★ Rename :- [Renaming the column]

Syn :- alter table tablename
rename column oldcol to newcol;

→ alter table emp
rename column empno to eid;
Table is altered

→ Alter table emp
rename column ename to Name;
Table is altered

→ alter table emp
rename column deptno to did;
Table as altered.

★ modify :- [changing the data]

→ if the records already present it will through error. if we try to change one data type to another data type.

→ Only we can increase and decrese the size if records are already present.

→ data must be empty to change one data type to another data type.

→ alter table emp
modify ename varchar(20);
table altered

→ DESC EMP:

EMPNO
ENAME
JOB
MGR
HIREDATE
SAL
COMM
DEPTNO

→ alter table emp
modify ename varchar(6);
table altered.

→ alter table emp
modify ename varchar(5);
Error at line 2
Cannot decrease column length because some value is to big.

→ alter table emp
modify CNAME number(10) unique;
Table altered

→ alter table emp

☆ Drop :- [Delete the table]

syn : Drop table tableName;

syn 2 : show recyclebin;

syn 3 : Flashback table tableName to before drop;

syn 4 : purge table tableName;

syn 5 : purge recyclebin;

→ Select * from tab;

→ drop table jspidey;

Table dropped

→ drop table workers;

Table dropped

→ select * from tab;

→ show recycle bin;

→ purge table jspidey;

→ Table purged

→ show recycle bin;

→ select * from tab;

Table purged

→ show recycle bin;

→ select * from tab;

→ drop table course;

Table dropped

→ drop table product;

Table dropped

→ drop table P;

Table dropped

→ drop Table student;

table dropped

→ show recycle bin;

→ Flash back table p to before drop;

Flashback complete

→ select * from tab;

→ show recyclebin;

→ recyclebin purged.

→ select * from tab;

→ show recyclebin;

→ purge table p;

*

error at line 1:

object not in recycle bin.

Truncate: [Delete all the records permanantly]

Syn: Truncate table tablename;

→ Truncate table bonus

Table truncated

→ select * From bonus;

no rows selected

Rename:

syn: rename oldtable to newtable;

→ rename bonus to bon;

→ select * from tab;

→ Difference between Delete and truncate?

✲ Delete:

→ Delete is temparary, we can undo the delete operation

→ Delete is slower than truncate.

→ In delete wewill use where clause to delete specified record.

→ Delete is "DML" Command.

✲ Truncate:

→ Truncate is DDL Command

→ Truncate will delete the all records permanantly.

→ Truncate is faster than delete.

→ In truncate we cannot use where condition. so that we cannot delete particular record.

★ Drop:-

→ drop "ddl" command.

→ drop is used to delete table permanantly.

→ when you delete the table it will be stored in the recycle bin

→ if you want to delete the table permanantly from recycle bin we will use purge.

→ SQL Functions:-

→ SQL function is used to do some specific task. These are the inbuilt function.

There are two types:-

(1) single row function
(2) multi row function.

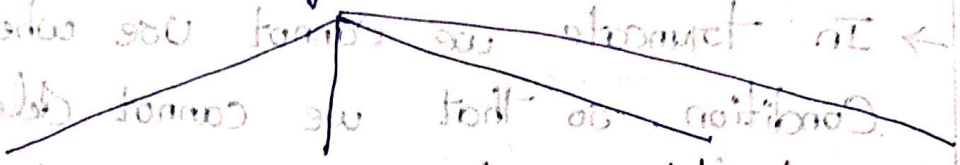
→ single row function:-

single row function it will work for every row but it gives multiple answers.

→ multi row function:-

Multi row function works for every rows gives the single answer.

Single row functions



number functions :-	char functions	date functions	Conversion functions
power()	length()	months_between()	To_date()
mod()	substr()	current_date()	
abs()	upper()	sessiontimezone()	
sqrt()	lower()	sysDate()	
floor()	replace()		
ceil()	Trim()		
round()			

→ Multi row functions :-

Max()
min()
Avg()
sum()
count()

→ Power() :- [pass the 2 values]

→ Select power(5,8)
from dual;

→ Select ename, power(sal,2)
from emp;

→ Select power(4.6,3)
from dual;

power(46,3)

97.336

→ select (56,3) → power (56,3).
from dual;
175616.

