display (emp)

(16) WQD employee details only the designation should consist of "man" as a sub-string and they should be hired after blake hiredate. and they should be located in the dallas loc.

cond(1) (emp) cond(1) (emp) unknown cond(1) (dept)

```
Select *
from emp
where lower(job) like '%man%' and hiredate >
      (select hiredate
       from emp
       where lower(ename) = 'blake') and
      deptno in (select deptno
                 from dept
                 where lower(loc) = 'dallas');
```

Super Advance knowledge based

display (dept)

(15) WQD department details only the department name should consist of 'a' character and the [redacted] should be same as (scott salary.

cond(2) emp unknown (1 Q)

```
Select *
from dept
where lower(dname) like '%a%' and deptno in
      (select deptno
       from emp
       where sal in (select sal
                     from emp
                     where lower(ename) = 'scott'));
```

display (dept)

(16) WQD dname, location only the location should begans with 'n' char and the manager name should be same as (King employee numb

cond(1) (dept) cond(2) (emp) unknown (1 Q)

```
Select dname, loc
from dept
where lower(loc) like 'n%' and deptno in
        (select empno deptno
         from emp
         where ~~lower(ename) = 'king');~~
               mgr in (select empno
                       from emp
                       where lower(ename) = 'king'));
```

(16) ~~WAQD~~

(17) WAQD department/name only employee should be hired after ('king' hiredate.) → unknown
display(dept)
↳ cond① (emp)

```
Select dname
from dept
where deptno in (select deptno
                 from emp
                 where hiredate >
                 (select hiredate
                  from emp
                  where lower(ename) = 'king'));
```

display(dept)

(18) WAQD department/details of 2st max sal/ 2st highest salary
↳ cond① (emp)

```
Select ~~de~~ *
from dept
where deptno in (select deptno
                 from emp)
                 where sal = (select max(sal)
                              from emp));
```

Draw back :-
Sale can look multiple tables and we cannot fetch data from all the tables (only one table)

Note :-
Upto 10 Query's syllabus ends on Case 2.
From 11 to 15 Query's is out of syllabus only for knowledge purpose.
To built physically strong about Case 2.

Assignment

① WAQ department/name only the salary should be less than highest Commission.
display (dept)
cond① (emp)

```
Select dname
from dept
where deptno in (select deptno
                 from emp
                 where sal < (select max(comm)
                              from emp));
```

② WAQ department/details of 1st recent hiredate.
display (dept)
cond① (emp)

```
Select *
from dept
where deptno in (select deptno
                 from emp
                 where hiredate = (select max(hiredate)
                                   from emp));
```

display (emp)

③ WAQD employee details only the department number should be same as Smith, James department number and location should beganing second character should be 'a'.

cond(1)

cond(2) (dept)

```
Select *
from emp
where deptno in (Select deptno
                 from emp
                 where lower(ename) in ('smith', 'james'))
      and deptno in (Select deptno
                     from dept
                     where lower(loc) like '_a%'));
```

④ WAQD employee details only the annual salary should be less than king annual salary and the employee number should be same as blake, turner manager number. and the department name Consist of 's' char.

```
Select *
from emp
where (sal*12) < (Select (sal*12)
                  from emp
                  where lower(ename) = 'king' and(
      empno in (Select mgr
                from emp
                where lower(ename) in ('king', 'turner'))
      and deptno in (Select deptno
                     from dept
                     where lower(dname) like '%s%'));
```

Doubt

(5.) WAQD department details only employee design-ation should be same as Jones designation and they should be earning some commission.

known

```
Select *
from dept
where deptno in (Select deptno
                 from emp
                 where job in (Select job
                               from emp
                               where (lower(ename) = 'jones') and
                                     (comm is not null)))
```

(6) WAQD number of employees, min sal, max sal only employee should be located in New york chicago location.

display (emp)

↳ cond① (dept)

```
Select count(*), min(sal), max(sal)
from emp
where deptno in (Select deptno
                 from dept
                 where loc in (Select loc
                               from dept
                               where lower(loc) in ('new york', 'chicago')));
```

(7) WAQD department number wise, number of employees, total salary, total comm only the department name should not began with a character 'a' & display the output in ascending order based on total salary.

display (emp)

↳ cond①

```
Select deptno, count(*), sum(sal), sum(comm)
from emp
where deptno in (select deptno
from dept
where lower(dname) not like 'a%')
group by deptno
order by sum(sal);
        (or)
order by sum(sal) asc;
```

display (emp) wise

⑧ WAQ designation wise where recent hiredate, oldest hiredate only the manager number should be in a range of 7600 to 7900 and loc should not consist of 'new' as a substring.
cond①

cond① (dept)

```
Select job, max(hiredate), min(hiredate)
from emp
where mgr between 7600 and 7900 and
                deptno in (Select deptno
from dept
where lower(loc) not like '%new%')
group by job;
```

display

⑨ WAQ department wise no of employee only employee should be working for a department accounting, sales and atleast two clerks should be working in each department
cond①

```
Select deptno, count(*),
from emp
where deptno in (select deptno
                from dept
        where lower(dname) in('accounting', 'sales')
group by deptno
having count(*) >= 2;
```

display (dept)

10 WAQ department/details only the employees sho-uld be working for some department number.
↳ card① (emp)

```
Select *
from dept
where deptno in (select deptno
                 from emp
                 where deptno is not null);
display (dept)
```

11. WAQ department/details only the employees not working for any of the department number.

```
Select *
from dept
where deptno in (select deptno
                 from emp
                 where deptno is null);
display (dept)
```

⑫ WAQ Department/name of 1st highest Commission.

```
Select dname
from dept
where deptno in (select deptno
                 from emp
                 where comm = (select max(comm)
                               from emp));
```

11/8/21

Case 3 :-

Case 3.1 :- To Determine nth max & nth min

EMP	
SAL	
4000	---> 3
8000	---> 2
3000	---> 4
9000	---> 1
2000	---> 5

Note :-

Q) All sal = 1st max (sal)
A) 1st max (sal)

Q) All Sal < 1st max (sal)
↓
2nd max (sal)

WQD 2nd max salary.

```
Select max (sal)          -> 2nd max -> "outer Query"
from emp --> 1
where sal < (select max(sal) --> 2    "Inner Query" -> 1st max, "9000"
             from emp); --> 1
```

Sal < 9000

4000 < 9000 ---> T
8000 < 9000 ---> T
3000 < 9000 ---> T
~~9000 < 9000 ---> F~~
2000 < 9000 ---> T

("According to Condition these 4 records should be retain & Select clau-se will pic record bas-ed on "Condition")

O/P :-

MAX(SAL
8000

WQD 2nd Min Sal

```
Select min(sal)
from emp
where sal > (select min(sal)
             from emp);
```

(3) WAQ 3rd maximum Salary.

```
Select max(sal)
from emp
where sal < (select max(sal)
             from emp
             where sal < (select max(sal)
                          from emp));
```

Note:-

* How many times they are asking about a query. that many times should be present in query.
* According to above query.
  If they ask 3rd max sal. they one max of sal present in outer query and two max sal present in inner query.
* Likewise if they ask 1000.
  then 999 inner Query & 1 outer Query.

Draw back:-

To determine nth max or nth min value we should write n-1 inner Query. So query length will get increase and perfor-mance will get decrease.

(4) WAQ second min salary.

```
Select * min(sal)
from emp
where sal > (select min(sal)
             from emp);
```

Note:-

Max value ---> max() ---> <

min value ---> min() ---> >

Assignment

1 WAQ 2nd minimum Commission.

```
Select min(comm)
from emp
where comm > (select min(comm)
              from emp);
```

2 WAQ 3rd max Comm

```
Select max(comm)
from emp
where comm < (select max(comm)
              from emp
              where comm < (select max(comm)
                            from emp));
```

3. WAQ 3rd recent hiredate

```
Select max(hiredate)
from emp
where hiredate < (select max(hiredate)
                  from emp
                  where hiredate <
                  (select max(hiredate)
                   from emp));
```

4. WAQ 2nd max annual Salary

```
Select max(sal*12)
from emp
where (sal*12) < (select max(sal*12)
                  from emp);
```

5 WAQ 4th min Annual Commission

```
Select min(comm*12)
from emp
where (comm*12) >= (select min(comm*12)
from emp
where (comm*12) > (select min(comm*12)
from emp
where (comm*12) > (select min(comm*12)
from emp)));
```

~~where (comm*12))))~~

6 WAQ 3rd oldest hiredate

```
Select min(hiredate)
from emp
where hiredate > (select min(hiredate)
from emp
where hiredate > (select min(hiredate)
                  from emp));
```

7 WAQ 4th min new salary with 1000rs increment

```
Select min(sal+1000) New_min
from emp
where min(sal+1000) > (select min(sal+1000) new_min
from emp
where min(sal+1000) > (select min(sal+1000)
from emp                                    new_min
where min(sal+1000) > (select min(sal+1000)
                       from emp)));          new_min
```

8 WAQD 2nd min new Comm with 50% Increment

```
Select min(comm * 1.5) New_Comm
from emp
where (comm*12) > (select min(comm * 1.5) New_comm
                   from emp);
```

9 WAQD 2nd recent hiredate after one year.

```
Select max(hiredate + 365)
from emp
where(hiredate + 365) < (select max(hiredate + 365)
                         from emp);
```

Note:-
- Alias-name will write in both outer query and Inner query.

10 WAQD 4th min Annual salary.

```
Select min(sal * 12)
from emp
where (sal*12) > (select min(sal*12)
from emp
where (sal*12) > (select min(sal*12)
from emp
where (sal*12) > (select min(sal*12)
                  from emp)));
```

Assignment done

Case:- 3 syllabus Complete.

Advance Query's purpose
knowledge

(1) WAQD employee name, date of joining, salary of 3rd oldest hiredate
display

```
Select ename, hiredate, sal
 from emp
where hiredate = (select min(hiredate)
from emp
where hiredate > (select min(hiredate)
from emp
where hiredate > (select min(hiredate)
from emp
where hiredate > (select min(hiredate)
from emp ))));
```

2 WAQD emp details of second max sal.

```
Select *
from emp
where sal = (select max(sal)
             from emp
             where sal < (select max(sal)
                          from emp));
```

3. WAQD dname, loc, 2nd max annual salary.

```
Select dname, loc
from dept
where deptno in (select deptno
from emp
where (sal*12) = (select max(sal*12)
from emp
where (sal*12) < (select max(sal*12)
                  from emp)));
```

4 WAQD employee name, designation, department number of 2nd min Comm.

```
Select ename, job, deptno
from emp
where comm = (select min(comm)
from emp
where comm > (select min(comm)
              from emp));
```

5 WAQD department details of 3rd minimum annual Commission

```
Select *
from dept
where deptno in (select deptno
                 from emp
                 where (comm * 12) = (select min(comm*
from emp
where (comm*12) > (select min(comm*12)
from emp
where (comm*12) > (select min(comm*12)
from emp)));
```

6 WAQD the employee details only the salary should be more than second min comm.

```
Select *
from emp
where sal > (select min(comm)
             from emp
             where comm > (select min(comm)
                           from emp));
```

Out of syllabus
Knowledge purpose
No Interview questions

(No need to study for interview)

Case 3.2 :-

Case

Mapping values present in Same table but in different rows.

	Emp		
empno	ename	Job	mgr
1	Mr. Girish Sir	CEO	
2	Mr. Pradeep Sir	VP	1
3	Mr. Nisha mam	Sr. QA	2
4	Mr. Pavan Sir	Verbal	2
5	Mr. Keerthi Sir	Dev	2

Note:-

Display Condition

(mgr_name) (ename)

empno = f(mgr)

① WAQ mgr_name of an employee Mr. Keerthi Sir

disp (mgr_name) ~~outer query~~ cond (ename)

```
Select ename as mgr_name --> 3
from emp --> 1
where empno in (select mgr --> 3      "Inner Query"
                                "2"
                from emp --> 1
                where lower(ename) = 'mr. keerthi sir');
                                  --> 2
```

empno = 2

1 = 2 → F

2 = 2 → T

3 = 2 → F

4 = 2 → F

5 = 2 → F

mr. girish sir = Mr. Keerthi Sir → F

Mr. pradeep sir = Mr. Keerthi sir → F

Mr Nisha mam = Mr. Keerthi Sir → F

Mr. pavan sir = Mr. Keerthi Sir → F

Mr. Keerthi Sir = Mr. Keerthi Sir → T

O/P:- mgr_name

Mr. pradeep Sir.

② WAQ mgr_name of an employee Smith.

```
Select ename mgr_name
from emp
where empno in (Select mgr
                from emp
                where lower(ename) = 'smith');
```

3 WAQ mgr_name, mgr_designation, mgr_deptno of an employee blake, miller.

```
Select ename as mgr_name, job, deptno
from emp
where empno in (Select mgr
                from emp
                where lower(ename) in ('blake', 'miller'));
```

4 WAQ manager details of an employee James.

```
Select *
from emp
where empno in (Select mgr
                from emp
                where lower(ename) = 'james');
```

5 WAQ department details of turner manager.

```
Select *
from dept
where deptno in (Select deptno
from emp
where empno in (Select mgr
from emp
where lower(ename) = 'turner'));
```

6 WAQ manager manager of a employee Mr. pavan Sir.

```
Select ename mgr_name
from emp
where empno in (select mgr
from emp
where empno in (select mgr
from emp
where lower(ename) = 'mr. pavan sir'));
```

7 WAQ manager manager manager of an employee Adams

```
Select ename mgr_name
from emp
where empno in (select mgr
from emp
where empno in (select mgr
from emp
where empno in (select mgr
from emp
where lower(ename) = 'adams')));
```

8 WAQ department details of miller manager manager.

```
Select *
from dept
where deptno in (select mgr
from emp
where empno in (select mgr
from emp
where empno in (select mgr
from emp
where lower(ename) = 'miller')));
```

To Display for employee

① WAQD ename (display) only the employee those who are reporting manager mr. pradeep sir. (cond (mgr_name))

```
Select ename --> 3
from emp --> 1
where mgr in (select deptno --> 3
              from emp ---> 1
              where lower(ename) = 'mr. pradeep sir') ;
                                                ---> 2
```

mgr = 2

null = 2 --> F

1 = 2 --> F Mr. girish sir = Mr. pradeep sir --> F

2 = 2 --> T mr. pradeep sir = mr. pradeep sir --> T

2 = 2 --> T mr. keerthi sir = mr. pradeep sir --> F

2 = 2 --> T mr. nisha mam = mr. pradeep sir --> F

mr. pavan sir = mr. pradeep sir --> F

Op:-

Disp	Cond
ename	mgr_name
OQ	IQ
mgr	= empno (mgr_name)

Op:- ename

Mr. keerthi sir

Mr. Nisha mam

Mr. pavan sir.

