEN

Syntax: Concat ('String1' 'String2')

3)
Uppeq() :- It convert Given string in uppeq case

Syntax: Uppeq ('String')

Eg:- select Ename
from Emp
Where Uppeq (Ename) = 'ALLEN';

Output:- ALLEN

4)
Loweq () :- It convert Given String into Loweq case.

Syntax: Loweq ('String')

Eg: Select Loweq (Ename)
from Emp;

Output:-
Smith
allen
Word
James
Scott

5)
Initcap() :- It can convert the first character is uppeq case remain all the character Loweq case.

Syntax :- initcap('string')
Select initcap (Ename)
from Emp;

O/p :- SMith
Word
JONES

6)
Reverse () :- It convert the string into reverse foqmat

Syntax :-
Reverse ('string')

Example: select reverse (Ename)
from Emp

Output :-
HTIMS
dOOW
SENOJ

7)
Replace () : it replace old string with New string

Syntax :
Replace ('original string', 'old string', [New string])

Example:
Select replace ('pysipdeq', 'PY', Q')
from Dual;

Output: qsipdeq

Example
Select replace ('pyspideqs', 'PY')
from Dual

Output: spideqs

SubStr() :— It is used to ... from the original string

Syntax :-

SubStr ('original string', postion, [length]);

```
-6 -5 -4 -3 -2 -1
( B A N A N A )
  1  2  3  4  5  6
```

1,1 => B
1,2 => BA
3,4 => NANA
2 => ANANA
-5,2 => AN
4,6 => ANA

Example: Select SubStr ('BANANA', -5, 2)
from dual;

WNBTD Name of the Employees if the name start with vowel

Select Ename,
from Emp
Where SubStr (Ename, 1, 1) in ('A', 'E', 'I', 'O', 'U');

WNBTD name of the employees if the name ends with vowels

Select Ename
from Emp
Where SubStr (Ename, -1, 1) in ('A', 'E', 'I', 'O', 'U');

WNBTD Name of the Employees if the name start with consonents

Select Ename
from Emp
Where SubStr (Ename, 1, 1) NOT IN ('A', 'E', 'I', 'O', 'U');

InStr() :— It return index value of the given string.

Syntax :- InStr ('original string', Postion, [Occurence]);

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

'A', 1, 1 => 2
'A', 1, 2 => 4
'A', 3, 2 => 6
'A', 3, 5 => 0
'AN', 1, 1 => 2
'A', -4, 1 => 2
('A', 1) => 3

Select InStr ('BANANA', 'A', -4, 1)
from Emp;
Output :- 2

WNBTD name of the Employ if the has atleast two A

Select Ename
from Emp
Where InStr (Ename, 'A', 1, 2) > 0;

WNBTD name of the Employ if the name has exactly 2A

Select Ename
from Emp
Where InStr (Ename, 'A', 1, 2) > 0 and InStr (Ename, 'A', 1, 3) = 0;

10)
Round() :— It round the number to nearest integer number value.

Syntax:

Round (number);

5: 2.5 >= 5 = 3
6: 2.5 >= 5 = 3 } increase one value
4 >= 5 = 2
5 >= 5 = 2 } No change the value

TRUNC() - It round the number to the nearest integer value.

Syntax :-

TRUNC (NUMBER)

2.9 => 2
3.9 => 3
3.5 => 3
4.5 => 4

12) MOD() :- It return remender value.

Syntax: MOD(m,n)
m → number
n → column

MOD(Sal,2) = 0 ; Even number
MOD(Sal,2) != 0 ; odd number

WAQTD Even record from Emp Table
Select *
from (Select Rowno AS SLNO, Emp* from Emp)
Where MOD (SLNO,2) = 0;

WAQTD odd record from Emp Table?
Select *
from (Select Row Num AS SLNO, Emp*
from Emp)
Where MOD (SLNO,2) != 0

WAQTD Details Employ Even Salary?
Select *
from Emp
Where MOD (Sal,2) = 0

WAQTD Details of odd Salary
Select *
from Emp
Where Mod(Sal,2) != 0

13) NVL() :- It used to Eleminated Null value (Null value logic)

Syntax :-

NVL (avg1, avg2)

avg1 → NULL → avg2
avg1 → NOT NULL → avg1

Write a query to display total salary of the all the Employ?