→ Mod():

→ select sal, mod (sal, 2)
from emp;

* pass the two values [first number always big % second number small]

→ select mod (45, 3)
from dual;

O·P :- 0

→ select mod (65·89, 3)
from dual; O·P :- 2·89

→ select mod (-34·5, 2)
from dual; O·P :- -5

→ Sqrt :- [only one number pass]

→ select sqrt (625)
from dual;

sqrt
25

→ select sqrt (121)
from dual;
O·P :- 11

→ Select sal, sqrt (sal)
from emp;

→ select sqrt (25)
from dual;
O·P :- 5

→ select sqrt (49)
from dual;
O·P :- 7

→ select sqrt (56·45)
from dual;
O·P :- 7.51333215

→ select sqrt (-243)
from dual;
Error line :1
argument Out of the range

Never accept -ve values

→ abs() :- (-"values converted into" positive values)

→ select abs (45.79)
from dual;

ABS (45.79)
45.79

→ select abs (-35)
from dual;

ABS (-35)
35

→ select abs (23)
from dual;

ABS (23)
23

→ floor() :- [floor every time will give down [lower] values]

→ select floor (4.1)
from dual;

floor (4.1)
4

→ select floor (4.8)
from dual;

floor (4.8)
4

→ select from (4.999)
from dual;

floor (4.999)
4

→ Ceil(): [ceil every time will give high value]

→ Select ceil(4.2)
from dual;

CEIL(4.2)
5

→ Select ceil(4.99)
from dual;

CEIL(4.999)
5

→ Select ceil(-4.5)
from dual;

CEIL(-4.5)
-4

→ Round(): [floor and ceil combine Round]

→ select round(4.999)
from dual;

Round(4.999)
5

→ Select round(-4.2)
from dual;

Round(-4.2)
-4

→ Select round(-4.5)
from dual;

Round(-4.5)
-5

char @unctions :-

→ **length :-**

→ Select length ('hello')
from dual ;

O·P :- 5

→ Select length ('_hellow__') → space also include
from dual;

O·P : 8

→ select ename , length (ename)
from emp;

→ select job , length (job)
from emp;

→ select empno , length (empno)
from emp;

→ **Upper() :-**

It convert lower to upper case. if provide Query is already upper it convert upper.

→ select upper ('Hellow')
from dual;

O·P :- HELLOW

→ select upper ('HEllow')
from dual;

O·P : HELLOW

→ Lower() :-

→ Select Lower ('HELLOW')
from dual;
o.p :- hellow

→ Select Lower ('hellow')
from dual;
o.p :- hellow

→ Trim() :-

→ It remove the extra space stating and ending
→ ending and starting letter will be same trim work on starting and ending letter remove.

→ select trim ('-hi hellow---')
from dual;

TRIm ('HI' HEL')

hi hellow

→ select trim ('h' from 'heelh')
from dual;

TRI
EEL

→ Replace() :-

(1) syn :- Replace (Source, Old char, new char);
(2) syn : Replace (source, delete char);

→ select Replace ('mounika', 'o', 'a')
from dual;

REPL
maunika

→ select replace ('chinna', 'a', 'i')
from dual;

O·p :- chinni

→ Select replace ('Bhaskar', 'h', 'i')
from dual;

O·p :- Biaskar

→ select ename, replace (ename, 'A', 'z')
from emp;

second syntax :-

→ select ename, replace (ename, 'e')
from emp;

→ select replace ('Madhuria', 'M')
from dual;

O·p :- adhuria

→ select replace ('Gruna', 'a')
from dual;

O·p :- Grun

→ Substring :-

+,- values consider

syn 1 :- substr (source, pos);

+ values consider

syn 2 :- substr (source, pos, length);

→ select substr ('Good morning', 10)
from dual;

SUB
ing

-12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1
Good Morning
1 2 3 4 5 6 7 8 9 10 11 12

Direction →

→ select substr ('Good morning', 8)
from dual;

sub
string

syntax :- 2 :-

→ select substr ('good morning', 6, 2)
from dual;

O.P :- MO

→ select substr ('good morning', 6, 4)
from dual;

O.P :- morn

→ select ename substr ('ename', 1, 4)
from emp;

ename	SUBS
SMITH	SMIT
ALLEN	ALLE

→ date functions :-

Current_date :- it will show system date

sys date :- it will show Oracle date

months_between :- Calculate One month to another month.

→ Current_date :-

Select current_date
from dual;

Current_D
07-Nov-22

→ select sysdate
from dual;

SYSDATE
07-Nov-22

→ select sessiontimezone
from dual;

Sessiontimezone
+05:30

→ select months_between ('07-nov-22', '01-jan-22')
from dual;

O·P :- 10.1935484

→ select months_between (current_date, '01-jan-22')
from dual;

O·P :- 10.2056773

→ Write a query to display employ name number of years of experience of all the employee?

Select ename, months_between (current_date, hire date)/12
from emp;

→ Conversion ():-

To_date ():-

syntax :- To_date ('01/10/22', 'DD/mm/yy')

→ select to_date ('01/10/22', dd/mm/yy')
from dual;

O·P :- Oct-01-22
01-oct-22

→ select to_date ('01/10/22', 'mm/dd/yy')
from dual;

O·P :- Oct-01-22

→ Select to_date ('01/10/22', 'yy/mm/dd')
from dual;

O·P :- 22-Oct-01

→ select to_date ('01/10/2022', 'dd/mm/yyyy')
from dual;

O·P :- 01-Oct-22.