(2) WAQ employee name, designation, department number (display) those who are reporting to a manager blake

```
Select ename, job, deptno
from emp
where mgr in (select empno
from emp
where lower(ename) = 'blake');
```

(3) WAQ employee details those are reporting to a manager king, james.

```
Select *
from emp
where mgr in (select empno
from emp
where lower(ename) in ('king', 'james'));
```

(4) WAQ department name only the employees those are reporting a manager miller.

```
Select dname
from dept
where deptno in (select deptno
                 from emp
                 where mgr in (select empno
                               from emp
                               where lower(ename) = 'miller'));
```

Draw back:-
we can display empno as mgr but we cannot write both empno & mgr at same time.
→ To overcome SELF JOIN'S

Drawback's of Sub-query :-

Case 2 :-

* We can link multiple tables but we can't fetch data from all the tables.
  → TO overcome joins.

Case 3 :-

* To determine nth max and nth min value we should write n-1 inner query. So query length will get increase and performance will get decrease.
  → To overcome Core-related Sub-query

13/08/21 No interview Questions knowledge purpose

ANY AND ALL OPERATOR

ANY :-

* ANY is the multi valued operator.
* ANY operator will accept single value in LHS side and multi values in RHS side
* ANY operator will compare single LHS value with all the value present in RHS side.
* ANY operator will return if any one of the condition or comparision is TRUE.
* ANY operator will return FALSE if all the condition or comparision is FALSE.
* ANY operator is working similar to OR operator

Syntax :-

where LHS_value relational_operator RHS_value

↓ C.n/expr

↓ >ANY <ANY >=ANY <=ANY =ANY !=ANY

↓ (values 1, value 2 ....)

example 2 :-

90 > ANY (80, 100)

90 > 80 ---> T
90 > 100 ---> F
} --> T

example :- 2

100 > ANY (200, 300)

100 > 200 ---> F
100 > 300 ---> F
} F

Note :-

* IN and ANY operator both are different because IN will perform equal to (=) comparision and ANY will perform all relational operator comparision.
* IN and =ANY both are Same:
because both will perform equal to (=) comparision.

ALL :-

* ALL is the multi-valued operator.
* All operator will accept single value in LHS-side and multiple-values in RHS-side.
* All operator will compare single LHS-value with all the values present in RHS-side.
* All operator will written (pass) TRUE when all the condition or comparision is TRUE
* All operator will written (pass) FALSE if any one of the condition or comparision is FALSE
* All operator will working similar to AND operator.

Syntax :-

where LHS-value relational operator RHS-value

↓ C.n/expr ↓ >ALL <ALL >=ALL <=ALL =ALL !=ALL ↓ (V1, V2, V3, ...)

Example-1

90 > ALL (50, 100)

90 > 50 ---> T
90 > 100 ---> F
} F

example-2:-

100 > ALL (200, 300)

100 > 200 ---> F
100 > 300 ---> F
} F

Example-3:

90 > ALL (80, 60)

90 > 80 ---> T
90 > 60 ---> T
} T

Note:-

* IN and ALL both are different because IN will perform (=) Comparision & ALL will perform ALL relational operator Comparision.
* IN and =ALL both are different because IN will written TRUE if any Comparision is TRUE. but ALL will written TRUE if all the Comparision is TRUE.

When to go for "ANY & ALL" operator:-

* ONLY in Sub-query when we have Single LHS and multiple RHS and we should perform (<, >, <=, >=) comparision then we should go any and ALL operator.
* writing ANY & ALL operators is depends on question.

1 WAQD employee details only the salary should be more than any of miller, king salary.

```
Select *
from emp
where sal > any(select sal
                from emp
                where lower(ename) in ('miller', 'king'));
```

2 WAQD employee details only salary should be less than ALL james, martin salary.

```
Select *
from emp
where sal < all (select sal
                 from emp
                 where lower(ename) in('james','martin');
```

3 WAQD employee details salary should be more than ALL blake salary.

```
Select *
from emp
where sal >all (select sal
                from emp
                where lower(ename) = 'blake');
```

JOINS

* Joins is a "Special possibility" of SELECT Statement / clause
* By using Joins "we can fetch multiple tables information simultaneously" is called as JOINS.
* According to oracle RDBMS JOINS are classified into 5 types.
  - → Cartesian Join / Cross Join
  - → Inner Join / Equvi Join / Natural Join (Imp)
  - → Non - Equvi Join
  - → Self Join
  - → Outer Join
    - a) Left outer Join / Left Join
    - b) Right outer Join / Right Join
    - c) Full outer Join / Full Join

15/08/21

1) Cartesian Join / Cross Join :-

* Cartesian Join is a Cross product of multiple tables.
* Cartesian Join is working on a principle of "Cross product".

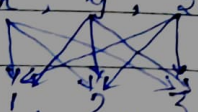
Cross product :-

* Value from one table with all the values present in another table is called as Cross product

example :-

A = { x, y, z }

B = { 1, 2, 3 }



A x B = { x1, x2, x3 }
{ y1, y2, y3 }
{ z1, z2, z3 }

* Cartesian join will result in both "valid and invalid records".
* Cartesian join we will not specify any "joining Condition"

Oracle Syntax:-

```
Select */c.n's
from table1, table2
where <cond's> ;
```

ANSI Syntax:-

```
Select */c.n's
from table1 Cross Join table2
where <cond's> ;
```

Example:

	emp:				dept	
empno	ename	deptno		deptno	dname	loc
1	Dia	10	→	10	Accounting	NEW YORK
2	Dolly	30	→	20	research	dallas
3	Bachu	20	→	30	sales	chicago
4	Dingi	10	→	40	operations	boston
5	Shashi	20				

WAQD employee name, department name of all the employees

disp(emp) / display(dept)

Oracle syntax:-

```
Select ename, dname --> 2
from emp, dept; --> 1
```

ANSI syntax:-

```
Select ename, dname
from emp Cross Join dept;
```

o/p ename	dname	
Dia	Accounting	→ Valid record
Dia	research	→ Invalid record
Dia	Sales	→ Invalid record
Dia	operations	→ Invalid record
Dolly	accounting	→ Invalid record
Dolly	research	→ Invalid record
Dolly	Sales	→ valid record.
Dolly	operations	
⋮	⋮	
~~20~~		

5 × 4 = 20 records as o/p

5 records valid — 15 records invalid

Note:- (Draw backs)

→ To overcome joins.

* Cartesion join will result in both valid and invalid record because we are not specifing "Joining Condition".
* Cartesion join will give result in both "Valid and invalid record," Compare to valid records, invalid records are more hence it is not a preffered type of join.
* Cartesion join can applicable any kind of table "with a Common c.n (or) without a Common c.n"

Ex:- SQL > select ename, grade
from emp, salgrade ; → without Common c.n

SQL > Select ename, dname, grade
from emp, dept, salgrade ; → Joining ×3 tables or 'n' tables.

SQL - INNER JOINS

(10 Crore)

2 INNER JOIN / EQUVI JOIN / NATURAL JOIN :-

* Joining the multiple tables by 'specifying joining Condition' between the tables is called as Inner Join

(or)

Joining the multiple tables by "utilizing Common Column" between tables is called as inner Join.

* Inner Join will result in only valid record
* Inner Join is applicable only a table consis -st of Common c.n
* In inner Join we will Specify Joining Cond.

Joining Condition :-

Joining the multiple tables by utilizing com -mon c.n between tables is called as Joining Condition.

SYNTAX :- where table1.cn = table2.cn

cn --> Common column.

oracle Syntax :-

Select */c.n's
from table1, table2
where <table1.cn = table2.cn> and <cond's> ;

Normal conditions ↑

ANSI Syntax :-

Select */'c.n's
from table1 [inner] Join table2
on <table1.cn = table2.cn>
where <cond's> ;

example:-

emp

empno	ename	deptno
1	Dia	10
2	Dolly	30
3	Dachu	20
4	Dingi	10
5	Shashi	20

dept

deptno	dname	loc
10	Accounting	New York
20	Research	dallas
30	Sales	Chicago
40	operations	boston

WAQ employee name, department name of all the employees. display(emp) ; display(dept)

Oracle syntax

```
Select ename, dname
from emp, dept
where emp.deptno = dept.deptno ;
                   ↳ Joining Condition.
```

ANSI Syntax:-

```
Select ename, dname
from emp inner join dept
on emp.deptno = dept.deptno ;
                ↳ Joining Condition.
```

10 = 10 → T 30 = 10 → F 20 = 10 → F
10 = 20 → F 30 = 20 → F 20 = 20 → T
10 = 30 → F 30 = 30 → T 20 = 30 → F
10 = 40 → F 30 = 40 → F 20 = 40 → F

10 = 10 → T 20 = 10 → F
10 = 20 → F 20 = 20 → T
10 = 30 → F 20 = 30 → F
10 = 40 → F 20 = 40 → F

o/p:-

ename	dname
Dia	Accounting
Dolly	Sales
Dachu	research
Dingi	accounting
Shashi	research

6/01/21

2) WAQD employee name, designation, manager number, dname deptno, loc only the manager number should be in a range 7600 to 7900.
(display(emp), display(dept), cond)

```
Select ename, job, mgr, dname, loc
from emp, dept
where emp.deptno = dept.deptno and
      (mgr between 7600 and 7900);
```

Note:-
when display from multiple tables we will go with joins. (Inner join)
(or)

```
Select ename, job, mgr, dname, loc
from emp inner join dept
on emp.deptno = dept.deptno
where mgr between 7600 and 7900;
```

3. WAQD employee name, date of joining, annual Salary department name only the location should consist of new as a sub-string:
(display(emp), display(dept), cond)

```
Select ename, hiredate, (sal*12) ann_sal, dname
from emp, dept
where emp.deptno = dept.deptno and
      (lower(loc) like '%new%');
```

4. WAQD employee name, empno, Commission, department name, loc only : the employees should be earning some Commission.

```
Select ename, empno, comm, dname, loc
from emp, dept
where emp.deptno = dept.deptno and
                    (comm is not null);
```

5 WAQD employee (display(emp)) name, salary, new salary with 1000rs increment, new Commission with 500rs decrement, dname (display(dept)) only employee should be working as salesman, clerk. (condition)

```
Select ename, sal, (sal+1000) New_sal, (comm-500) new_comm,
                                                    dname
from emp, dept
where emp.deptno = dept.deptno and (lower(job) in ('salesman', 'clerk'));
```

6 WAQD employee (display(emp)) name, employee number, designation, department name, loc (display(dept)) only. the salary should be more than 500 (cond①) and the department name should be exactly 5 char. (cond②)

```
Select ename, empno, Job, dname, loc
from emp, dept
where emp.deptno = dept.deptno and
        (sal > 500) and (lower(dname) like '_____');
```

display (emp)

7 WAQ employee name, salary, commission, department details only the employee name should consist of 'a' char.

```
Select ename, sal, comm, dept.*
from emp, dept
where emp.deptno = dept.deptno and
                (lower(ename) like '%a%');
```

Note:-

when ever we are writing combination of C.n and * (meta char) then it is better to write |table_name.*|

display (emp) display (dept)

8 WAQ employee details, department name, location only employee should be located in chicago location and display the output in descending order based on salary.

```
Select emp.*, dname, loc
from emp, dept
where emp.deptno = dept.deptno and
                lower(loc) = 'chicago'
Order by sal desc;
```

display (emp) display (dept)

9 WAQ employee details, annual salary, department name, only employee name & department name should consist of 'e' char.

```
Select emp.*, sal*12 ann_sal, dname
from emp, dept
where emp.deptno = dept.deptno and
    (lower(ename) like '%e%') and
                (lower(dname) like '%e%');
```

10 WAQD employee details, department details only the employee should be working foor a department number 10, 20

Select emp.*, dept.* (or) Select *
from emp, dept
where emp.deptno = dept.deptno and
emp.deptno in (10,20);
(or)
dept.deptno in (10,20);

Note:-
when ever we are writing common column as a condition in "where clause" and display in "select clause" it is better to write /table-name.common column/.

11. WAQD ename, empno, job, deptno, dname only employee should be hired on year 81.

Select ename, empno, job, emp.deptno, dname
(or) " " , " , " , dept.deptno, "
from emp, dept
where emp.deptno = dept.deptno
and hiredate like '%81';

12 WAQD employee name, date of joining, salary, department details only employee should not be working foor a department number 40.

Select ename, hiredate, sal, dept.*
from emp, dept
where emp.deptno = dept.deptno and
emp.deptno != 40;

13 WAQ employee details, department name only employee should be earning some sal and display the output in ascending order based on department number.

```
Select emp.*, dname
from emp, dept
where emp.deptno = dept.deptno
                and (sal is not null)
order by emp.deptno ;
```

Assignment

display(emp) display(dept)

1 WAQ employee details, department details only employee should be working for a deptno accounting, research (condition)

```
Select emp.*, dept.*
from emp, dept
where emp.deptno = dept.deptno and
      deptno in(select deptno from dept
      where lower(dname) in('accounting', 'research');
```

2. WAQ emp details, annual salary, dept details only the department name should consist of two 'a' char and display the output in ascending order based on annual salary.

```
Select emp.*, sal*12 ann_sal, dept.*
from emp, dept
where emp.deptno = dept.deptno and
      lower(dname) like '%a%a%'
order by ann_sal ;
```

V. Imp

3 WAQ emp details only employee should be located in chicago, New york location

```
Select emp.*
from emp, dept
where emp.deptno = dept.deptno and
                    lower(loc) in ('chicago', 'new york')
```

Display multiple table → Inner Join
Display one table & cond from another table → Inner Join

4 WAQ dept details only employee should be hiredate after a year 80.

```
Select *
from emp, dept
where emp.deptno = dept.deptno and

        hiredate >= '01-JAN-81');
```

5 WAQ employee name, employee number, salary department details only the designation should be same as miller designation

```
Select ename, empno, sal, dept.*
from emp, dept
where emp.deptno = dept.deptno and job
    in (select job
        from emp
        where lower(ename) = 'miller');
```

6. WAQ department wise number of employees, min salary, max salary only the employee should not be working for a department operations and their salary should be four digit.

```
Select emp.deptno, count(*), min(sal), max(sal)
from emp, dept
where emp.deptno = dept.deptno and
      lower(dname) != 'operations') and
      sal like '____')
group by emp.deptno;
```

7 WAQ designation wise, recent hiredate, oldest hiredate only employee should be located in dallas, chicago and atleast two employees should be present in each designation.

```
Select job, max(hiredate), min(hiredate)
from emp, dept
where emp.deptno = dept.deptno and
      lower(loc) in ('dallas', 'chicago'))
group by job
having count(*) >= 2;
```

8 WAQ employee name, designation, salary, dname, location of thirst max salary.

```
Select ename, job, sal, dname, loc
from emp, dept
where emp.deptno = dept.deptno and
      sal = (select max(sal)
             from emp);
```

17/08/21

Joining more than two tables :-

Oracle Syntax :- (to join two tables)	ANSI Syntax (To join two tables)
Select */c.n's from table 1, table 2 where <table.1.cn = tabel2.c.n> and <cond's>	Select */c.n's from table1 inner join table 2 on <table.1.cn = table2.c.n> where <cond's>;

Oracle Syntax :- (To join more than 2 tables)	ANSI Syntax :- (To join more than two tables)
Select */c.n's from table 1, table 2, table 3, table 4.. where <table 1.cn = table 2.cn> and <table2.cn = table3.cn> and <table 3.cn = table4.cn> and..... and <cond's>	Select */c.n's from table1 inner join table 2 on table 1.cn = table 2.cn inner join table 3 on table2.cn = table 3.cn inner join table 4 on table3.cn = table 4.cn where <cond's>;

~~Ore~~ Oracle Syntax :-

In oracle Syntax to join 'n' number of tables we will write 'n' number of table names in single "from clause" and (n-1) joining condition's in single "where clause"

ANSI Syntax :-

In ANSI Syntax to join 'n' number of tables we should write 'n' number of table name in (n-1) FROM clause and (n-1) joining conditions in (n-1) ON key word.