Select Sal, Commn, Sal + NVL (Comm,0)
from Emp;

Output

Sal	Comm	Sal + NVL (Comm,0)
800		800
1600	300	1900
1250	500	1750

14) NVL2() :-

Syntax :-

NVL2 (arg1, arg2, arg3)

arg1 → NULL → arg3
arg1 → NOT NULL → arg2

WAQTD total salary of the Employ (Sal+Comm) if the Employ already eaqin Comm Do not given any new Commoration else Given New Commoration as 100.

Select Sal, Comm, Sal + NVL2 (Comm, Comm, 100)
from Emp;

Output:

Salary	Comm	Sal+NVL2 (Comm, Comm, 100)
800		800
1600	500	1900

15) Months_between() :- It return number between the given two data.

Syntax :-

Months_between ('date', 'date2')

Example :

Select Months_between ('01-Jan-25', '01-Dec-24')
Print Dual;

output :

1

16) To_char() :- It is used to astract the format Model form the given date.

Syntax :-

To_char (Date, "format models")

Format Model :-

1> Yeag	8> MM
2> YY	9> HHR
3> YYYY	10> HH24
4> DD	11> DAY
5> D	12> DY
6> Month	13> SS
7) MON	14> MI

Example :-

Select To_char (Hiredate, "HH12 : MI : SS")
from Emp;

TO_CHAR

12 : 00 : 00
12 : 00 : 00
12 : 00 : 00

Select TO_CHAR (HIREDATE, 'MONTH')
from Emp;

output :-

December
JANUARY

WAQTD Details of the employ hair on Monday.

Select *
from Emp
where TO_CHAR (HIREDATE, 'DY') = 'MON';

17) ASCII() :- It return ASCII value for the given character

syntax :-

ANCII ('char')

Example :

Select ASCII ('A')
from Dual;

output :- 65

18) Chr() :- It return character for the given ASCII Number

syntax :

char (Number)

Ex :-

select char(67)
from Dual;

Output :- c

Window function():-

1. Row numberg()
2. Rank()
3. Dense - Rank()

1. Row numberg():-

It is used to asign sequencial integer numberg to the result table.

Syntax:

Row_numberg oveg ([partition optional by coloumn] ordeg by col.name/expression [ASC] [DSC])

Example:-

Select Ename, dept no, sal, row_numberg() oveg (ordeg by sal desc() as Emp sal
From Emp;

Output:

Ename	DeptNO	Sal	Empsal
		5000	1
		3000	2
			3

With partition:-

Select Ename, deptNO, Sal, row_numberg(), oveg(partition by deptno ordeg by sal dsc() as Emsal)
from Emp;

Output:

Ename	deptNO	Sal	Emsal
King	10	500	1
Clerk	10	2450	2
Millon	10	1300	3

Ex:- 1. WAGTD top 3 paid Emp?

Select *
from (Select Ename, deptNO, Sal, row_numberg()
oveg (ordeg by sal desc() as Emp sal
from Emp)
where Emp sal <= 3;

Output:

2. WAGTD top 3 paid Emp from each department?

Select *
from (select Ename, deptno, sal, row_numberg() oveg (partition by deptno ordeg by sal desc) as Emp sal from Emp)
where Emp sal <= 3;

Output:

Ename	deptNO	Sal	Emp sal.

2. Rank():

It is used assign rank with gap.