→ Multirowfunction :-

→ select max (sal) → inside select clause we can use
from emp
where job in ('salseman', 'manager');

→ select *
from emp
where max (sal); → inside where clause we cannot use

Error : group function is not allow here.

→ Write a Query to fetch max salary?
Select max (sal)
from emp

→ write a Query to fetch to max salary among the clerk?
Select max (sal)
from emp
where job = 'CLERK';

→ Write a Query to fetch max sal among deptNo 10 & 30?

```
Select max (sal)
from emp
where deptNo In (10,30);
```

→
```
select max (HIREDATE)
from emp;
```

→
```
select max (EMPNO)
from emp;
```

→ min ():-

→ write a Query to fetch min sal?

```
select min (sal)
from emp;
```

→ Write a Query to fetch min salary among the Salseman?

```
select min (sal)
from emp
where job = 'SALESMAN';
```

→ write a Query to fetch min sal among the smith, allen, ward.

```
select min (sal)
from emp
where ename In ('SMITH', 'ALLEN', 'WARD');
```

→ avg() :- [U cannot find the one person details]

→ write a Query to fetch avarage sal?

```
select avg (sal)
from emp;
```

→ write a Query to fetch avarage salary among the salseman?

```
Select avg (sal)
from emp
where job = 'SALSEMAN';
```

Sum function :-

→ Select sum (sal)
from emp;

sum (SAL)
29025

→ Select max (sal) + sum (SAL)
from emp;

max (SAL) + sum (sal)
34025

→ select max (sal), min (sal)
from emp;

max (sal)	MIN (sal)
5000	800

→ Count :-

How many "rows" are available in the table or column.

→ select count (comm)
from emp;

Count (comm)
4

→ select count (mgr)
from emp;

Count (mgr)
13

→ select count (empno)
from emp;

count (empno)
14

→ select count(*)
from emp;

→ single row function can be used in select clause, where clause group by and having clause and etc

→ multi row function can be used in select group by having but multirow function cannot be used in the where clause.

Sub Query:

→ Query inside the Query is known as "Sub Query".

→ when we are moving for an a subQuery is based On One Query results we are writing another Query.

syn 1:- select
from table name
where condition (sub Query);

syn 2:- select
from (sub Query)
where condition.

→ Write a Query to fetch all the details of the employee who are getting maximum salary?

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

→ select ename
from emp
where sal = (select max(sal)
from emp);

→ Write a Query to fetch all the details of the employ to get min salary?

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

→ write a Query to display all the details of the employee who join company first?

```
select *
from emp
where hiredate = (select min(hiredate)
                  from emp);
```

→ Write a Query to fetch all the details of the employee who are working in sales department?

```
select *
from emp
where deptNo = (select deptNo
                from dept
                where dname = 'SALES');
```

→ Write a Query to fetch all the details of employe who are working in research department?

```
Select *
from emp
where deptNo = (select deptNo
                from dept
                where dname = 'RESEARCH');
```

→ Write a Query to fetch all the details employee who are working in newyork city?

```
(select *
from emp
where deptNo = (select deptNo
                from dept
                where loc = 'NEWYORK');
```

→ Write a Query to display deptName, loc of employee SMITH and ALLEN?

```
select dname, loc
from dept
where deptNo In (select deptNo
                 from emp
                 where ename In ('SMITH',
                                 'ALLEN'));
```

→ write a Query to fetch all the details of employee whos loc consist of at least one 'a' in it?

```
select *
from emp
where deptNo In (select deptNo
                 from dept
                 where Loc like '%a%');
```

→ Write a Query to fetch all the details of the employe who got second highest salary?

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

→ Write a Query to fetch second highest sal?

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

→ **Order by:**

→ if Order is not specified default Order is ascending Order

→ when you pass 2 or more columns it will arrange the based On previous data.

syntax:- select
from tablename
where condition
Order by columnName asc/desc,...
columnName asc/desc.....

→ select sal
from emp
Order by sal asc;

→ select sal
from emp
Order by sal desc;

→ select sal
from emp
Order by sal ;

→ select sal, ename
from emp
Order by sal desc, ename desc;

→ select ename, sal
from emp
Order by ename asc, sal desc;

→ select ename, sal
from emp
Order by ename asc, sal desc
where ename in ('SMITH', 'ALLEN', 'WARD', 'RAM', 'FORD');

Error : SQL Command not properly endded.

→ select ename
from emp
Order by ename;

→ insert into emp (ename, empno)
values ('ram', 101);

→ select ename
from emp
Order by ename desc;

→ select sal, ename
from emp
Order by sal desc,
ename;

→ Select ename, sal
from emp
where ename in ('SMITH', 'ALLEN', 'WARD', 'RAM', 'FORD')
Order by ename asc, sal desc;

→ rownum :-

→ rownum is One of "veptual" column.
→ Every time start with an a first row.
→ it will work "<, <=".