1 WAD region_name, Country_name, City name of all the locations.

oracle syntax:-

Select region_name, Country_id, City
from regions, Countries, locations
where regions.region_id = Countries.~~Count~~ region_id
and Countries.Country_id = locations.Country_id ;

Table aliasing:-

Select region_name, Country_name, City
from regions r, Countries c, locations l
where r.regiad_id = c.region_id and
c.country_id = l.Country_id ;

ANSI Syntax:-

Select region_name, Country_name, City
from regions r inner Join Countries C
on r.region_id = c.region_id
inner Join locations l
on c.Country_id = l.Country_id ;

Note:-

when ever we are writing table aliasing in a query throughout the query Same table aliasing should be Consider. that is in Select Clause and where Clause.

2 WAQD regions details, Country_name, location_id, Street_address only the Country_name should be india.

```
Select r.*, Country_name, location_id, street_address
from regions r, Countries c, locations l
where r.regions_id = c.regions_id and
      c.country_id = l.country_id and
lower (country_name) = 'india' ;
```

3 WAQD region_name, Country details, location_id Street address, postal code only for a Country_id IN, US, UK

```
Select region_name, c.*, location_id, street_address,
                                    postal_code
from regions r, Countries c, locations l
where r.region_id = c.region_id and
      c.country_id = l.Country_id and
lower(c.country_id) in ('in', 'us', 'uk');
```

4 WAQD region_name, Country_name, location details only the region_id should be 2, 3

```
Select region_name, Country_name, l.*
from regions r, Countries c, locations l
where r.region_id = c.region_id and
      c.country_id = l.location_id and
      r.region_id in (2, 3) ;
```

Assignment

1 WAQD region_name, country name, location details only the country id IN, US

```
Select region_name, country_name, l.*
from regions r, countries c, locations l
where r.region_id = c.region_id and
      c.country_id = l.country_id and
      country_id) in ('in', 'us');
```

2 WAQD region_name, country_details, street address, postal code, city name only for the region_id 2, 3

```
Select region_name, c.*, street address, postal_Code,
                                                City
from regions r, countries c, locations l
where r.region_id = c.region_id and
      c.country_id = l.country_id and
      r.region_id in (2,3);
```

3 WAQD region details, country_id, location_id, postal code only for the city Bombay, Tokyo and display the output in descending order based on locationid

```
Select r.*, c.country_id, location_id, postal_Code
from regions r, countries c, locations l
where r.region_id = c.region_id and
      c.country_id = l.country_id and
      lower(city) in ('bombay', 'tokyo')
order by location_id desc;
```

4 WAQD region details, Country details, location details except for the Country india, china and their street address should Consist of 'c' char

```
Select r.*, c.*, l.*
from regions r, Countries c, locations l
where r.region_id = c.region_id and
      c.country_id = l.country_id and
not lower(City) in('india', 'china') and
      lower (street_address) like '%a%' ;
```

When to go Inner Join :-

→ when ever we want to display from multiple table.

→ when display one table and condition ano-ther table. (through Joining Condition)

Why Inner Join is called as equvi Join ?

* Joining the multiple tables by specifying equal to (=) Comparision in Joining Condition. So Inner Join is known as equvi Join

Ex :-

where emp.deptno (=) dept.deptno

↓

Equal to (=) Comparision

(Not important for interview)

3. Non-Equvi Join :-

→ Joining the multiple tables when there is no common column b/w the tables

→ (or)

→ Joining the multiple tables without specifying equal to (=) operator in joining condition is called as non-equvi join.

→ In non-equvi join, joining condition except equal to (=) operator will specify other all relational operator (<, >, <=, >=, !=)

When to go Non-equvi Join :-

Case 1 :-

When there is no common column between the tables then we should go Non-equvi Join.

Case 2 :-

If there is no common c.n,

1 W/Q n employee name, salary, grade only the salary should be in a orange of low sal and high sal.

```
Select ename, sal, grade
from emp, dept
where sal between losal and highsal;
```

(Joining cond with relevant column)

display (emp)

2 WAQ employee name, designation, salary, salgrade details only the salary should be more than low sal.

display (salgrade)

↳ condition

```
Select ename, Job, sal, s.*
from emp, salgrade s
where sal > losal ;
```

display (emp) display (salgrade)

3. WAQ employee details, low sal, high sal only the commission should be in a range of lowsal and high sal and they should be working for a deptno 30.

```
Select emp.*, losal, hisal
from emp, salgrade
where (comm between losal and hisal)
      and deptno = 30 ;
```

Joining condition (with relevant c.n)

c.n condition

4 WAQ emp details, salgrade details only

```
Select *
from emp, salgrade
where sal != hisal and lower(job) = 'salesman';
```

(No Interview Qustn- Not important)

5 outer Join :-

→ Joining the multiple tables by utilizing common column between the tables is called as outer Join

→ Outer Join is Combination of inner Join and invalid records.

| Outer Join = inner Join (valid records) + invalid records |

→ outer Join will result in both Valid and invali-d records.

→ outer Join will result Valid records from all the tables and invalid from Specific table.

→ based on invalid records outer Join are classi-fied into three types.

a) Left outer Join / Left Join
b) Right outer Join / Right Join
c) full outer Join / full Join.

Left outer Join / Left Join :-

It is a Combination of inner Join and a record from Left table which does not have any matching pair in right table is called as

(left outer Join (LOJ)
(OR)

Valid record from all the table and invalid record from left table is called as Left outer Join (LOJ)

Oracle syntax:-

```
Select */c.n's
from table1, table2
where table1.cn = table2.cn(+) ;
```

ANSI syntax:-

```
Select */c.n's
from table1 Left [outer] Join table2.
on table1.cn = table2.cn
where <cond's> ;
```

Ex: WAQD employee name, department name only the employee not working for any of the department.

	emp	
empno	ename	deptno
1	Dia	10
2	Dolly	30
3	Dachu	20
4	Dingi	10
5	Shashi	

	dept	
deptno	dname	loc
10	Accounting	New York
20	research	dallas
30	Sales	Chicago
40	operations	boston

Oracle syntax:-

```
Select ename, dname ---> 3
from emp, dept ----> 1
where emp.deptno = dept.deptno(+) ; --> 2
```

ANSI Syntax:-

```
Select ename, dname
from emp left [outer] Join dept
on emp.deptno = dept.deptno;
```

Verification

10 = 10 --> T	30 = 10 --> F	20 = 10 --> F
10 = 20 --> F	30 = 20 --> F	20 = 20 --> T
10 = 30 --> F	30 = 30 --> F	20 = 30 --> F
10 = 40 --> F	30 = 40 --> F	20 = 40 --> F

10 = 10 --> T	null = 10 --> F
10 = 20 --> F	null = 20 --> F
10 = 30 --> F	null = 30 --> F
10 = 40 --> F	null = 40 --> F

o/p :-

ename	dname	
Dia	accounting	Valid record (Inner Join)
Dolly	Sales	
Dachu	research	
Dingi	accounting	
Shashi		Invalid record (left table)

Right outer Join :-

-> It is a Combination of Inner Join and a record from right table which does not have any matching pair in left table is called as Right outer Join.

(or)

-> valid record from all the table and invalid record from the right table is called as right outer Join.

Oracle Syntax :-

```
Select */cn's
from table 1, table 2
where table 1. cn (+) = table 2. cn;
```

ANSI Syntax :-

Select */c.n's
from table 1 Right outer Join table 2.
on table 1. cn = table 2. cn
where < cond's > ;

Ex :- W/QD employee name, department name only for the department number employees are working.

	emp				dept	
empno	ename	deptno		deptno	dname	loc
1	Dia	10		10	account -ing	New York
2	Dolly	30		20	research	dallas
3	Dachu	20		30	sales	chicago
4.	Dingi	10		40	operat -ions	boston
5	shashi					

Oracle Syntax :-

Select ename, dname ---> 3
from emp, dept ---> 1
where table 1. cn(+) = table 2. cn ---> 2

ANSI Syntax :-

Select ename, dname
from emp right outer Join dept
on emp. deptno = dept. deptno
where < cond's > ;

10 = 10 --> T 20 = 10 --> F 30 = 10 --> F 40 = 10 --> F
10 = 30 --> F 20 = 30 --> F 30 = 30 --> T 40 = 30 --> F
10 = 20 --> F 20 = 20 --> T 30 = 20 --> F 40 = 20 --> F
10 = null --> F 20 = null --> F 30 = 10 --> F 40 = 10 --> F
10 = null --> F 20 = null --> F 30 = null --> F 40 = null --> F

o/p :-

ename	dname	
Dia	accounting	Valid record (Inner Join)
Dolly	Sales	
Dachu	research	
Dingi	accounting	
	operations	Invalid record (Right table)

Full outer Join :-

→ It is a combination of left outer join and right outer join is called as full outer join.

(or)

→ Valid records from all the tables and invalid records from all the tables. it is called as Full outer join.

Oracle Syntax :-

```
select */c.n's
from table 1, table 2
where table 1.cn = table 2.cn (+)
union
select */c.n's
from table 1, table 2
where table 1.cn (+) = table 2.cn ;
```

ANSI Syntax :-

```
Select */c.n's
from table 1 full [outer] join table 2
on table 1.cn = table 2.cn
where < cond's > ;
```

Ex:- WAQ employee name, department name only the employee not working any of the department and only for the department number employees are working.

Oracle Syntax:-

```
Select ename, dname
from emp, dept
where emp.deptno = dept.deptno (+)
union
Select ename, d.name
from emp, dept
where emp.deptno (+) = dept.deptno ;
```

ANSI Syntax:-

```
select ename, dname
from emp full [outer] join dept
on emp.deptno = dept.deptno ;
```

o/p:-

ename	dname	
Dia	accounting	↑
Dolly	sales	Valid record (inner join)
Dachu	research	
Dingi	accounting	↓
shashi		* Invalid record (Left table)
	operations	* Invalid record (Right table)

5 Self Join :-

→ Joining same table to itself is called as a self join.

→ except self join in other all types of joins we will specify more than one table-name. but in self join we join same table to itself.

→ based on joining condition self join as classified into 2 cases.

Case 1 :-

Joining same table by utilizing same column.

```
Select * / c.n's
from table1 a, table1 b
where a.c1 = b.c1 ;
```

Case 2 :-

" Joining same table by utilizing different c.n's

```
Select * / c.n's
from table1 a, table1 b
where a.c1 = b.c2 ;
```

Case 2

Ex :- WAP ename, mgr-name of all the employees.

emp(a)

empno	ename	Job	mgr
1	Mr. Girish Sir	CEO	
2	Mr. pradeep Sir	Vp	1
3	Mr. Keerthi Sir	QA	2
4	Ms. Nisha mam	VERBAL	2
5	Mr. Tanush Sir	DEV	2

emp(b)

empno	ename	Job	mgr
1	Mr. Girish Sir	CEO	
2	Mr. Pradeep Sir	VP	1
3	Mr. keerthi Sir	QA	2
4	Ms. Nisha mam	VERBAL	2
5	Mr. Tanush Sir	DEV	2

Select a.ename, b.ename as mgr_name ---> 3
from emp a, emp b ---> 1
where a.mgr = b.empno; ---> 2
↳ Joining Cond (case 2)

· Verification

null = 1 ---> F	1 = 1 ---> T	2 = 1 ---> F
null = 2 ---> F	1 = 2 ---> F	2 = 2 ---> T
null = 3 ---> F	1 = 3 ---> F	2 = 3 ---> F
null = 4 ---> F	1 = 4 ---> F	2 = 4 ---> F
null = 5 ---> F	1 = 5 ---> F	2 = 5 ---> F

2 = 1 ---> F	2 = 1 ---> F
2 = 2 ---> T	2 = 2 ---> T
2 = 3 ---> F	2 = 3 ---> F
2 = 4 ---> F	2 = 4 ---> F
2 = 5 ---> F	2 = 5 ---> F

O/P:-

ename	mgr_name
Mr. pradeep sir	Mr. Girish Sir
Mr. Keerthi Sir	Mr. pradeep sir
Ms. Alisha mam	Mr. pradeep Sir
Mr. Tanush Sir	Mr. pradeep Sir

2 WAQ employee name, employee designation, employee salary, manager name, manager designation, manager salary only employee should be working as Salesman, Clerk.

Select a.ename, a.Job, a.sal, b.ename mgr_name
b.Job mgr_Job, b.sal mgr_sal
from emp a = emp b
where a.mgr = b.empno and
lower(a.Job) in ('Salesman', 'Clerk');

3 WAD employee name, employee department number, manager name, manager department number only employees and manager's department should be same.

```
Select a.ename, a.deptno, b.ename mgr_name,
                          b.deptno mgr_deptno
from emp a, emp b
where a.mgr = b.empno and a.deptno = b.deptno;
```

4 WAD emp details, manager name, manager salary only the employee salary should be less than manager salary.

```
Select a.*, b.ename mgr_name, b.sal mgr_sal
from emp a, emp b
where a.mgr = b.empno and a.sal < b.sal ;
```

5 WAD employee name, employee designation, employee date of joining, manager details only employee should be hired after manager hired date.

```
Select a.ename, a.Job, a.hiredate , b.*
from emp a, emp b
where a.mgr = b.empno and
          a.hiredate > b.hiredate ;
```

Case 1

1) WAD ename, sal who is earning same salary.

```
Select a.ename, a.sal
from emp a, emp b
where a.sal = b.sal and a.empno != b.empno;
```

2 WAQD duplicate Sal.

```
Select sal
from emp
group by sal
having count(*) > 1 ;
```

Note :-

Here we got a display "only duplicate values" as an output. (removing all other values we got only duplicate values).

→ To get WAQD distinct Sal.

```
Select distinct (sal)
from emp ;
```

Note :-

Here we got a output ignoring (removing) the duplicate values from salary.

3 WAQD employee name, designation who is working in same designation.

```
Select a.ename, a.Job
from emp a, emp b
where a.Job = b.Job and a.empno != b.empno ;
```

Knowledge Determining unknown values in Self Join :-

1 WAQD employee name, department ~~name~~ number, designation, only the department number should be same as blake "department number".

```
Select a.ename, a.deptno, a.Job
from emp a, emp b
```

where a.deptno = b.deptno and
lower(b.ename) = 'blake' ;

Note :
Sub-query case 1 (determining the unknown values)

2 WAQD employee details only salary should be more than 'James' salary and designation should be same as 'smith' designation
display (emp)
cond (unknown)
cond 2 (unknown)

Select a.*
from emp a, emp b, emp c
where a.sal > b.sal and lower(ename) = 'james'
and a.Job = c.Job and lower(ename) = 'smith';

Note :
One Condition Should be Consider as one table.