Syntax:

Rank() oveg ([partision by colname]) ordeg by col-name/expression [ASC]/[DSC])

Ex:-

Select Ename, deptno, sal, rank() oveg (ordeg by sal desc() as Emp sal
from Emp;

Ename	deptno	Sal	
	10	5000	1
	20	3000	2
	20	4000	3

1. WAQTO top 3 paid emp with rankgap?

2. WAQTO top 3 paid Emp from each dept with Rank gap?

3. Dense_Rank():

It is used to assign rank without gap.

Syntax:

Dense_Rank() over([partition by colname] order by colname/expression [ASC]/[DESC]);

Example:-

Select Ename, deptno, Sal, dense_rank() over (order by sal desc) as Emp Sal

from Emp;

output:

Ename	Deptno	Sal	Empsal

Ename Deptno Sal

1. WAQTO top 3 paid Employees without rank gap.

2. WAQTO top 3 paid Emp from each department without rank gap?

Set Operations:-

1. union
2. union all
3. intersect
4. minus

union:- It combines multiple query result and remo
& duplicate values

Example

```
select * from a
union
select * from b;
```

Output A/B

Ename	DEPTNO	sal	empsal
	10	5000	1
	20	3000	2
	20	4000	2

1. WAQTD top 3 paid emp with rankgap?

2. WAQTD top 3 paid emp from each dept with Rank gap?

3. Dense_Rank():

It is used to assign rank with out gap.

Syntax:

Dense_Rank() over([partition by colname] order by Colname/expression [ASC]/[DSC]);

Example():-

select ename, deptNo, sal, dense_Rank() over (order by sal desc) as emp sal
from emp

Output:

Ename	DEPTNO	sal	Empsal

Ename DEPTNO sal

1. WAQTD top 3 paid employees, without rank gap.

2. WAQTD top 3 paid emp from each department with out rank gap?

Set Operatoqs :-

1. union
2. union all
3. intersect
4. minus

union :- It combines multiple query result and remov & duplicate values

Example:

```
select * from a
union
select * from b;
```

Output A/D

1
2
4
5

union all :-
It is similar to union but not removes duplicate values.

Example :

```
Select * form a
union all
Select * form b;
```

Output:

```
AID
1
3
6
9
10
1
4
6
8
10
```

intersect ( ) :-
It return common values from multiple Tables

```
Select * from a
intersect
select * form b;
```

Output :

```
1
6
10
```

Minus :-
It return query result of table, values is the value not present in table two query result

ex

```
select * form a
minus
select * form b;
```

Output :

```
3
9
```

case :
It is used to write conditional logic with in query

syntax :

```
case :- when condition 1  then result 1
        when condition 2  then result 2
        —
        =
        —
        else 'default res'
        end as aliyas.
```

1. WAQTD emp earning low salary, mid, high salary.

```
select ename, sal, case,
        when sal <= 1000  then 'low sal'
        when sal between 1001 and 3000 then mid
        else 'high sal'
        end as empsal
from emp;
```

union all :-
It is similar to union but not removes duplicate values.

Example:

```
Select * form a
union all
Select * form b;
```

Output:

```
AID
1
3
6
9
10
1
4
6
8
10
```

intersect() :-
It return common values from multiple Tables

```
Select * from a
intersect
select * form b;
```

Output:

```
1
6
10
```

Minus :-
It return query result of table, values is the value not present in table two query result

ex

```
select * form a
minus
select * form b;
```

Output:-

```
3
9
```

case:
It is used to write conditional logic with in query

syntax:

```
case :- when condition 1  then result 1
        when condition 2  then result 2
        —
        —
        —
        else 'default res'
        End as aliyas.
```

1. WAQTD Emp earning low salary, mid, high salary.

```
Select ename, sal, case
      when sal <= 1000 then 'low sal'
      when sal between 1001 and 3000 then mid
      else 'high sal'
      end as Empsal
from Emp;
```

DDL :- Data Definition language.-

This language is used to construct or modify or remove object (tabel) form the data base.