Syn:- select....
from tableName
where condition.

→ select *
from emp
where rownum >=5; → No rows selected

→ select *
from emp
where rownum <=5; → 5 rows selected

→ select *
from emp
where rownum <5; → 5 rows selected

→ select *
from emp
where rownum >= 6; → No rows selected

→ select *
from emp
where rownum >= 8; → No rows selected

→ select *
from emp
where rownum > 1; → No rows selected

→ select *
From emp
where rownum >= 1; → 15 rows selected

→ select *
from emp
where rownum = 1; → 1 row selected

→ select *
from emp
where rownum < 100; → 15 rows selected

→ distinct :-

→ Remove duplicate values.

→ fetching the Unique values.

syn :- select distinct columnName
from tableName;

→ select distinct sal
from emp;

→ select distinct job
from emp;

→ select distinct job, sal
from emp;

→ select distinct sal, job, deptno
from emp;

→ select distinct *
from emp;

sal	job
2000	clerk ✓
1000	salseman ✓
2000	clerk ✓ → Duplicate
2000	salseman ✓
2000	clerck → Duplicate
3000	clerck ✓
1000	manager ✓
1000	salseman → Duplicate

→ Write a Query to fetch third highest salary?

```
select sal
from (select sal
from (select distinct sal
from emp
order by sal desc)
where rownum <=3
order by sal asc)
where rownum <=1;
```

→ Write a Query to fetch all the details of the employee who get third highest salary?

```
Select *
from emp
where sal = (select sal
from (select sal
from (select distinct sal
from emp
order by sal desc)
where rownum <=3
order by sal asc)
where rownum <=1);
```

→ Write a Query to 4th highest sal?

```
Select sal
from (select sal
from (select distinct sal
from emp
Order by sal desc)
where rownum <=4
Order by sal asc)
where rownum <=1;
```

→ write a Query to fetch all the details of employee who are getting more than avarage salary?

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

→ write a Query to fetch all the details of the employee who get salary more than avarage salary of deptNo 20.

```
Select *
from emp
where dept=20 and sal > (select avg (sal)
from emp;
```

→ write a Query to fetch all the details of the Allens manager?

```
Select *
from emp
where empNo = (select mgr
               from emp
               where ename = 'ALLEN');
```

→ write a Query to display location of the allen, ward, and scott?

```
Select loc
from dept
where deptNo In (select deptNo
                 from emp
                 where ename In ('ALLEN',
                                 'WARD', 'SCOTT'));
```

→ write a Query to display employe num and name who are working as a clerk and earning highest salary among the clerk?

```
Select EmpNo, ENAME
from emp
where job = 'CLERK' and sal = (select max(sal)
                               from emp
                               where job = 'CLERK');
```

→ Write a Query to display employe name of the salseman who are earning salary more than the highest salary of clerk?

```
Select ename
from emp
where job = 'SALSEMAN' and sal > (select max(sal)
                                  from emp
                                  where job = 'CLERK');
```

→ Write a Query to fetch all the details of the employee whose getting salary greater than ward?

```
Select *
from emp
where sal > ( select (sal)
              from emp
              where ename = 'WARD');
```

Joins :-

→ To Fetch the data from the multiple table.

(1) Cross join

(2) inner join
- self join
- equi join

(3) Outer join
- Left join
- Right join
- full Outer join.

→ Cross join :-

→ don't need any connection One table to another table

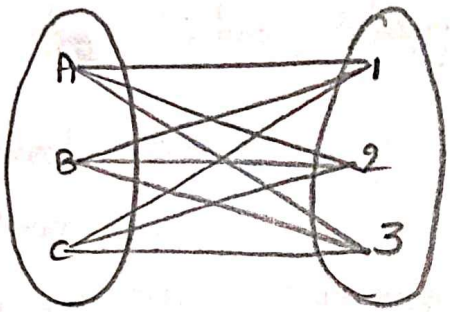
→ Compare One record with every record

→ where condition is not mandatory.

Syntax :- Select table.col, table.col....
From table 1, Table 2;

(or)

Select table.col, table.col...
From table 1 cross join Table 2;



O·P

A_1	1A
A_2	2A
A_3	3A
B_1	1B
B_2	2B
B_3	3B
C_1	1C
C_2	2C
C_3	3C

→ Select emp.ename , stu.sname
from emp , stu;

→ Select emp.ename , stu.sname
from emp , stu
where empname = 'SMITH';

→ select emp.ename, stu.sname
from emp , stu
where stu.sName = 'RAVI';

→ select e.ename, d.dname
from emp e, dept d

→ Write a Query to display all the details of Courses which abc student will grow throw in fullstack development?