Assignment

1) WAQD employee details, manager name, manager annual salary only employee and manager designation should not be same.

Select a.*, b.ename mgr_name, b.sal*12 mgr_annsal
from emp a, emp b
where a.mgr = b.empno and a.Job != b.Job ;

2 WAQD employee name, employee number, employee Commission, manager details only the employee salary should be more than manager Commission.
display emp(a)
emp(b)

Select a.ename,

Select a.ename, a.~~mgr~~ empno, a.comm, b.*
from emp a, emp b
where a.mgr = b.empno and a.sal > b.comm;

3 WAQ employee name, employee manager number, employee new salary with 1000rs increment, manager name, manager new salary with 500rs decrement only employee new salary should not be same as manager new ~~sal~~ Sal.

Select a.ename, a.mgr, a.sal+1000 New_sal, b.ename mgr name, b.sal-500 mgr_new sal
from emp a, emp b
where a.mgr = b.empno and
a.sal+1000 != b.sal-500;

4 WAQ employee details, manager name, manager designation only manager should be hired before the employee hiredate and display the output in descending order based on employee salary.

Select a.*, b.ename mgr-name, b.job mgr-job
from emp a, emp b
where a.mgr = b.empno and b.hiredate > a.hiredate
order by sal;

Sub-query principle Inner Query & outer Query

Co-related sub-query (CRSQ) :-

(case-3)

→ CRSQ is a advance version of sub-query.

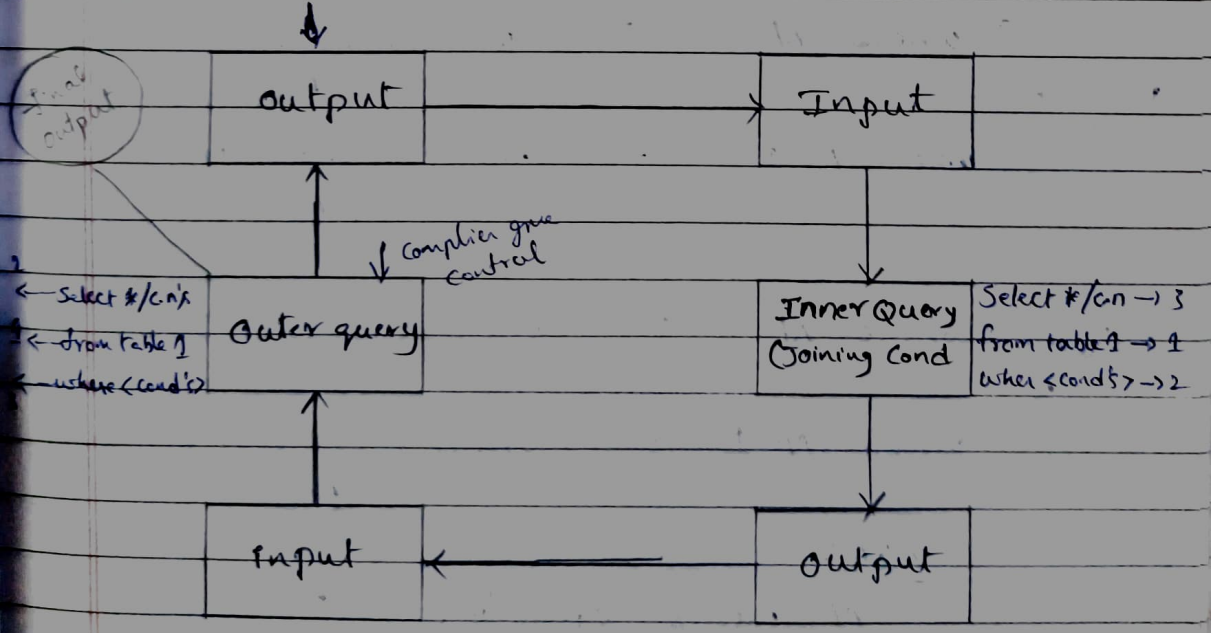
→ CRSQ is working on a principle of Sub-query and Joins

→ In CRSQ Inner query and outer query will be present.

→ In CRSQ Inner query is depending on outer query and outer query is depending on Inner query.

→ In CRSQ, inner query and outer query both are inter dependent on each other because of specifying Joining Condition in inner query

Working Mechanism of CRSQ :-



→ Compiler first will give Control to outer query. Outer query will execute for the first time partially (From clause) then it will prepare's output.

→ The output of outer query will passes as an input to inner query, inner query will execute for the first time Comp-letely. (select, from, where) based on the condition and input taken from outer query it will prepare output.

→ The output of inner query will pass as an input to outer query. outer query will execute for the 2nd time completely based on condition and input taken from inner query it will prep-are final output.

Logic

(Joining Condition)

outer query		inner query
max value: where N	=	where table1.cn <= table2.cn
max value: where N-1	=	where table1.cn < table2.cn
min value: where N	=	where table1.cn >= table2.cn
min value: where N-1	=	where table1.cn > table2.cn

Note:-

'N' mean's equal to (=) Ex:- >=

'N-1" mean's no equa to (Not present)

Ex:- >

1 WAQ 3rd max salary.

$N = 3$ (or) $N - 1 = 2$

```
Select sal ---> 3
from emp a ---> 1
where 2 = (select count (distinct sal) ---> 3
                    mandatory.
from emp b ---> 1
where a.sal < b.sal); ---> 2
          (OR) → Joining cond
```

```
Select sal
from empa
where 3 = (select count (distinct sal)
from empb
where a.sal <= b.sal);
          ↳ Joining cond.
```

emp (a)		emp (b)
Sal		Sal
4000	→ 4	4000
3000	→ 5	3000
7000	→ 3	7000
9000	→ 1	9000
8000	→ 2	8000

4000 <= 4000 --> T
4000 <= 3000 --> F
4000 <= 7000 --> T
4000 <= 9000 --> T
4000 <= 8000 --> T
3 = 4 --> F

3000 <= 4000 --> T
3000 <= 3000 --> T
3000 <= 7000 --> T
3000 <= 9000 --> T
3000 <= 8000 --> T
3 = 5 --> F

7000 <= 4000 --> F
7000 <= 3000 --> F
7000 <= 7000 --> T
7000 <= 9000 --> T
7000 <= 8000 --> T
3 = 3 --> T

9000 <= 4000 --> F
9000 <= 3000 --> F
9000 <= 7000 --> F
9000 <= 9000 --> T
9000 <= 8000 --> F
3 = 1 --> F

8000 <= 4000 --> F
8000 <= 3000 --> F
8000 <= 7000 --> F
8000 <= 9000 --> T
8000 <= 8000 --> T
3 = 2 --> F

o/p :-

Sal
7000

2 WAQD 2nd min sal

```
Select sal
from emp a
where 2 = (select count (distinct sal)
           from emp b
           where a.sal >= b.sal);
```

Advance

1) WAQD 2nd, 4th, 6th max (sal)

```
select sal
from emp
where (select count (distinct sal)
       from emp b
       where a.sal <= b.sal) in (2,4,6);
```

2 WAQD first 3 recent hiredate.

```
select hiredate
from emp a
where (select count (distinct hiredate)
       from emp b
       where (a.hiredate <= b.hiredate)
             in (1,2,3);
             (or)
             <= 3;
             (or)
             between 1 and 3
```

out of topic

1 WAQD emp details of 4th min annual salary

```
Select *
from emp a
where 4 = (select count (distinct sal*12)
           from emp b
           where a.sal*12 >= b.sal*12
```

(Joins) out of topic

chandra's DE Pg

1 WAQD employee name, designation, commission, department name, location of 2nd max Comm.

```
Select ename, job, comm, dname, loc
from emp a, dept
where a.deptno = dept.deptno and
      2 = (Select count (distinct comm)
           from emp b
           where a.comm <= b.comm);
```

2 WAQD department details of 5th min salary.

```
Select dept.*
from emp a, dept d
where a.deptno = d.deptno and
      5 = (Select count (distinct sal)
           from emp b
           where a.sal >= b.sal);
```

Assignment

1) WAQD 5th min annual salary.

```
Select sal*12 as ann_sal
from emp a
where 5 = (Select count (distinct sal*12)
from emp
where a.sal >= b.sal);
```

2) WAQD 7th min annual salary.

```
Select sal*12 Ann_sal
from emp a
where 7 = (Select count (distinct sal*12)
from emp
where a.sal >= b.sal);
```

3 WAQD 4th oldest hiredate

```
Select hiredate
from emp a
where 4 = (select count (distinct hiredate)
from emp b
where a.hiredate >= b.hiredate);
```

4 WAQD 10th recent hiredate.

```
Select hiredate
from emp a
where 10 = (select count (distinct hiredate)
from emp b
where a.hiredate <= b.hiredate);
```

5 WAQD 2nd max Commission.

```
select comm
from emp a
where 2 = (select count (distinct comm)
from emp b
where a.comm <= b.comm);
```

(or)

```
Select comm.
from emp a
where 1 = (select count (distinct comm)
from emp b
where a.comm < b.comm);
```

6 WAQD 3rd min Comm.

```
Select comm
from emp a
where 2 = (select count (distinct comm)
from emp b
where a.comm > b.comm);
```

chandra's

7 WAQD 16th min. new salary with 1000 rs increment

```
Select sal+1000 new-sal
from emp a
where 9 = (select count (distinct sal+1000)
from emp b
where a.sal+1000 >= b.sal+1000);
```

8 WAQD 5th recent hiredate after one year.

```
Select hiredate+365 as New-hiredate
from emp a
where 4 = (select count (distinct hiredate+365)
from emp b
where a.hiredate+365 <= b.hiredate+365);
```

Advanced

doubt → Not working `7=5;`

1 WAQD 1st 5th minimum sal

```
Select sal
from emp a
where (select count (distinct sal)
from emp b
where (a.sal >= b.sal)) in (1,2,3,4,5);
```

LHS_side — Special operator — RHS_side.

2) WAQD 5th, 7th, 10th min annual salary.

```
Select sal*12 Ann_sal
from emp a
where (select count (distinct sal)
from emp b
where (a.sal >= b.sal)) in (5, 7, 10);
```

3 WAQD 10 to 15 oldest hiredate:

```
Select hiredate
from emp a
where (select Count (distinct hired.b)
from emp b
where a.hiredate >= b.hiredate) in
                        (10,11,12,13,14,15);
```

Advance

1 WAQD employee name, salary, date of joining, department details of first 5 min sal

```
Select a.ename, a.sal, a.hiredate, dept.*
from emp a, dept
where a.deptno = dept.deptno and
    (select count (distinct sal)
     from emp b
     where (a.sal >= b.sal)) in (1,2,3,4,5);
```

2 WAQD emp details, dept details of first and 3rd max Comm.

```
Select a.*, dept.*
from emp a, dept d
where a.deptno = d.deptno and
    (select count (distinct comm)
     from emp b
     where (a.comm <= b.comm) in (1,3);
```

→ How to display rowid, rownum?

```
Select emp.*, rowid, rownum
from emp;
```

23/8/21

Pseudo column (False column) :-

pseudo c.n are the False c.n which are not actually present in the table but it will reflect in the ~~table~~ output only when we are specifying in the select statement/clause is called as pseudo column (False column)

table 1

c1	c2	c3	rowid	rownum

original c.n | pseudo c.n

→ Each and every table in a data base should Consist of pseudo column's

→ According to oracle RDBM's pseudo column's are classified into two types.

(i) Rowid

(ii) Rownum.

Rowid :-

→ rowid is the unique and permanent value given to each and every record present in a table.

→ rowid is the 18 bit alpha numeric value

→ rowid represents physical address of each and every record present in a table. (physical address mean's data base name, table name, column name

→ rowid will be automatically generated by insert statement.
→ rowid is the [Static.] → No change

Rownum :-
→ rownum is the unique and temperory value given to each and every record present in table.
→ rownum is the Sequential integer value. (→ one by one (1, 2, 3 et...))
→ rownum represents total number of records selected
→ rownum will be automatically generated by Select statement.
→ rownum is a [dinamic.] → change's.

Note :-
Pseudo c.n are also known as imaginary c.n.

Logic

Outer query — Inner query (Joining Cond)

Top record {
where N = where table1.rowid >= table2.rowid
where N-1 = where table1.rowid > table2.rowid

Bottom record {
where N = where table1.rowid <= table2.rowid
where N-1 = where table1.rowid < table2.rowid

Note :-

Max <	Top >	N (=)
Min >	bottom <	N-1 (No equal to)

1 WAQD Top first employee details.

```
Select *
from emp a
where 1 = (select count (distinct rowid)   <- optional
           from emp b
           where a.rowid >= b.rowid);
```

(or)

```
Select *
from emp a
where 1 = (select count (rowid)
           from emp b
           where a.rowid >= b.rowid);
```

(or)

```
Select *
from emp
where rownum = 1;
```

2 WAQD Bottom Second emp details.

```
Select *
from emp a
where 2 = (Select Count (rowid)
           from emp b
           where a.rowid <= b.rowid );
```

3. WAQD employee name, Salary, department name, loc of top 3 employees.

```
Select ename, sal, dname, loc
from emp a, dept
where 3 = a.deptno = dept.deptno and
          (select count (rowid)
           from emp b
           where a.rowid >= b.rowid) in (1,2,3);
```

Advance

1 WAQ emp details except 1st employee.

```
Select *
from emp a
where (select Count (distinct rowid)
       from emp b
       where a.rowid >= b.rowid) != 
             (select min (rownum)
              from emp);
```

2 WAQ emp details except last employee

```
Select *
from emp a
where (select Count (distinct rowid)
       from emp b
       where a.rowid >= b.rowid) != 
             (select max (rownum)
              from emp);
```

3 WAQ emp details except 1st & last employee

```
Select *
from emp a
where (select Count (distinct rowid)
       from emp b
       where a.rowid >= b.rowid) not in
       ((select min (rownum)
         from emp), (select max (rownum)
                     from emp));
```

4 WAQ emp details 1st and last employee

```
Select *
from emp a
where (select Count (distinct rowid)
       from emp b
```

where a.rowid >= b.rowid) pn
(select min(rownum)
from emp), (select max(rownum)
from emp));

Interview Most asked

5. WAQD 1st half off the employee details.

```
Select *
from emp a
where (select count(distinct rowid)
       from emp b
       where a.rowid >= b.rowid) <=
             (select count(*)/2
              from emp);
```

6 WAQD 2nd half off the employee details.

```
Select *
from emp a
where (select count(distinct rowid)
       from emp b
       where a.rowid >= b.rowid) >
             (select count(*)/2
              from emp);
```