1. Create
2. Rename
3. Alteg
4. Truncate
5. Drop

1. Create : This statement is used to create the tabl inside data base.

Syntax : This statement is used to create the table inside data base.

```
Create Table T_N
(
Colname1  Data type [constraints]
Colname2  Data type [constraints]
- - -
- - -
Colname n  Data type [constraints]
);
```

Example :

```
Create Table product
(
Pid number(2) primary key,
Pname varchar(20) not null,
Price number(8,2) not null check (price>0);
Quantity number(2 Default '1')
Mfd Date
);
```

Dese Product :

To create child table with foriqn key :

Syntax :-

```
Create Table T_N
(
Coluname  Data type [constraints]
Coluname  Data type [constraints]
- - -
- - -
Col.name n Data type,
constraints coiname_fk foreign key (coiname)
            reference product (PID)
```

Example :

```
Create Table Castomeg
(
(ID numbeg (2) primary key,
(name vaqchaq (20) not null,
(no numbeg (10) unique not null check (Length((no) =10);
Address vaqchaq (20),
PID numbeg (2),
Constraints PID_fk foreign key (PID) reference product (PID)
```

2. Rename :

It is used to change the existing table name.

Syntax :

```
Rename Table name To new_name;
```

Example :

```
Rename Castomeg to cus;
```

3. Alteg :

It is used to modify the existing table structure

1. To add column

Syntax :

```
Alteg Table T_N
Add colname Data type [ constraints];
```

(ii) To Rename column :-

Syntax : Alter table table-name
Rename columnname To newname

Example :
alter table cus
Rename mail id To mail

(iii) To change date type :-

Syntax : Alter table t-name
modify column name

Example : Alter table cus
modify mail char(30);

(iv) To change constraints :-

Syntax :- Alter table t-name
modify col-name newdatetypeconstraints;

Example :- Alter Table cus
modify mail char(30) null;

(v) To Drop column :-

Syntax :- Alter table table-name
Drop column-name;

Example :- Alter table cus
Drop column mail;

(4) Truncat : It is used to delete the all records permanently form the table without affecting table structure

Syntax :-
Truncate Table Table_name

(5) DROP : It is used to Delete table form database

Syntax :- Drop table table-name

Example :- Drop table salesman;

select *
form Recyclebin;
It is used to check table is present in recycle bin or not

Flash back :- It is used to recovery the table form the Recycle bin

Syntax : Flashback table - table name
To before drop;

PURGE :- It is used to Delete the table form Recycle bin

Syntax : purge table table-name
example : purge table salesman;

DESC

DML (Data manipulation Language)

This language is used to add, modify or delete the data form the table [It is non auto comment]

It has 3 statements :-

1) Insert
2) update
3) Delete

INSERT :- It is used to insert the data into a table.

Syntax:-

insert into T_N (col-n_1, col-n_2, col-n_3) value (&col-n_1, &col-n_2, &col-n_3)

Example:-

insert into product (Pid, Pname, price, Quantity, MFD) value ('&Pid', '&Pname', '&price', '&Quantity', '&MFD');

Enter value for Pid: 01
Enter value for Pname: samsung
Enter value for price: 2000
Enter value for Quantity: 2
Enter the value for MFD: 13-Sep-23

1 Row Created

/→ To insert more than one row

→ Commit; # save the data

To insert null values:-

Enter value for Pid: 02
Enter value for Pname: VIVO
Enter value for price: 30000
Enter value for Quantity: 1
Enter value for MFD: —

To assign different Quantity other than default value

Enter value of Pid: 01
Enter value for Pname: Samsung
Enter value for price: 20,000
Enter value for Quantity: 2
Enter value for MFD: 13-Sep-2022

To assign Default values:-

Syntax:- insert into T_N (col-n_1, col-n_2) values (&col-n_1, '&col-n_2', Default, &col-n_3);

Example:

insert into product (Pid, Pname, price, Quantity, MFD
values (&Pid, &Pname, &price, Quantity, Default, &M

Enter value for Pid: 03
Enter value for Pname: MI
Enter value for price: 2000
skip out
Enter value for MFD: 22-08-2024

→ Commit;

Update: It is used to modify the existing data form the table.