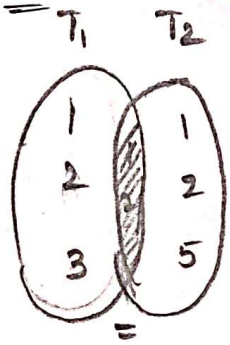
Select S.sname, C.*
from student S, Course C
where S.sname = 'abc';

→ Write a Query to display student name who are done with an a job Python?

```
Select s.sname , c.name
from stu s , cpurse c
where c.cname = 'JAVA';
```

→ Primary Key :-

→ Equi join :-



→ Connection is compulsory

→ equi join done by common column exist

→ if the connection between two tables we will do the equi join

→ where condition is mandatory.

→ equi join done by "=" symbol.

Syntax :- Select table.coloum , table.col...
From table 1, table 2
where table 1.commcol = table 2.commcol;

→ Select e.name, e.deptNo, d.deptNo, d.dname
from emp e, dept d
where e.deptNo = d.deptNo;

→ Select S.SName, S.CID, C.CID, C.Name
From STU.S, Cour C
where S.CID = C.CID;

→ Write a Query all the details of the student and current course details?

```
Select S.*, C.cname
from stu s, cour c
where S.CID = C.CID;
```

→ Write a Query to fetch all the details of employee and they current working location?

```
Select e.*, d.loc
from emp e, dept d
where e.deptNo = d.deptNo;
```

→ Write a Query to fetch all the details of the Smith and allen and display the deptname and their loc?

```
Select e.*, d.dname, d.loc
from emp e, dept d
where e.deptNo = d.deptNo and e.ename In
('SMITH', 'ALLEN');
```

→ display all the details of the salseman and manager and display they current working location?

```
select e.*, d.loc
from emp e , dept d
where e.deptNo = d.deptNo and e.job in ('SMITH', 'MANAGER');
```

→ Inner join :-

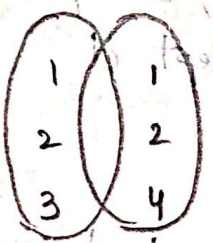
→ Combination of equi join.

→ Every time compare one record to another record

→ where condition is mandatory

→ Connection is compolsory to fetch data.

→ =, !=, <, >, <=, >=



Syntax :- select table.col, table.col ....
From table 1 innerjoin table 2
on table 1.commcol = table 2.commcol;

!=
<=
>=
<>

optional :- select table.col, table.col....
From table 1 join table 2
On table 1.commcol = table 2.commcol;

>=
!=
<=

→ Select e.ename, e.deptNo, d.deptNo, d.dname
from emp e join dept d
on e.deptNo = d.deptNo;

→ Select e.ename, e.deptNo, d.deptNo, d.dname
from emp e join dept d
on e.deptNo != d.deptNo;

→ Select e.ename, e.deptNo, d.deptNo, d.dname
from emp e join dept d
On e.deptNo <= d.deptNo;

→ Select e.ename, e.deptNo, d.deptNo, d.dname
from emp e join dept d
On e.deptNo >= d.deptNo;

→ Select e.ename, e.deptNo, d.deptNo, d.dname
from emp e inner join dept d
On e.deptNo > d.deptNo;

→ Select e.ename, e.deptNo, d.deptNo, d.dname
from emp e inner join dept d
On e.deptno < d.deptno;

→ Write a Query to fetch all the details of the employee and they current working location?

```
Select e.*, d.loc
from emp e join dept d
On e.deptNo = d.deptNo;
```

→ Write a Query to fetch all the details of the saleman and manager and display the current location?

```
Select e.*, d.loc
from emp e join dept d
On e.deptNo = d.deptNo
where e.job in ('SALSEMAN', 'MANAGER');
```

→ Write a Query to fetch all the details of the smith and allen and display they department name and their loc?

```
Select e.*, d.dname, d.loc
from emp e join dept d
On e.deptNo = d.deptNo
where e.ename in ('SMITH', 'ALLEN');
```

→ Write a Query to display all the details of the ram and display rest of the course still he left to do?

```
Select s.*, c.cname
from stu s join course
On s.CID != c.CID
where s.sname = 'RAM';
```

Self join :-

→ A join it self is known as self join.

→ self join is possible when there are Common Columns in the table.