Note:-

Encrypted:-

Not possible to study

like:- AAN34CN, CNNPNOT042
PNPSMNT, INO43FLX

Decrypted:-

possible to study.

like:-

CAP, EMPLOYEE, VISHNU,
SKYPE, Google etc---

Assignment

1 WAQD Top 5th employee details.

```
Select *
from emp a
where 5= ( select count (rowid)
from emp b
where a.rowid >= b.rowid);
```

2 WAQD employee name, designation, salary, annual salary of bottom 7th employee.

```
Select ename, job, sal, sal*12 as ann_sal
from emp a
where 7= (select count (rowid)
from emp b
where a.rowid <= b.rowid);
```

3 WAQD employee name, annual salary, annual commission of top with employee.

```
Select ename, sal*12 ann_sal, comm*12 Ann_Comm
from emp a
where 10= (select count (rowid)
from emp b
where a.rowid >= b.rowid);
```

4 WAQD employee details of first 5 employees (If they ask first 5 then defaultly should take top 5 records.

```
Select a.*
from emp a
where (select count (rowid)
from emp b
where a.rowid >= b.rowid ) in (1,2,3,4,5);
(or)
```

where a.rowid >= b.rowid) <= 5 ;

(or)

where a.rowid >= b.rowid) between 1 and 5 ;

5 WAQ bottom 5th, 7th, 10th employee name, commission, department number.

Select ename, comm, deptno
from emp a
where (Select count(rowid)
from emp b
where a.rowid <= b.rowid) in (5, 7, 10) ;

6 WAQ emp details, departname of bottom 5 employees.

Select a.*, dname
from emp a, dept
where a.deptno = dept.deptno and (select count(rowid)
from emp b
where a.rowid <= b.rowid) in (1, 2, 3, 4, 5) ;

(or)

<= 5 ;

(or)

between 1 and 5 ;

7. WAQ dept details of 2nd employee.

Select dept.*
from emp a, dept
where a.deptno = dept.deptno and 2= (select count(rowid)
from emp b
where a.rowid >= b.rowid)

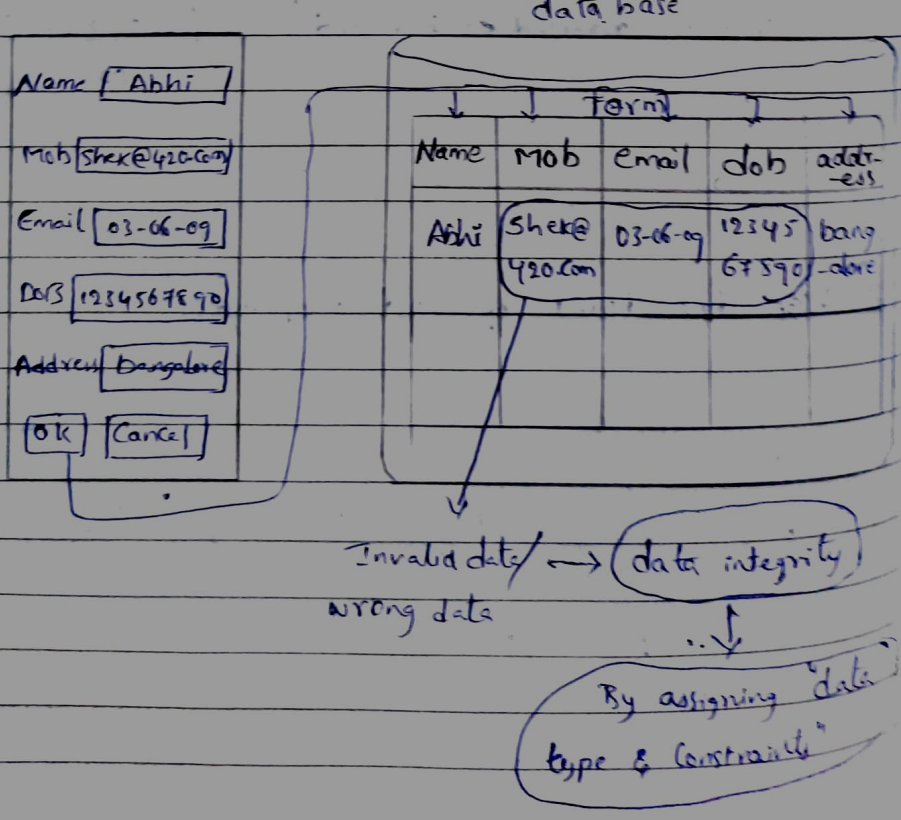
24/08/21

Data integrity :-

* The process of restricting of storing wrong data / invalid data inside the table in database is called as data integrity.

(or)

* The process of checking correctness and accuracy of data inside the table in database is called as data integrity.
* The process of data integrity will avoid of storing "wrong data / invalid data".
* The process of data integrity will ensure we are storing only "valid data".
* The process of data integrity can be achieved through assigning "data type and constraints" to the column present in table.
* "RDBMS" will support data integrity concept.



Note:-

Data will not store in horizontal format. it will store the data in vertical format in data-base.

Data types :-

* Data type represent what type of data should accept or store inside the c.n in a table is called as data type.
* In the table each and every column should be mandatory to assign with data type.
* A column in the table can be assigned with maximum of 1 data type.
* In "ORACLE" we have 'N' number of data types and present but we preffered only 4 data types.
  a) char
  b) Var char
  c) Number
  e) date.

Extra data type's

BLOB (Binary Large object) ---> image, audio, Videos.

CLOB (Character large object) ---> document.

integer etc....)

a) 1. char (character) :-

If any c.n is assigned with "data type of char" then that "c.n will accept only characters".

character means

A-Z, 0-9

a-z, $-#.

Syntax:

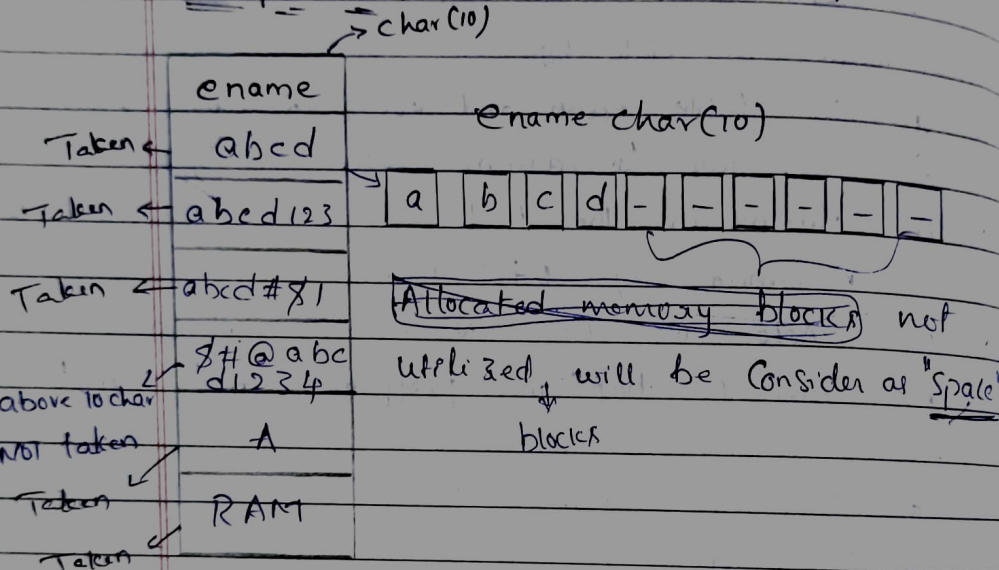
c.n char (size)

Note:

→ The default size of character data type is 1.

→ The ~~default~~ maximum size of Character data type is 2000.

example:- 1 :-



Drawbacks:-

* The allocated memory block which are not utilized will Consider as "SPACE"

SPACE → It is also one "special char"

* Char is fixed length memory allocation.

Note:-

'Char' is a shortcut of "Character"

b) Varchar (variable character) :-

If any c.n is assigned with "data type of Varchar" then that "c.n" will accept only characters".

Character means
- → A-Z (Alphabets)
- → a-z
- → 0-9 (Numbers)
- → $-# (special char)

⇒ Character

Syntax :-

c.n Varchar (size)

example :-

varchar(10)

ename
Vishnu
Vishnu488
488Vishnu
Pavan@48 #$
Pavan@# 420.m
RAM

Taken
Taken
Taken
Taken
Not Taken
Taken

ename varchar(10)

V	I	S	H	N	U				

Not utilized blocks should be considered as NULL

∴ Null = Nothing, No data.

* The allocated memory block which is not utiliz-ed will be considered as "NULL"
* Varchar is a variable memory allocation
* Varchar data type doesn't have any default size.
* Varchar data type can accept maximum 4000 characters. (We can say 10000 char).

Note:-

* From oracle 10g onwards they have introduced a new datatype that is Varchar2 which accepts maximum 10,000 Characters.
(Varchar and Varchar2 both are same).

→ what is the difference between char and Varchar data type :-

Char	Varchar
→ It is fixed length memory allocation.	It is a Variable length memory allocation.
→ Allocated memory blocks Not utilized allocated memory blocks can be Considered as "SPACE"	Not utilized allocated memory blocks can be Considered as "NULL"
→ Default size is 1	No Default Size.
→ Maximum it can accept 2000 char.	Maximum it can accept 4000 char (10,000 char) (After oracle 10g onwards Varchar can also be call as Varchar2).

chandras

c) Number :-

If any c.n / attributes is assigned with a data type as number then it will accept only number / digits.

Number / digits means

[0-9]

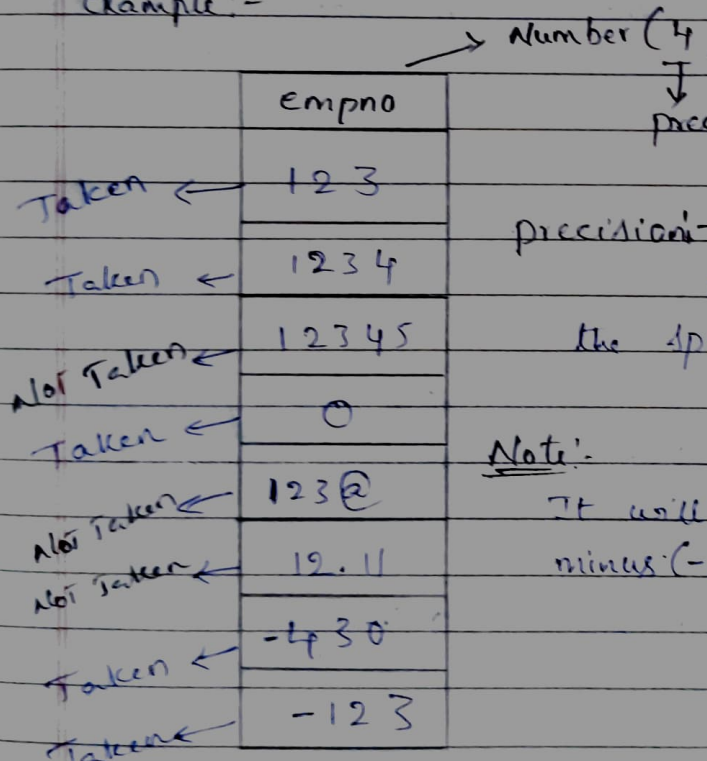
Syntax :-

Number (precession, scale)

Case-1 :- Number (precision)

Precision means maximum number of digits a column (c.n) can be accept.

Example :-



precision :- How many values are to taken from the specific data type.

Note :-

It will also accept minus (-) values.

Case 2 :- Number (precision, Scale)

* Scale represent decimal.
* precision means maximum number digits can accept including decimal.

Example :-

Number(4, 2)
precision scale

Per
1.12
+.1
1256.12
124.123
12.21
-155.1
182.76
-66.78
1255.123

Taken
Taken
Not Taken
Taken
Taken
Taken
Taken
Not Taken

Number(4,2) :-
when we are writing precision and Scale at a time we should want to satisfy both precision and Scale otherwise it throws an ERROR.

* Here we can write upto 38 precision number → (38) → Maximum.

Note :-
-9999 to +9999 range

d. **Date:-**

When a c.n/attribute is assigned with date type it will accept only "date format".

Syntax:-

c.n date.

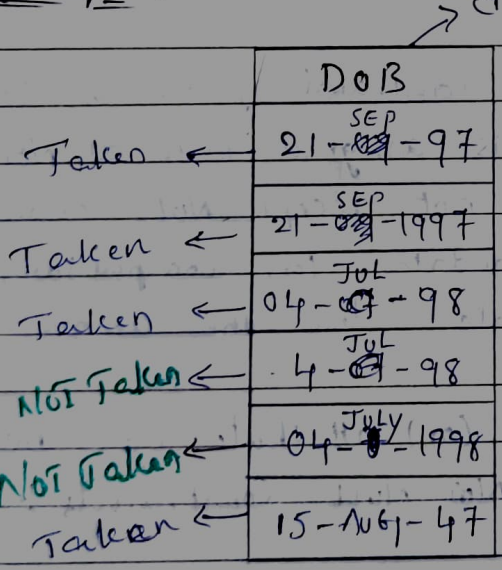
oracle date format :-

DD-MON-YY

(or)

DD-MON-YYYY

Example:-



Note:-

It is mandatory to follow above two formats.

25/08/21

(restriction, limitation, check, obstruction etc.)

a. Constraints :-

* Constraints are the set of values rules (or) validation done to the column to restrict storing wrong data in a column is called as Constraints.

→ Constraints are classified into 5 types. They are :-

a) NOT NULL
b) UNIQUE
c) CHECK
d) PRIMARY KEY
e) FOREIGN KEY

a) NOT NULL

NOT NULL is a constraint given to column, if a column is assigned with a NOT NULL then it will not accept NULL values.

i.e., So it is mandatory (or) compulsary to enter some data below the assigned c.n.