Syntax:- Update T_name
set col_name = value, col-n_2 = value ---N
where <filter condition>

Example:
Update product
Set MFD = NULL
where Pid = 5
→ 1 row update.

→ select *
form product
Display the modified record in column/table.

Delete:- It is used to delete particular record

syntax:- Delete form table_name
where <filter_condition>

Example:
Delete form product
where Pid = 5;

TCL - Transaction control Language:

It is used to control the transaction done on DML opeqations.

They are three type.

1. Commit
2. Save point
3. Roll back

COMMIT :- It is used to control

Syntax :- Commit;

Savepoint :- It is used to mark the transaction

Syntax :- Save point spname;

4 → commit

(4 - S1
5 - S2
6 - S3)

Roll back :- It is used to perform undo operation

Syntax :- Roll back;

DCL (Data CONTROL Language)

* SHOW USER
→ USER IS "SCOTT"
* Connect :
Enter user_name : HR
Enter password : Tiger
Connections.

Defination :- It is used to control the data flow between users.

They are two types.

1. Grant
2. Revoke

i) Grant :- It is used given the permission to user.

Syntax :-

Grant SQL Statement (Insert, Delet, Select) on TN
To user name;

Example : Grant select on EMP to HR;

Grant succeeded.

(ii) Revoke : It is used to take back the permission form user.

Syntax :-

Revoke SQL Statement (Select, Delete) ON T_N
form username

Example : Revoke select an emp form HR;

HOW to Copy a table :-

Syntax :- Create Table T_N (New)
AS
Select Statement
form TN (old)
Where <filter condition>

Example :-

Create table CLERK
AS
Select *
form Emp
Where JOB = 'CLERK';

TCL :- Transaction control Language

control the transaction control language done on DML operatoy it has three statement.

1. Commit
2. save point
3. Roll back

Commit :- It is used to save the transaction

Syntax :-

Commit ;

save point : it is used to mark the transaction

Syntax :-

save point save point name ;

Roll back :- it is used to form formed to undo opeqation

Syntax :- Roll back ;

DCL :- Data control language

It is used to control is data flow between the user

They aqe two types

1. grant
2. revoke

Grant :- it is used to give peqmission to useq

Syntax :

grant SQL statement on T-N

To useq name.

Example :

Grant Select ON EMP To HR ;

Revoke : It is used take back to peqmission from useq

Syntax :

Revoke SQL statement ON T-N

from useq_name ;

Example :

Revoke Select ON EMP foq HR ;

View :- View aqe virtual table created based on result set of query.

They aqe three type.

1. Simple view
2. Complex view
3. Materialized view

simple view :-

* it is form single from table
* If we change any thing are update in view it affect parent table (vice-versa)

syntax :-

```
Create view V-N
AS
select statements
form TN
wheqe <filteq condition> ;
```

connect :

useq name : System

Password : Tigeq

Grant Create view to SCOTT ;

```
create view salesman
 AS
select *
-from emp
wheqe JOB = 'SAESMAN' ;
```

Complex view :- it is form multiple table joints multi and also combination of row function qroup by and having

* we can't update complex view.

syntax :-

```
Create view V-N ;
AS
Select group function / qroup by expreq
from TN, TN+
wheqe < join condition>
Group by col-name / expression
```

Having < group-filteq condition >

WABTD No.of Employ, DEPTNO with, Dname?

```
Create view cv
AS
Select Count(*) AS Count, Emp.DEPTNO, Dname
form Emp, DEPT
Where Emp.DEPTNO = DEPT.DEPTNO
Group by Emp.DEPTNO, Dname.
```