Syntax :-

```
Select table.col, table.col
from table t1, table t2
where t1.commcol = t2.commcol;
```

→ Write a Query to display employee name, salary job and manager name of employee smith ?

```
Select e1.ename, e1.sal, e1.job, e2.ename
from emp e1, emp e2
where e1.mgr1 < e2.empNo and e1.ename = 'SMITH
```

→
```
Select *
from Emp e1, emp e2
where e1.mgr1 = e2.empNo and e1.ename = 'SMITH
```

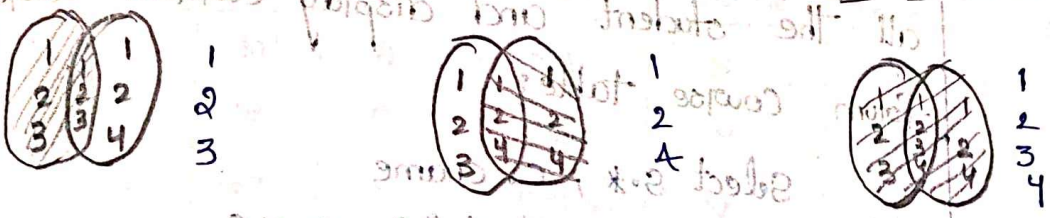
→
```
Select e1.ename, e1.sal, e1.job, e2.ename, e2.sal
                                          e2.job
from emp e1, emp e2
where e1.mgr1 = e2.empNo and e1.ename = 'SMITH';
```

→
```
Select e1.ename, e1.ename || ' manager is ' || e2.ename
from emp e1, Emp e2
where e1.mgr1 = e2.empNo and e1.ename = 'SMITH';
```

→ Select e_1.ename || 'manager is' || e_2.ename
from emp e_1, emp e_2
where e_1.mgr = e_2.empNo;

→ Outer join

left join | right join | full outer join



ओ

left join :-

Left join will fetch common data and also fetch the remaining data from left table.

syntax :- Select table.col, table.col
from table 1 left join table 2
On table 1.commcol = Table 2.commcol;

right join :-

Right join fetches common data and also fetching remaining data from right table.

syntax :- select table.col, table.col
from table 1 right join table 2
On table 1.commcol = Table 2.commcol;

Full Outer Join:-

Full Outer Join fetchs "common data" and also fetches all the "remaining data from the two tables.

→ Write a Query to display all the details of all the student and display current course from course table?

```
Select S.*, C.cname
from stu s left join cour c
On s.cid = c.cid;
```

→ Write a Query to display all the details of the student course currently and display all the course details display which are availa ble in the institute?

```
Select s.*, c.cname
from stu.s right join cour c0
On s.cid = c.cid;
```

→ write a Query to display all the details of the all the students all the courses details which are available institute?

```
Select S.*, C.*
from stu s full Outer join cour c
on S.sid = c.cid;
```

→ **Normalization :-** [we are follow the Normalization rules to create a table]

(1) insertion Anomaly
(2) updation Anomaly
(3) deletion Anomaly

Eid	sid	sname	Cname	Tname	marks
1	101	A	sql	X	60
1	102	B	sql	X	70
2	103	C	Python	Y	80
2	104	D	python	Y	90
3	102	E	java	Z	95
3	106	F	java	Z	75

cid	cnam	T&i

sid	Sname	marks	cid

Normalization dividing larger table into the smaller table and link them using relationship as known as normalization

→ it removes insertion anomaly, updation anomaly, deletion anomaly.

Normalization 3 types :-

1) 1NF
2) 2NF
3) 3NF

→ 1NF :-

sid	sname	marks	cname
101	A	60	c++,c
102	B	70	java
103	C	80	python
104	D	90	sql

sid	sname	marks	c_1	c_2
101	A	60	C	C++
102	B	70	java	
103	C	80	python	
104	D	85	sql	