→ which column (or) attribute can be assigned with NOT NULL Constraints :-

the column is mandatary (or) compulsory for a table / entity / object for those columns (c.n) will assig NOT NULL Constraints.

chandra's

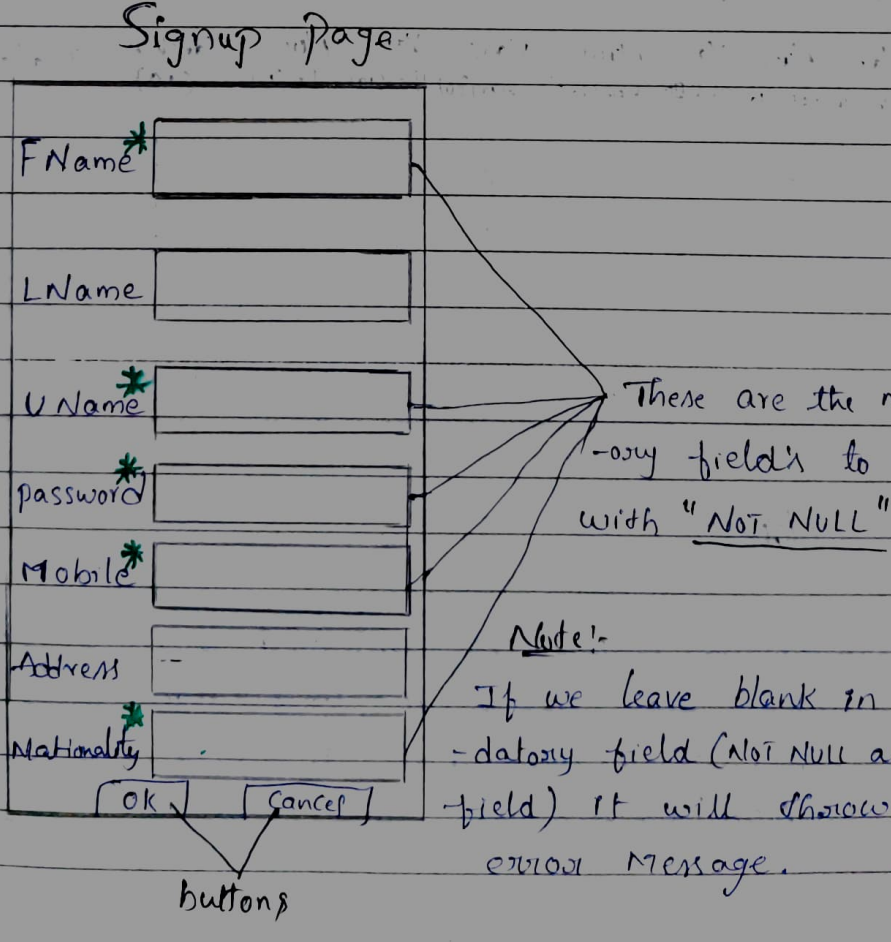
STUDENT

USN	Sname	Mob	Address	Per	email	Branch	yop	Gender	DOB
Var-char(10)	Var-char(20)	Num-ber(10)	Varch-ar(20)	Num-ber(4,2)	Var-chay(20)	Varcha-r(10)	Num-ber(4)	varch-ar(10)	date
NOT NULL	NOT NULL	NOT NULL	NOT NULL	~~NOT NULL~~	NOT NULL	NOT NULL	~~NOT NULL~~	NOT NULL	NOT NULL

Note:-

If a column is assigned with a NOT NULL Const-raint then it will wont accept NULL values but it will accept duplicate value.

Real time example:-



b. UNIQUE :-

→ UNIQUE is a Constraint given to Column. If a Column is assigned with a UNIQUE then it will not accept "duplicate Value". So it will not accept a duplicate value which is already present (or) existing.

→ which Column or attribute can be assigned with UNIQUE Constraints :-

If any Column / attribute which accept only distinct values for those Column will assign UNIQUE Constraints.

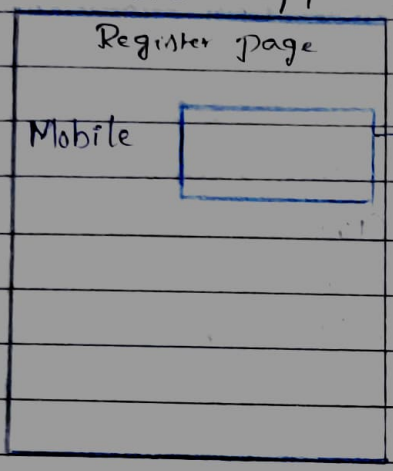
USN	S-name	Mob	Address	per	email	branch	yop	Gender	DOB
Varchar(10)	Varchar(10)	Number(10)	Varchar(20)	Number(4,2)	Varchar(20)	Varchar(10)	Number(4)	Varchar(10)	Date
UNIQUE		UNIQUE			UNIQUE				

Note :-

If a Column is assigned with a Constraint as UNIQUE it will not accept duplicate values but it will accept NULL Values.

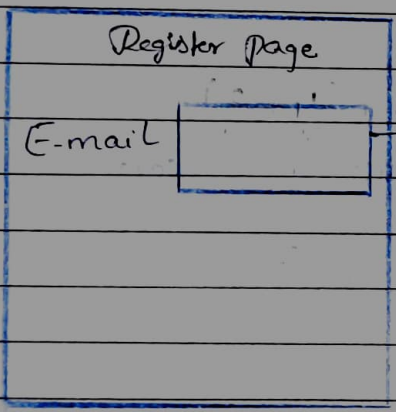
Real time example:-

Whats app



→ Entering a mobile Number which is already exists (or) registered --> then it throws a error message. --> UNIQUE

Flip kart



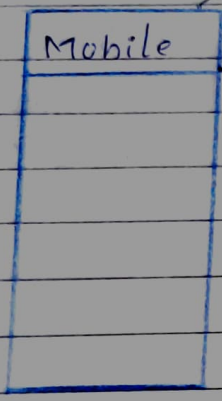
→ Entering a E-mail ID which is already exists (or) regist-ered --> then it throw's a error message --> UNIQUE.

C Check :-

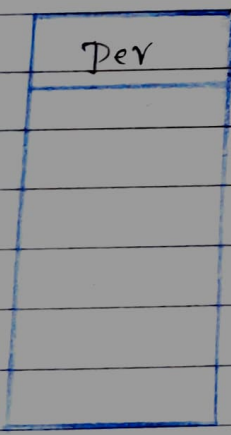
Check is a Constraint which is used for "additional validation." (Not mandatory).

Syntax :-

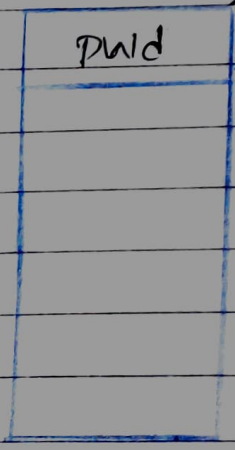
Column_name check (cond).



→ Number (10)
precision
→ check (length (mob) = 10)



→ Number (4, 2)
precision, scale
→ Check (per >= 0)



→ Varchar (6-10) (1 up, 1 Num, 1 Spec char)

Request :-
It should be cont 6 to 10 char
1 upper case, 1 number, 1 special char

Check ((length (pwd) between 6 and 10)) and (pwd = = [A-Z]) and (pwd = = [0-9]) and (pwd = = [$-#]).

chandra's

4 Primary Key :-

Primary key is a Constraint given to a column which is used to identify or represent a record uniquely.

→ which C.n (or) attribute can be eligable to become primary key :-

A Column (or) attribute which is "mandatory" and it should Consist of "distinct value". i.e, ~~that is~~ Combination of "NOT NULL" and "UNIQUE" Constraint ~~for~~ for those type of Column's will eligable to become primary key.

→ Characterishics of primary key :-

→ Primary key is a Combination of "NOT NULL" and "UNIQUE" Constraint

→ If a column is assigned with "Primary key" Constraint then that column will not accept both "NULL" and "duplicate values" (repeated values)

→ In a table more than one column can be eligable to become a "primary key" but only one column can be Considered as a "primary key".

→ In a table "primary key" is not mandatory but it is highly recomended.

→ If a table has "primary key" then we can say that the table has a Strong entity.

→ If a table doesn't has a "primary key" then we can say the table has a weak entity.

Candidate key:-

The Column's which are all eligable to become a "primary key" is know as Candidate key.

Alternative key:-

The Column's which are all eligable to become a "primary key" but which are the Column's not choosen as a primary key is called as "alternative key"

Example:-

According to above example "mobile, email, username" are the Candidate key.

If we choosen "mobile" --> "primary key" then "email, username" --> alternative key.

If we choosen "email" --> "primary key" then "mobile, username" --> alternative key.

If we choosen "username" --> "primary key" then "mobile, email" --> alternative key.

Example :-

VSP_student										
NAME	Mobile	Email	Address	percentage	yop	Branch	Colly Name	Gender	USN	DOB
Varchar (20)	Number (10)	Varcha -r(20)	Varcha -r(20)	Number (4,2)	Numb -er(4)	Varcha -r(10)	Varchar (20)	Varcha -r(10)	Varcha -r(10)	Date
NN, ~~UN~~	NN, UN	NN, UN	NN, ~~UN~~	~~NN~~, ~~UN~~	~~NN~~, ~~UN~~	NN, ~~UN~~	NN, ~~UN~~	NN, ~~UN~~	NN, UN	NN, ~~UN~~

5. Foreign key :-

Foreign key is a referential integrity Constraint it is a column/attribute which belongs to another table.

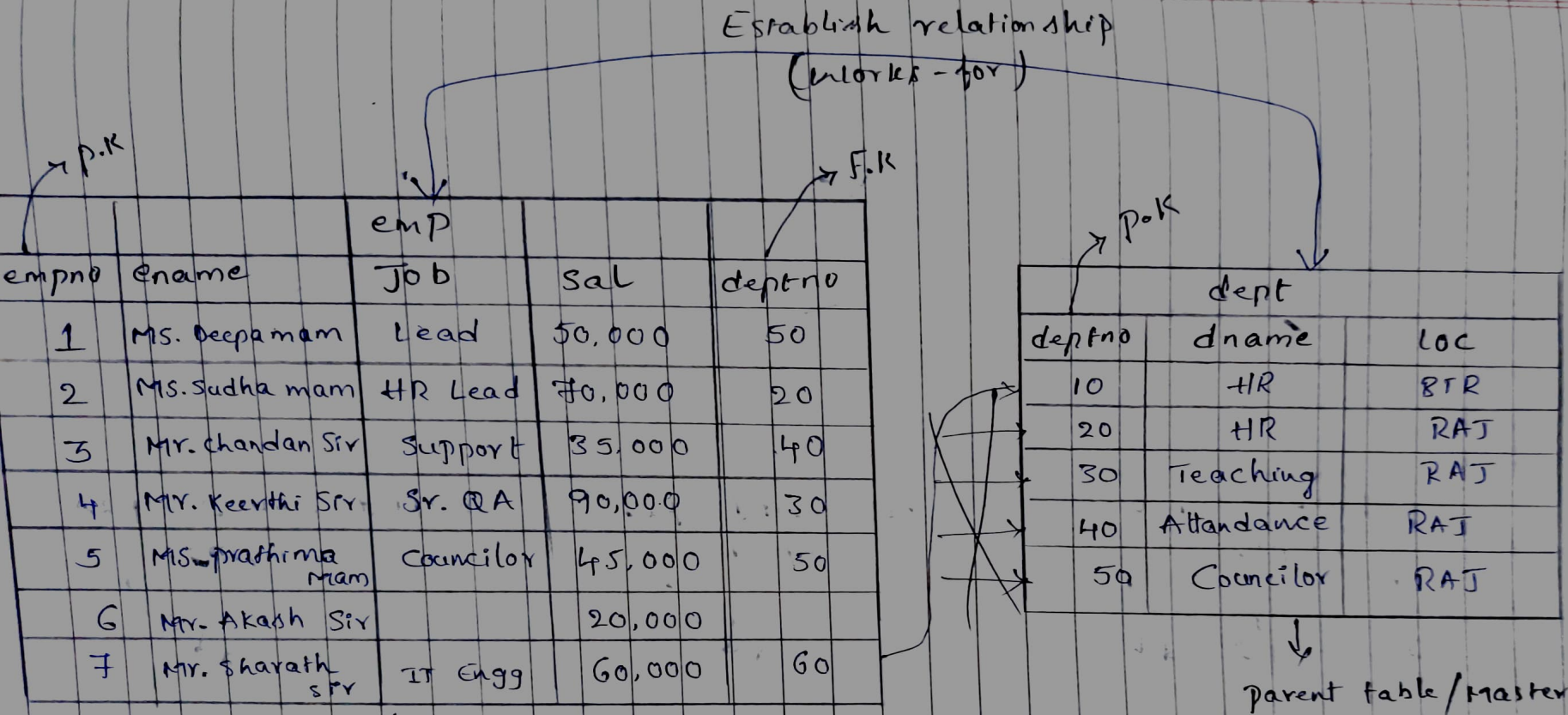
→ why we should use Foreign key :-

It is used to establish the relation between the tables. (the relation ship is called as as Works_for)

→ Characteristics of foreign key" :-

→ The table in which "foreign key" is present is called as "Child table"/"slave table".
→ The table to which actually "foreign key" is belongs is called as "parent table"/"master table".
→ Only the "primary key" can be considered as a "foreign key".
→ In a table we can have more than one column as "foreign key".
→ If a column is assigned with "foreign key" then that column will accept both "Null" and "duplicate values" but it will not accept a values which is not present in "primary key" (parent table column).

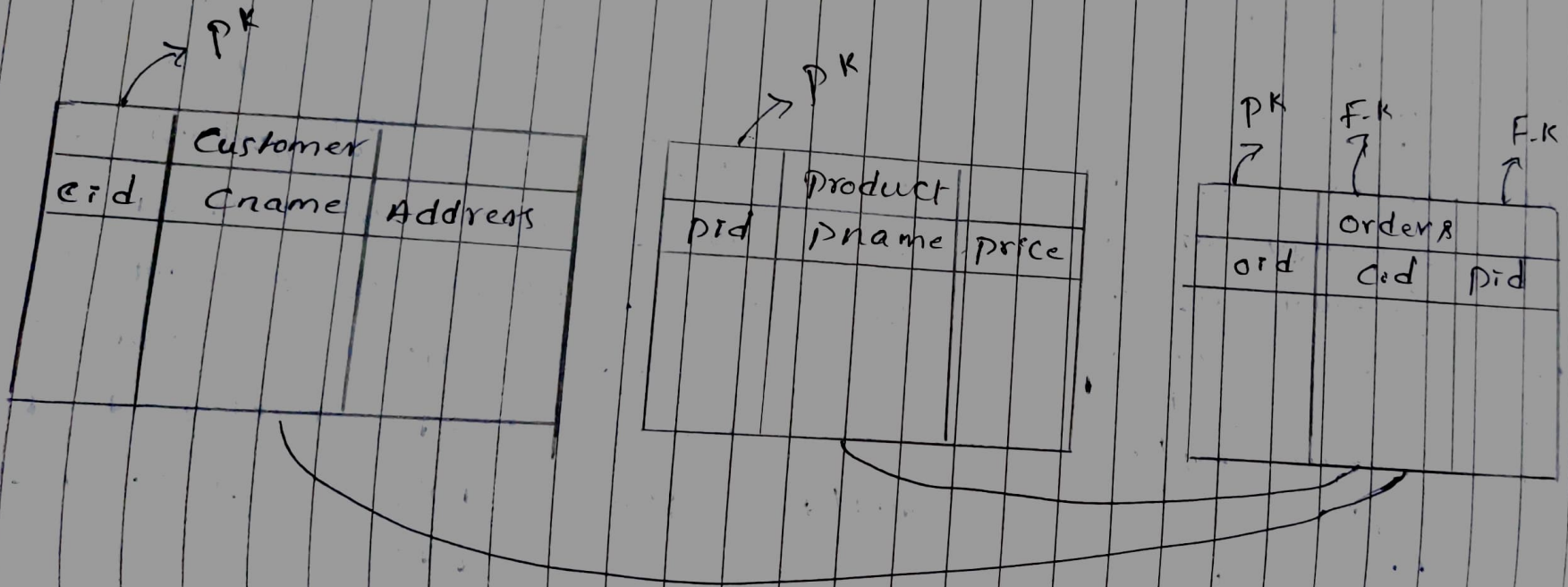
Example 1 :-



Child table / slave table

Parent table / Master table

Here F.K can accept NULL & duplicate but it wont accept a value which is not present in parent table column.