Materialized view :-

Normal view store the query but materialized store the result

* We can't update materialized view.
* it very fasteq than normal view.

conn:
user name : system
Password : password
grant create materialized view to SCOTT;

Syntax :-

```
Create materalized view V-N
AS
Select Group function / group by expression
form TN1, TN2
Where < join condition >
Group by col.name / Expression
Having < Group filteq condition >
```

WABTD no.of Employ with DEPTNO

```
Create materialized view MV
AS
Select count(*), DEPTNO
from EMP
Group by DEPTNO;
```

To refresh

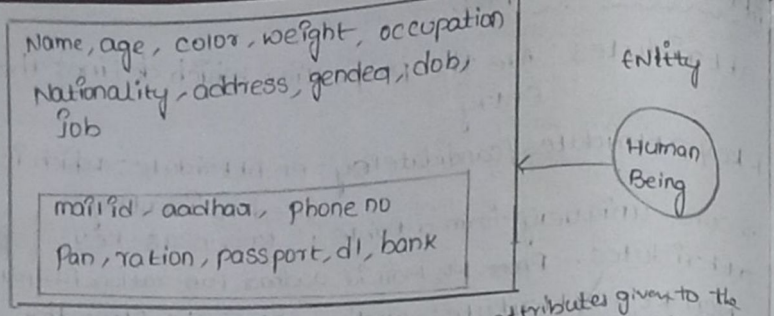
```
execute.DBMS_MVIEW.REFRESH ('MV');
```

Attributes in SQL

Attributes : Are the properties which defines the entry.

1. key attributes / candidate key : An Attributes which is used to identify a record uniquely form a table is known as key attributes. Phone_No, Mail_id, aadhaq, Pan, ration, Passport
2. NON key attribute : All the attribute otheq than by key attributes
   Ex : Name, age, gendeq, dob.
3. primaqy key attribute :- Among the key attribute is chosen to be main attribute is chosen to be main attribute to identify a record uniquely form the table is known as prime key attribute.
   Ex : phone_no :
4. NON-prime key attribute : All the key attributes otheq than prime key attribute.
   Ex : mail_id, aadhaq, Pan, ration, Passport, dl, bank
5. composite key attribute : it is combination of two oq moge non key attributes, which is used to identify a record uniaue form the table.
   * Composite key is found whenevep there is no key attri
6. Foreign key attribute :- it is an attribute which behave as an attribute of anotheq entity to represent the relationship. (phone_NO-mail, aadhaq, Pan, ration, Passport.
   super, setofall.
7. Foreign key attribute : it is an attribute which behave

   Ex : DNO



FUNCTIONAL DEPENDENCY :- attributes given to the another attributes.

There exits A Dependency such that an Attribute in a relation determines another attribute.

EMP - (EID, Ename)

EID → Ename

EID	Ename
1	A
2	B
3	C

Types of functional Dependencies:

1. Total functional Dependency
2. Partial functional Dependency
3. Transitive functional Dependency.

1. Total functional Dependency :-

If an attribute in a relation determine all the attributes it is known as TFD or if the attributes are dependent on a single attribute the it is known as TFD

EMP - (EID, Ename, Sal, DOB)

EID * KEY Attribute

EID → Ename
EID → Sal
EID → DOB
EID → (Ename, Sal, DOB)

Partial functional Dependency :-

There exists a dependency such that a part of composite determine another attribute uniquely

COSTOMER - (Cname, Address, mail_ID, phone_No)

(phone_NO, MAIL_ID) → Composite key attribute

phone_NO → Cname, Address
mail_ID → Cname, Address

Customer

Cname	Address	mail ID	phone ID
Smith	mysore	Smith@gmail.com	
miller	Bangalore		1001
scott	mangalore	scott@gmail.com	
Adams	mysore		2002
scott	Delhi		3003

3. Transitive functional Dependency :-

There exists a dependency such that an attribute is determined by a non-key attribute which is intern determined by a key attribute.