↓

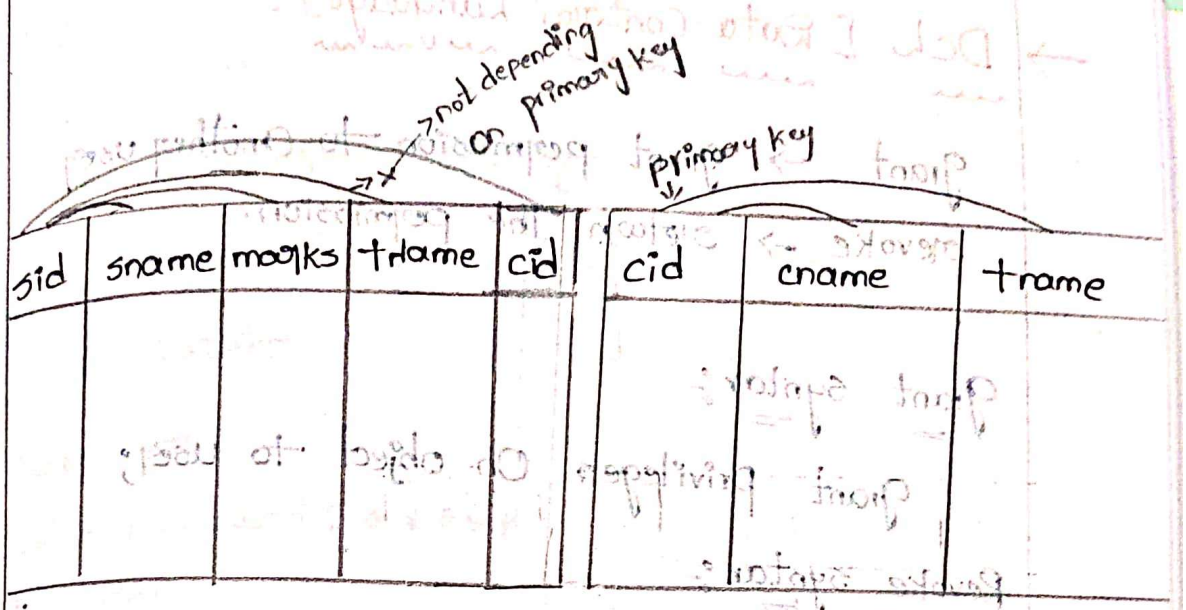
This table is not in Normalization

→ single attribution should be present
→ Unique column name should be present.
→ Order doen't matter.

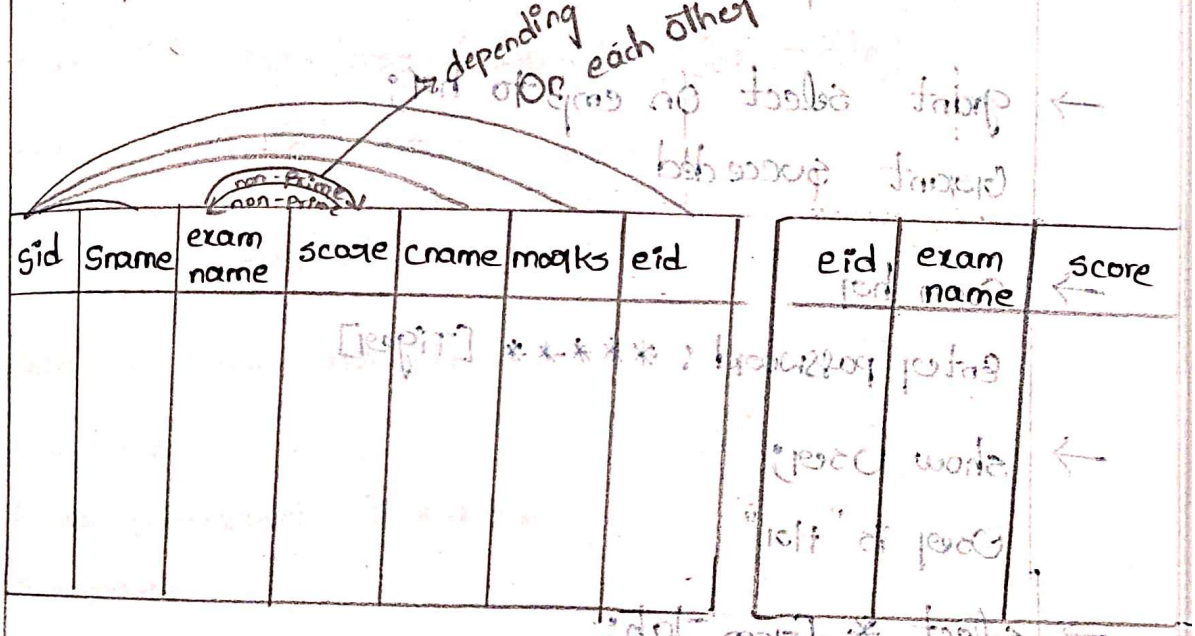
→ 2NF :-

→ Follow the first normal form.
→ no partially dependence.

[partially dependence means every column should be present depending on primary column if any column not depending on primary key column it is called as a partially dependency.]



→ 3NF:



→ Follow the second normal form.

→ No transitive dependence.

[It means non-prime column and non-prime column depending on each other is known as transitive dependence]

[non prime means not a primary key column]

→ Dcl [Data Control Language] :-

grant → grant permission to another user
revoke → return the permission.

Grant syntax :-

grant privileges on object to user;

Revoke syntax :-

revoke privileges on object from user;

→ grant select on emp to hr;
Grant succeded

→ Conn hr
Enter password : ***** [Tiger]

→ show user;
User is "HR"

→ select * from tab;

→ Select * from emp;
Error at line 1
table or view does not exist

→ select * from scott.hr;
Error at line 1
Table or view does not exist

→ select * from scott.emp;

→ insert into scott.emp (empno)
values (101);
Error : Insufficient privilages.