PK
Customer
Cid
Cname
Address
PK
Product
Pid
Pname
price
PK
F.K
F.K
Orders
Oid
Cid
Pid

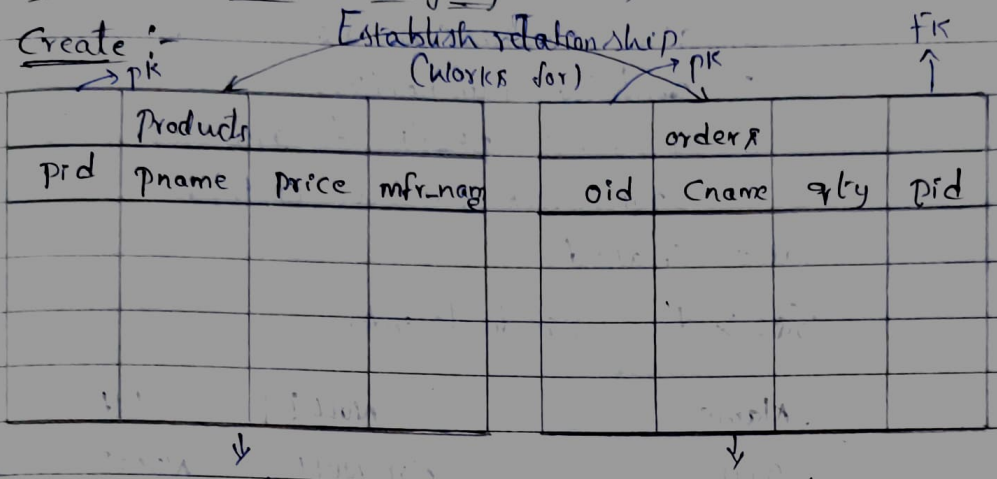
27/8/12

DDL

SQL Stmt's / lang's :-

2 → Data Definition Language (DDL) :-

a) Create :-




```
SQL > Create table product (
  2 pid number(10) primary key,
  3 pname Varchar(20) not null,
  4 price number(4,2) not null,
  5 Mfr_name Varchar(20) not null);
Table Created.
```

SQL > desc products

Name	NULL ?	Type
PID	NOT NULL	NUMBER(10)
PNAME	NOT NULL	VARCHAR2(20)
PRICE	NOT NULL	NUMBER(4,2)
MFR_NAME	NOT NULL	VARCHAR2(20)

Note :-

Here we have ~~only~~ consider only a "primary key".

primary key located table is also known as parent table / Master table.

```
SQL> Create table orders (
  2  oid number(5) primary key,
  3  cname varchar(20) not null,
  4  qty number(10) not null,
  5  pid number(4),
  6  Constraint pk_fk foreign key (pid)
       references product (pid));
```

Table Created.

SQL> desc ~~products~~ orders

Name	NULL?	Type
OID	NOT NULL	NUMBER(5)
CNAME	NOT NULL	VARCHAR2(20)
QTY	NOT NULL	NUMBER(10)
PID		NUMBER(4)

SQL> Syntax

Alter table table_name

Add column_name datatype Constraints;

b. Alter :-

Alter statement is used to modify the table structure

→ Case 1 :- To add a column.

Ex:-

```
SQL> desc orders
```

Name	Null ?	Type
OID		
CNAME		
QTY		
PID		

```
SQL> alter table orders
  2  add discount number(4,2);
```

Table altered

SQL > desc orders

NAME	Null ?	TYPE
OID		
CNAME		
QTY		
PID		
DISCOUNT		NUMBER (4, 2)

→ Case ②: To drop or delete a column.

Ex:- SQL > alter table orders
2 drop column cname ;

Table altered.

SQL > desc orders

NAME	NULL ?	TYPE
OID		
QTY		
PID		
DISCOUNT		

→ Case ③:- To rename a column :-

Ex:- SQL > alter table orders
2 rename column discount to offers ;

Table created.

SQL > desc orders.

NAME	Null ?	TYPE
OID	NOT NULL	
QTY		
PID		
OFFERS		

Syntax :- Drop table table_name

C. Drop :-

Drop is used to delete the table permanently from a data base (db).

Ex:-

```
SQL > desc demo
 NAME                    NULL?    TYPE
 SNAME                            VARCHAR2(20)

SQL > Select * from demo ;

 SNAME
   a
   b

SQL > drop table demo ;
  Table dropped.

SQL > desc demo
ERROR :
ORA-04043 : object demo does not exist.

SQL > Select * from demo ;
Select * from demo
       *
ERROR at line 1 :
ORA-00942 : Table or view does not exist.
```

Flash back statement :-

From oracle 10g onwards we have a concept of recycle bin, we get back the table from recycle bin by using flash back statement.

Syntax :-

Flashback table table_name to before drop;

Ex:

```
SQL > flashback table demo to before drop;
Flash back Completed.

SQL > desc demo

 NAME                    NULL ?    TYPE
 SNAME
```

```
SQL > select * from demo;

 SNAME
 a
 b
```

Purge statement :-

We can also delete the table from recycle bin by using purge statement.

Syntax:- purge table table_name;

Ex:

```
SQL > drop table demo;
Table dropped.

SQL > purge table demo;
Table purged
SQL > flashback table demo to before drop;
flashback table demo to before drop.
*
ERROR at line 1:
ORA-3805: object not in RECYCLE BIN
```

4. Truncate:-

Truncate statement is used to delete all the records from the table permanently.

Syntax:- Truncate table table_name;

Ex:- SQL> Select * from demo;

SNAME
a
b
c

SQL > truncate table demo;

Table truncated.

SQL > desc demo

NAME	NULL?	TYPE
SNAME		VARCHAR2(20)

SQL > Select * from demo;

no rows selected.

5. Rename:- statement

Rename is used to change the table name permanently.

Syntax:-

Rename old_table_name to New_table_name;

Ex:- SQL> rename demo to Sample;

Table renamed.

SQL > desc Sample

Data manipulation Languages:

NAME	NULL?	TYPE
SNAME		VARCHAR2(20).

SQL > desc demo

ERROR:

ORA-04043: object demo does not exist.

→ Difference b/w Drop and Truncate

Drop	TRUNCATE
→ It is used to delete whole table	→ It is used to delete all the records present in table.
→ Table structure will not be present.	→ Table structure will be present
→ Flashback and purge will support	→ Flashback and purge will not support.

→ Difference b/w rename and alias_name.

Rename	Alias_name
→ It is used to change the table_name	→ It is used to change c.n (or) expression present in table and it displayed in result/output.
→ It is used for permanent changes	→ It is used for temporary changes.

Note:-

A Common Column is also known as PRIMARY KEY & FOREIGN KEY.

DML

3 Data Manipulation language :-

a Insert :-

Insert statement is used to add a record in a table.

Syntax :-

Insert into table_name values (V_1, V_2, V_3 -----);
Insert into table_name (col1, col2) values (V_1, V_2);

Ex :- SQL > insert into sdnt values (1, 'akash');
1 row created

SQL > insert into sdnt (rollno, sname)
values (2, 'abhi');

1 row Created

SQL > Select * from sdnt ;

ROLLNO	SNAME
1	akash
2	abhi

b update :-

~~update~~

update statement is used to modify the records present in a table ~~update table name~~

Syntax :-

update ~~sdnt~~ table_name
set col1 = val1 , col2 = val2 ;
where < cond's > ;

SQL

```
SQL> update sdnt
  2  set rollno = 4, sname = 'akashay'

SQL> Select * from stdnt;
```

ROLL NO	SNAME
1	akash
2	abhi
3	amith.

```
SQL> update sdnt
  2  set sname = 'abhi.p'
  3  where rollno = 2;
```

1 row updated.

```
SQL> Select * from sdnt;
```

ROLL NO	SNAME
1	akash
2	abhi.p
3	amith

```
SQL> update sdnt
  2  set rollno = 4, sname = 'akashay'
  3  where rollno = 3;
```

1 row updated.

```
SQL> Select * from sdnt;
```

ROLL NO	SNAME
1	akash
2	abhi.p
4	akashay

```
SQL > updat sdnt
  2  Set Sname = 'dinga';
3 rows updated.

SQL > Select * from sdnt;
```

ROLLNO	SNAME
1	dinga
2	dinga
4	dinga

C. **Delete:-**

Delete statement is used to delete the records present in a table.

Syntax:- delete from table_name where <cond's>;

Ex) SQL > Select * from sdnt;

ROLLNO	SNAME
1	dinga
2	dinga
4	dinga.

```
SQL > delete from sdnt
  2 where rollno = 2;
1 row deleted.
SQL > Select * from sdnt;
```

ROLLNO	SNAME
1	dinga
4	dinga

chandra's

```
SQL> delete from sdnt;
2 rows deleted.
SQL> select * from sdnt;
no rows selected.
```

→ Difference b/w Drop and delete.

Drop	Delete.
→ It will delete the whole table.	→ It will delete a specific record.
→ Table structure will not be present.	→ Table structure will be present.
→ Flashback and purge concept will support.	→ Flashback and purge concept will not support.
→ No where clause (We will not write a where clause)	→ where clause present. (We will write where clause).
→ It is belongs to DDL (Data definition language)	→ It is belongs to DML (Data manipulation language)
→ It is permanent changes	→ It is temporary changes.

→ Difference b/w Truncate & Delete.

Truncate	Delete.
→ It will delete all the records	→ It will delete a specific records.
→ It is belongs to DDL language	→ It is blong's to DML language.
→ No where clause	→ where clause present.
→ It is a permanent changes	→ It is a temporary changes.

DCL

4. Data Control language:

It is used to give the permission and get back the permission from user.

a. Grant:

It is used to give a permission to the user.

Syntax:

Grant Sql statements on table_name to username;

Ex:→ Login as SCOTT user.

→
```
SQL> Show user
USER IS 'SCOTT'
SQL> Select * from regions.
Select * from regions → ERROR
                        *
```

→ Login as HR user.

```
SQL> Show user
USER IS "HR"
SQL > grant insert, select on regions
                        to scott;
Grant Succeeded.
```

→ Login as Scott user.

```
SQL> Show user
USER IS "SCOTT"
SQL> Select * from hr.regions;
```

chandra's

Region_ID	Region_name
9	dubai
10	India
⋮	⋮

→ SQL> insert into hr.regions values (14, 'Dubai');
1 row created.

→ SQL> delete from hr.regions
2 where region_id = 14;
~~delete from hr.regions~~

*

Note 1 :-
what ever the conditions (insert, select) HR user given based on that in SCOTT user we want to build (or) write a query.
otherwise it throws an error message.

Note 2 :-
In Grant and revoke Concept when ever we are using a table from another user. It is always better to write User_name.table_name

b Revoke :-
It is used to get back the permission from the user.

Syntax :
Revoke Sql statement on table_name from user_name ;

→ Login as HR user

→ SQL> show user

USER is "HR"

SQL> revoke insert, select on regions from SCOTT;

Revoke Succedded.

→ Login as SCOTT user.

SQL> Show user

USER is "SCOTT"

SQL> Select * from hr.regions;;

Select * from hr.regions

*

ERROR at line 1

ORA-00942: Table or View does not exist.

Howl to Save in laptop:-

→ Spool

→ >Spool Drive name:\ Folder name append

type query / execute query

>Spool off.

29/08/21

chandra's

Views :-

→ Views is like a table but it is not a table.
→ Views will be created by selecting only specific/required no. of column's. is called as View.
→ A views are which is created by a table is called as target table.
→ Views are classified as into 2 types.
  a) Simple View
  b) Complex View

a) Simple View :-

A View which is created by a single table / target table is called as simple View.

Syntax :-

```
Create View View_name
as
Select c.n's
from table 1 ;
```

Ex:

```
SQL > Create View sm_View
  2  as
  3  Select ename, Job, deptno
  4  from emp ;
View Created.
```

```
SQL > Select * from sm_View ;
```

ENAME	JOB	DEPTNO
Smith	Clerk	20
Allen	Salesman	30
Ward	Salesman	30
:	:	:

b) Complex View:-

* A view which is created by a more than one table/target table is called as Complex View.
* In Complex View we should specify a Joining Condition.

Syntax:-

```
Create View View_name
as
Select cn's
from table 1, table 2.....
where <cond's> ;
```

Note:-

Views are not permanent, it will be delete based on system date.

Ex: SQL > Create View cmp_view
2 as
3 Select ename, job, dname, loc
4 from emp, dept
5 where emp.deptno = dept.deptno;
View Created.

SQL > Select * from cmp_view;

ENAME	JOB	DNAME	LOC
MILLER	CLERK	ACCOUNTING	NEW YORK

Ex: SQL > Grant select on sm_view to hr;
Grant sucessed.
SQL > login as hr user
SQL > Select * from Scott. sm_view;

ename	Job	deptno
akash	clerk	20

TCL

5 Transaction Control language:-

Note:-

* Any changes / modifaction done by using DDL lang is a permanent changes.
* Any changes / modification done by using DML lang is a temporary changes, to make it as permanent we should go TCL.
* Any changes / modification done WRT cn's then we should go "alter" stmt.
* Any changes / modification done WRT row's then we should go "DML" stmt.

a Commit:-

This statement is used to save the DML languages

Syntax:- Commit ;

Ex: SQL > insert into s1 values ('a');

1 row created.

SQL > insert into s1 values ('b');

1 row created

SQL > Select * from s1;

SNAME
a
b

SQL > Commit;

Commit Complete.

SQL > Select * from s1;

SNAME
a
b

b. **Roll back:-**

Roll back statement is used to UNDO the DML changes

Syntax:- Roll back;

Ex:-

```
SQL> Select * from s1;

SNAME
-----
a
b

SQL> Insert into s1 values ('c');
1 row created

SQL> Select * from s1;

SNAME
-----
a
b
c

SQL> rollback;
Roll back complete.

SQL> Select * from s1;

SNAME
-----
a
b

SQL> Insert into s1 values ('c');
1 row Created.

SQL> Commit;
Commit Completed.

SQL> roll back;
rollback complete
```

SQL > select * from s1;

SNAME
---
a
b
c

Returns 3 rows

SYLLABUS COMPLETED

Resume points :-

→ Good knowledge on JOIN, SUB-QUERY, CRSQ
→ Good knowledge on DATA TYPES, CONSTRA
- INTS.
→ Good knowledge on SQL Stmt / Lang.

→ Hands on experience on oracle 11g RDBMS Database.

Any company INTERVIEW :-

→ JOINS (All 5 types theory and syntax)
--> Inner join (query)

→ SQL stmt/lang (all syntax)
* Difference b/w delete / drop
* Difference b/w delete / truncate.
* " " drop / truncate

→ Sub-query (3rd max sal, 3rd min... query's and unknown values)

→ Pseudo Column (rowid and rownum), ~~CRSQ~~

→ CRSQ

→ Difference b/w dbms/rdbms, pk/fk, where clause/having clause.