X - (A, B, C, D)

A → B
A → D
D → C

A* key attributes
A = B
B = C
A = C

Costomer (CID, Cname, PINcode, STATE)

CID → Cname
CID → PINcode
PINCode → state.

Redundancy :- The Rreptition of data known as redundancy.

Anomaly :- The side effect caused during DML Opeqations is know as Anomaly.

Total	Partial	Transitive
No Redundancy	Redundancy Exists	Redundancy exists
No Anomaly	Anomalies are present	Anomalies are present

Customer

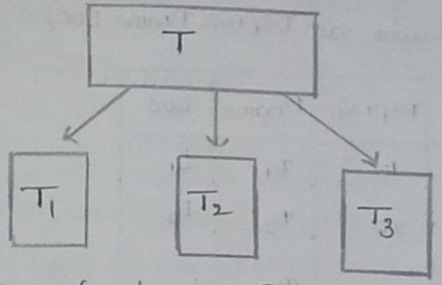
CID	Cname	Pincode	City	State
1	Smith	510001	Bangalore	Karnataka
2	Miller	510002	mumbai	Mah
3	Scott	510001	Bangalore	Karnataka
4	Adams	510001	Bangalore	Karnataka
5	Scott	510002	Mumbai	Maha

NORMALIZATION

The process of decomposing a large table into small table as known as Normalization.

(or)

Reducing a table to its Normal form is know as NORMALIZATION.



Length of NORMAL FORM

1. First Normal form (1NF)
2. Second Normal form (2NF)
3. Third Normal form (3NF)

Note :- if any Table/entry is reduced to 3NF, is then the table is said to the Normalized

1. First Normal form (1NF)
   * No duplicates record.
   * multivalued data should be NOT process.

Qspider

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

QID	Name	C1	C2	C3
1	A	Java		MT
2	B	Java	SQL	
3	C		SQL	MT

2. Second Normal form (2NF) :-
   * Table should be 1NF (Normal form)
   * Table should not have Paratial functional Dependency.

Employee (EID, Ename, Sal, DEPTNO, Dname, Loc)

Eid	Ename	Sal	DEPT NO	Dname	Loc
1	A	100	10	D_1	L_1
2	B	120	20	D_2	L_2
3	C	320	10	D_1	L_1
4	D	251	10	D_1	L_1

EId - Ename, Sal

DEPTNO → dname, Loc

(Eid, DEPTNO) → (Ename, Sal, Dname, Loc)

R_1 - (EID, Ename, Sal)

R_2 - (DEPT NO, Dname, Loc)

R1

Eid	Ename	Sal
1	A	100
2	B	120
3	C	320
4	D	251

R2

DEPTNO	dname	Loc
10	D_1	L_1
20	D_2	L_2

3. Third Normal from :-

* Table should be in 2NF
* Table should and have Transitive functional Dependancy

Employee :- (EID, Ename, Sal, comm, pincode, State, Country)

EID → Ename
Sal
COMM
PINCode → State

R_1 - {Eid, Ename, Country}

R_2 - {Pincode, State, Country}

Customer

CID	Cname	Pincode	City	State
1	Smith	510001	Bangalore	Karanataka
2	Milleq	510002	mumbai	maha
3	Scott	510001	Bangalore	karantaka
4	Adamn	510001	Bangalore	karantaka
5	Scott	510002	mumbai	maha.

Customer : (Cid, Cname, Pincode, City, State)

(Pk) Cid :- Cname
Pincode - City
State

R_1 - (Cid, Cname, Pincode)

R_2 - (Pincode, City, State)

R1

CID	Cname	Pincode
1	Smith	510001
2	milleq	510002
3	scott	510001
4	Adama	510001
5	Scott	510002

R2

Pincode	City	State.
510001	Bangalore	karanataka
510002	mumbai	maha