→ grant select, insert On jobs to scott;
grant succeed

→ Conn scott
Enter password : *****
Connected

→ select *, from hr.jobs;

→ insert into hr.jobs (job_id, job_title)
values ('t_id', 'hr');

1 row Created

→ select * from hr.jobs;

→ Conn hr
Enter password : *****
Connected

→ select * from jobs;

→ Revoke select On jobs from scott;
Revoke succeeded

→ Revoke all On jobs from scott;
Revoke succeeded

→ Conn scott
Enter password : *****
Connected

→ Select * from hr.jobs;

Error at line 1

Table or view doest exist

→ Revoke all on emp from hr;

Revoke succeed

One user to another user :-

✱ syntaxes :-

(1) Create User UserName identified by password;

(2) grant Connect to newUser;

(3) grant all privilegen to newUser;

(4) drop User Username Cascade;

→ Creat User MNO identified by m123;

Error :

insufficient privileges

→ Conn system

Enter password : ****

Connected

→ Create User mno identified by m123;

User Created

→ Select * from all users;

→ Conn mno

Enter password

Error :- User mno lacks create session privilage log
#defined. you r no longer connected to

→ Conn system.
Enter password : *****
Connected.

→ grant Connect to mno;
grant succeeded

→ Conn mno
Enter password : *****
Connected

→ Create table t1 (
tid number (10));

Error: insufficient privileges

→ show user;
User is "MNO"

→ Conn system
Enter password : *****
Connected

→ grant all privileges to mno;
grant succeeded

→ Conn mno
Enter password : *****
Connected

Views :-

→ View is One of vertual table.
→ view is the resultant set of stored Query
→ Take advantages of the table :-
→ To restrict the data access [security]
→ To make Complex Query easy
→ To provide data idependency.
→ To present different views of same datas.
→ Only give permission "DML" Commands.

index :-

syntax :- Create view viewName
as
select .....
from tableName;

→ Conn system
Enter password : * ****
Connected.

→ Grant Create view to scott;
Grant Succeeded

→ Conn scott
Enter pass word : *****
Connected

→ Create view v3
as
select sal, job, comm
from emp;

→ select * from table;

Index:-

syn:→ Create index indexName On tableName (columnName);

syn:→ drop index indexName;

Defination:-

→ we will use the index when we are putting the frequently condition On a same column we will Create a index

→ index is Created for "quick search"

→ it will reduce The cpu load / cost.

2 types of index:-

(1) clustey index

(2) non-clustey index

→ **Clustey index:-**

→ in a clustey index we can Create One index in a table

→ clustey index Fastey than the non-clustey index

→ in clustey index we will stoge datas in the "sOrted Orde".

→ **non-clustey index:-**

→ in a non-clustey index we can Create One ay moge index in a table

→ non clustey index is slowey than the clustey index.

→ non cluster index will not store the data in sorted order so that it is taking more space

→ select *
from emp
where deptNo = 10;

→ select *
from emp
where job = 'SALSEMAN';

→ **Set** :-

(1) Union :- fetch Uniaue value

(2) Unionall :- It will fetch the data all duplicate values also.

(3) Intepsect :- It will fetch the Common column

(4) minus :- It will not fetch the common data.

→ Select deptNo from emp
Union
select deptNo from dept;

O.P
10
20
30
40
60

→ select deptNo from emp
union all
select deptNo from dept;

O.P
10
20

→ select deptNo from emp
Intepsect
select deptNo from dept;

O.P
10
20
30

→ Select

deptNo
40
60

minus

select deptNo from emp;

Set defination:-

Set Command is used to update to specify which column and values that should be update in a table.

→ Group by :-

→ Grouping the data

→ we will use multirow function

syntax :- ① select
② tableName
③ where (condition)
④ Group by
⑤ having
⑥ Order by

→ select dept No, Count (*)
from emp
group by dept No;

Dept No	count
30	6
20	5
10	3

→ select job, count (*)
from emp
group by job;

job	Count (*)
clerk	4
salseman	4
president	1
manager	3
analyst	2

→ Select deptNo, job, count(*)
from emp
group by deptNo;

Error: not a group by expression.

→ Select deptNo, job, count(*)
from emp
group by deptNo, job;

→ select max(sal), deptNo
from emp
group by deptNo;

max(sal)	deptNo
2850	30
3000	20
5000	10

→ select deptNo, job, count(*)
from emp
group by deptNo, job
having count(*) >= 2;

DeptNo	job	count
20	clerk	2
30	salesman	4
40	Analyst	2

→ select deptNo, job, count(*)
from emp
group by deptNo, job
having count(*)>=3;

→ select deptNo, job, count(*)
from emp
having count(*)>=3;

example

→ select deptNo, job, count(*)
from emp
where job in ('SALSEMAN', 'MANAGER', 'CLERK')
group by deptNo, job
Order by job Asc;

→ select dept No, job, count(*)
from emp
where job in ('SALESEMAN', MANAGER', 'CLERK')
group by deptNo, job
having count(*)>=2
Order by job asc;

→ Select the depaytments which consist of more than three people and groups all the departments 10, 20, 30.

```
select deptNo, count(*)
from emp
group by deptNo
having count(*) >= 3;
```

kill U of success
Bury with U of smile

kill them with U of success
Bury them with U of smile