→ Grouping clause & having clause

→ operators (IN, BETWEEN, IS, LIKE).

SET operator

→ Set operator is used to combine multiple query's output into a single output.

1. union :-

This operator will combine multiple query o/p into a single without any duplicate.

Ex:- $A = \{x, 2, 4, a, b\}$

$B = \{3, x, y, b, 4\}$

$A \cup B = \{x, 2, 4, a, b, 3, y\}$

example :-

```
Select ename
from emp
where lower(job) = 'salesman'
Union
Select ename
from emp
where mgr = 7698;
```

o/p :-

ENAME

ALLEN
WARD
MARTIN
JAMES
TURNER.

2. union all :-

* This operator will combine multiple query's output into a single with duplicate values.

Example:

$A = \{x, 2, 4, a, b\}$

$B = \{3, x, y, b, 4\}$

$A \cup B = \{x, 2, 4, a, b, 3, x, y, b, 4\}$

example :-

```
Select ename
from emp
where lower(job) = 'salesman'
union all
Select ename
from emp
where mgr = 7698;
```

o/p:

ENAME

ALLEN
WARN
MARTIN
TURNER
ALLEN
WARD
MARTEN
TURNER
JAMES

3. Intersect :-

This operator return only the common values present in both the query.

$A = \{x, 2, 4, a, b\}$

$B = \{3, x, y, b, 4\}$

$A \cap B = \{x, 4, b\}$

example

```
Select ename
from emp
where lower(job) = 'salesman'
Intersect
Select ename
from emp
where mgr=7698;
```

o/p:-

ENAME

ALLEN
MARTIN
TURNER
WARD

4 MINUS :-

* It will return the output of a value which is not present in other query.

$A = \{x, 2, 4, a, b\}$

$B = \{3, x, y, b, 4\}$

$A - B = \{2, a\}$

$B - A = \{3, y\}$

example 1-

```
Select ename
from emp
where lower(job) = 'salesman'
minus
Select ename
from emp
where mgr = 7698;
```

o/p:- No rows Selected.

Example 2 :-

```
Select ename
from emp
where mgr = 7698
minus
Select ename
from emp
where lower(job) = 'salesman';
```

o/p:- ENAME :

JAMES

T.C.L

c) SAVE POINT

```
SQL> Insert into a1 values('a');
1 row Created.
SQL> Insert into a1 values('b');
1 row Created
SQL> Inser into a1 values('c');
1 row Created
SQL> Select * from a1;

SNAME
---
a
b
c

SQL> Save point a;
Save point Created.
SQL> delete from a1
  2  where Sname = 'b';
1 row deleted
SQL> Select * from a1;

SNAME
---
a
c

SQL> Save point b;
Save point created.
SQL> update a1 &
  2  Set Sname = 'd'
  3  where sname = 'c';
1 row updated.
```

SQL> Select * from a1;

SNAME

a

d

SQL> Save point 'c';

Save point Created.

SQL> roll back to c;

roll back Completed.

SQL> Select * from a1;

SNAME

a

d

SQL> roll back to b;

roll back completed.

SQL> Select * from a1;

SNAME

a

c

SQL> roll back to a;

Roll back Completed.

SQL> Select * from a1;

SNAME

a

b

c

chandra's

Save point :-

Save point statement is used to mark the position in data base (db) which helps to roll back statement to roll back to the original position.

~~Save point Savepoint_name;~~

Syntax :-

```
Savepoint Savepoint_name;
roll back to Save point_name;
```

12/9/21

Single row function's (SRF) :-

* Single row function are the functions which can accept n records as input and it provides n records as output.
* Single row function execute record by record
* It can be used in select clause and where clause.
* Single row functions can be nested
* We can pass CN/expr/literals as an argument to single row functions

Single row function as classified into 5 types:

* Character function.
* Number function
* General function
* Date function
* Convertion function

A Number function :-

1 Mod() :-

* This function is used to obtain modules of a given number.
* Modulus means remainder.

Syntax: MOD(m,n)

Ex

```
SQL> Select mod(10,2)
  2  from dual;

MOD(10,2)
---------
        0

SQL> Select mod(9,2)
  2  from dual;

MOD(9,2)
--------
       1
```

1 WAQ emp details only the employees those are earning EVEN salary.

```
Select *
from emp
where mod(sal,2) = 0;
```

ODD sal :-

```
Select *
from emp
where mod(sal,2) != 0;
```

2 Round function () :-

* This function is used to round off the given number.
* If the decimal >= .5 ---> round off to Next number.
* If the decimal < .5 ---> round off to the same number

Syntax :-

```
Round (arg 1)
```

```
SQL> Select round (59.49)
  2  from dual;

o/p:  ROUND(59.49)
        59

SQL> Select round (59.51)
  2  from dual;

o/p:  ROUND(59.51)
        60
```

3. Trunc() :-

* This function is used to round off the given number always to the same number.

Syntax:- Trunc (arg1)

Ex

```
SQL> select trunc (59.49)
  2  from dual;

TRUNC(59.49)
      59

SQL> Select trunc (59.59)
  2  from dual;

TRUNC(59.59)
      59
```

4 Sqrt() :-

* This function is used to obtain Square root of a given number.

Syntax:- Sqrt (arg)

Ex:-

```
SQL> Select sqrt(4)
  2  from dual;

SQRT(4)
-------
      2.

SQL> Select sqrt(2)
  2  from dual;

SQRT(2)
-------
1.414
```

5 power() :-

This function is used to obtain m^{nth} value.

Syntax:-

power(m,n)

Ex:-

```
SQL> Select power(2,3)
  2  from dual;

POWER(2,3)
----------
         8

SQL> Select power(3,2)
  2  from dual;

POWER(3,2)
----------
         9
```

15 General function :-

* NVL () :-

Note :-

WAQ ename, sal, comm, total salary (sal + comm) of all the employees.

```
Select ename, sal, comm, sal+comm totalsal
from emp;
```

→ According to the above we will get wrong output because if we perform any Arthimatic operation with the NULL. the result will be NULL only, To handle NULL values we should go NVL function.

Definition :-

It is used to handle null values

Syntax :-

NVL (arg1, arg2)

* If arg1 is null value nvl function will return the value present in arg2
* If arg1 is not null value NVL function will return arg1 itself.

* WAQ ename, sal, comm, total Salary (sal + comm) of all the employees.

```
Select ename, sal, comm, sal + nvl (comm, 0) as
                                        total_sal
from emp;
```

chandra's

2 NVL2() :-

It is used to handle null values.

Syntax:

NVL2 (arg1, arg2, arg3)

* If arg1 is null value NVL2 function will return the value present in arg3, if arg1 is not a null is returns value present in arg2.

(i) WAD ename, sal, comm only the employees those who are Comm by giving 30% increment than total sal.

Select ename, sal, comm, sal+nvl2(comm, comm*1.3, 0) total_sal
from emp;

3 Case() :-

* It is similar to if else statement.

Select Case
when cond1 then 'result1'
when cond2 then 'result2'
when cond3 then 'result3'
.
.
Else 'result'
End from table_name;

(ii) WAD employee name, salary only if the employee are earning salary in a range of 500 to 1500 display output as poor salary else if the salary in a range of 1501 to 2500 then display the output as ok ok salary

else if the salary in a range of 2501 to 3500 then display the output as average salary. else display the output as richest person.

```
Select ename, sal, case
when sal between 500 and 1500 then 'poor Salary'
when sal between 1501 and 2500 then 'ok ok Salary'
when sal between 2501 and 3500 then 'Average Salary'
else 'richest person'
end from emp;
```

palindrome String query:-

```
Select ename, Case
where reverse (ename) = ename. then
                        'palindrome String'
else 'it is a not a palindrome string'
end from emp.
```

C. Date function:-

1. Sysdate:-

It is used to obtain the system date where database is present

Syntax:- Sysdate;

Ex:

```
SQL > Select Sysdate
  2 > from dual
```

SYSDATE
12-SEP-21

① WAQD ename, hiredate, total years of experience?

```
Select ename, hiredate, round((sysdate - hiredate)/365)
                                                  expr
from emp;
```

2. systimestamp:-

This function is used to obtain date along with the time.

Ex:

```
SQL > select systimestamp
  2 from dual;
```

SYSTIMESTAMP
12-SEP-21 9.45.10.417000 AM +05:30

3. Months_between ( ) :-

It is used to obtain no of months between two dates.

Syntax:

Months_between (date, date)

Eg) SQL> select months_between ('12-SEP-2022', sysdate)
2 from dual;

o/p:- MONTHS_BETWEEN ('12-SEP-2022', Sysdate)
12

(1) WAQD ename, hiredate, total years of expr.

Select ename, hiredate, round (months_between (sysdate, hiredate)/12) expr
from emp;

4. Last_day ( ) :-

It is used to obtain the last day of the given month in the form of date.

Syntax:- Last_day (date)

Eg) SQL> select last_day (sysdate)
2 from dual;

LAST DAY ( )

30-SEP-21

5. extract function:

This function is used to extract the components of date

→ Date components are (day, month, year)

Syntax:

Extract (day/month/year from date)

→ SQL> Select extract (day from Sysdate)
2 from dual;

o/p:- 24

SQL> Select extract (month from Sysdate)
2 from dual;

o/p:- 10

SQL> Select extract (year from Sysdate)
2 from dual;

o/p:- 2021

Note:-

By using extract function whenever we are extracting month it will return in number format and year in 4 digit

WOD emp details only the employees should be hired in feb81. (without Like)

Select *
from emp
where extract (month from hiredate) = 2 and
extract (year from hiredate) = 1981;

D. Conversion function:-

1. Case manipulation function:-

→ upper()
→ lower()
→ initCap()

2 to_char():-

→ This function is used to convert the date format to character format.

Syntax To_char(date, 'Format models').

→ Format models are:-
D, DD, DY, DAY, MM, MON, MONTH, YY
YYYY, YEAR

→ D:-
This model it will display a day in week day numbers.

Sunday → 1
Monday → 2
:
:
:
Saturday → 7

Ex: Select to_char(Sysdate, 'd')
from dual;
o/p:- 1

→ DD :-

This model will display a day in number.

Ex:- Select to_char (Sysdate, 'dd')
from dual;

o/p :- 24.

→ DY :-

This model it will display a day in beginning 3 Alphabets

Ex:- Select to_char (Sysdate, 'dy')
from dual;

o/p :- SUN

→ DAY :-

This model it will display a day in Complete Alphabet

Ex:- Select to_char (Sysdate, 'day')
from dual;

o/p :- Sunday.

→ MM :-

This model will display a month in number format

Ex:- Select to_char (Sysdate, 'mm')
from dual;

o/p :- 10

→ MON:

This model is used to display first 3 alphabets of month

Ex: Select to_char (Sysdate, 'mon')
from dual;

o/p: oct

→ MONTH:

This model is used to display the month in Alphabet format.

Ex: Select to_char (Sysdate, 'month')
from dual;

o/p: october.

→ YY:

This model is used to display last 2 digits of the year

Ex: Select to_char (Sysdate, 'yy')
from dual;

o/p: 21

→ YYYY:

This model is used to display the year in number format

Ex: Select to_char (Sysdate, 'yyyy')
from dual;

o/p: 2021.

→ YEAR:

this model is used to display 'year' in the Alphabets format

Ex: Select to_char(Sysdate, 'year')
from dual;

o/p: twenty twenty-one

→ WAQD emp details should be hired in Weekends

Select *
from emp
where to_char(hiredate, 'd') in (7,1);

(not in weekends) (or)

Select * from emp
where to_char(hiredate, 'd') not in (7, 1);

→ WAQD emp details only the employee should be hired in feb-81. (without like, extract)

Select *
from emp
where to_char(hiredate, 'mon') = 'feb' and
to_char(hiredate, 'yy') = 81;

E. Character Function:-

1 Length:-

It is used to obtain length of a given string.

Syntax: Length (cn/'string')

Ex: Select length ('sumanth')
from dual;

O/p: 7

→ WAQD emp details only employee should be having exactly '4' digit salary.

Select *
from emp
where length (sal) = 4;

→ WAQD emp details only the ename should be exactly '5' char.

Select *
from emp
where length (ename) = 5;

2→ Concat Function:-

It is used to Concatenate two strings.

Syntax: Concat (arg1, arg2)

Ex: Select Concat ('don', 'key')
from dual;

O/p: donkey

Ex:2 Select Concat ('don', concat ('key', 'Sumanth'))
from dual;

O/P donkey Sumanth.

Note:-

Concat function will accept only two arguments

3 Reverse function:-

This function will reverse a given String

Syntax ('String'/c.n)

Ex: Select reverse ('sunil')
from dual;

O/P:- linus.

4. Replace function:-

This function is used to replace old character with new character.

Syntax: Replace (c.n/'string', 'old ch', ['new ch'])

Ex: Select replace ('spider', 'i', 'q')
from dual;

O/P: qspiders

Ex: Select replace ('spideri', 'i', 'q')
from dual;

O/P:- qspiderq

Ex: Select replace ('ispiders', 'i')
from dual;

o/p: spiders

5 Substr() (Sub String) :-

This function is used to extract characters or sub string from the given string.

Syntax:

SUBSTR (c.n/'string', Start_pos, [Length])

-Ve ---------->

-5	-4	-3	-2	-1
A	L	L	E	N
1	2	3	4	5

+Ve ---------->

Substr ('allen', 1, 2) --> al

Substr ('allen', 1, 4) --> alle

Substr ('allen', 3, 2) --> le

Substr ('allen', 1, 1) --> a

Substr ('allen', 3, 5) --> len

Substr ('allen', 4, 5) --> en

Substr ('allen', 1) --> allen

Substr ('allen', 3) --> len

Substr ('allen', -3, 2) -> le

Substr ('allen', -2, 1) --> e

Substr ('allen', -4) -> llen

chandra's

Note:

→ If Start Position is positive (+ve) (or) negative (-ve) extract will happen from left to right.

→ If we are not specifying length it will extract dell last character.

(i) WAQ employee name only the employee name should begans with 's' character (without like.)

```
Select ename
from emp
where Substr(ename, 1, 1) = 's';
```

(ii) WAQ ename only the ename should begin with vowels.

```
Select ename
from emp
where Substr (lower (ename), 1, 1) in ('a', 'e', 'i',
                                       'o', 'u'); --> Vowels.
where Substr (lower (ename), 1, 1) not in
                                       ('a', 'e', 'i', 'o', 'u'); --> Consonants.
```

(iii) WAQ ename only employee name should ends with 's' char

```
Select ename
from emp
where Substr (lower (ename), -1, 1) = 's';
                    (or)
where Substr (lower (ename), length (ename)) = 's';
```

6 instr (in String) :-

instr function is used to Search the character or a SubString from the given String

Syntax:

INSTR (c.n/'String', 'char or ss', [S-pos, occurance])