

★ 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(s)ename to Name;

Table is altered

→ alter table emp

rename column deptno to did;

Table is 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 decrease 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); qm2 stdnt psln
- table altered [stdnt stdt stdt] : qpsq
- DESC EMP: empno slot qrcd : n/a
 EMPNO mid stop words : 8 n/a
 ENAME empno slot hood doff : 8 n/a
 JOB qrcd
 MGR empno slot spcng : 4 n/a
 HIREDATE mid stoper spcng : 8 n/a
 SAL push back table ptn but no reqd
 COMM dot mact & fslb
 DEPTNO tblig slot qrcd
- alter table emp
 modify ename varchar(6); bqqrd slot
 table altered. bqqrd slot
- alter table emp
 modify ename varchar(5); tblig slot spcng
 cannot decrease column length because some value is too big. mid stopage words
- alter table emp
 modify cname number(10) unique; spcng slot
 Table altered mid stopage words
dot mact & fslb

→ alter table emp

(a) partion errors problem
old partition

★ 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 jspider;

Table dropped

→ drop table worker;

Table dropped

→ select * from tab;

→ show recycle bin;

→ purge table jspider;

→ 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 recyclebin; object deleted from recyclebin
- flashback table pto before drop; flashback complete
- select * from tab;
- show recyclebin; table recovered from recyclebin
- recyclebin purge; table *
- select * from tab; object not found in table
- show recyclebin; object not found in table
- purge table pto from recyclebin; object not found in table at line 7: object not in recycle bin.
- drop table course; no object found in table

Truncate: Delete all the records permanently.

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 temporary, we can't undo the delete operation.

- Delete is slower than Truncate.
- In delete we will use where clause to delete specified record.
- Delete is "DML" command.

* Truncate:

- Truncate is DDL Command
- Truncate will delete the all records permanently.

→ 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 permanently.

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

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

→ SQL Functions:

→ SQL function is used to do some specific task. These are the built-in functions.

There are two types:-

(1) single row function

(2) multi row function.

→ single row function: it works for every 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~~

<u>number functions :-</u>	<u>char functions :-</u>	<u>date/time functions :-</u>	<u>Conversion functions :-</u>
power()	length()	months_between()	To_date()
mod()	substr()	current_date()	
abs()	upper()	sessiontimezone()	
sign()	lower()	sysdate()	
floor()	initcap()	date format	
ceil()	replace()	of type not null	
round()	trim()	stop at	

Multi row functions :-

- Max() :- finds if there are more than one
- min() :- finds if there are less than one
- Avg() :- average of all rows
- sum() :- adds all rows
- count() :- counts the number of rows
- power() :- [pass the 2 values]
 - Select power(5,8) :- were slipper
 - from dual; :- were slipper
- Select ename, power(sal,2) :- were item
- from emp; :- were slipper
- Select power(4.6,3) :- [pass the 2 values]
 - power(46,3) :- were item
 - from dual; :- were item

- $\rightarrow \text{select } 56, 3 \rightarrow \text{power}(56, 3)$
 from dual; 15616
- $\rightarrow \underline{\text{Mod ()}}$
 $\rightarrow \text{select sal, mod(sal, 2)}$
 from emp;
- $\rightarrow \text{select mod}(45, 3)$
 from dual;
 $O.P \approx 0$
- $\rightarrow \text{select mod}(65, 89, 3)$
 from dual; $O.P \approx 2.89$
- $\rightarrow \text{select mod}(-34.5, 2)$
 from dual; $O.P \approx -5$
- $\rightarrow \underline{\text{Sqrt}}$ [only one number, pass] (sqrt) 49
- $\rightarrow \text{select sqrt}(625)$
 from dual;
sqrt
 ≈ 25
- $\rightarrow \text{select sqrt}(121)$
 from dual;
 $O.P \approx 11$
- $\rightarrow \text{Select sal, sqrt(sal)}$
 from emp;
- $\rightarrow \text{select sqrt}(25)$
 from dual;
 $O.P \approx 5$
- (sqrt) odd table
(sqrt) even
first number always big %, second number small
- $\rightarrow \underline{\text{(ce.)}}$ (ce.) 89
- $\rightarrow \text{select ce}(89)$
(ce.) odd table
(ce.) even
- $\rightarrow \underline{\text{(ce.)}}$ (ce.) 0.89
- $\rightarrow \text{select ce}(0.89)$
(ce.) odd table
(ce.) even
- $\rightarrow \underline{\text{(sqrt)}}$ (sqrt) 49
- $\rightarrow \text{select sqrt}(49)$
 from dual;
 $O.P \approx 7$
- $\rightarrow \text{select sqrt}(56.45)$
 from dual;
 $O.P \approx 7.5133215$
- $\rightarrow \underline{\text{(sqrt)}}$ (sqrt) -243
- $\rightarrow \text{select sqrt}(-243)$
 from dual;
error line: 1 argument out of the range
- $\rightarrow \underline{\text{(ppp+)}}$ (ppp+) 1000
- What concept?
Never accept negative values

- $\rightarrow \underline{\text{abs()}}$:- ("values" converted into "positive" values) ↳ looks simple
 $\rightarrow \text{select abs (45.79)}$
 from dual;
 $\underline{\text{Abs (45.79)}}$
 $\underline{45.79}$ ↳ (45.79) looks like 45.79
 $\rightarrow \text{select abs (-35)}$
 from dual;
 $\underline{\text{Abs (-35)}}$
 $\underline{35}$ ↳ (-35) looks like 35
 $\rightarrow \text{select abs (23)}$
 from dual;
 $\underline{\text{Abs (23)}}$
 $\underline{23}$ ↳ (23) looks like 23
 $\rightarrow \underline{\text{floor()}}$:- [floor every time will give down value] ↳ looks simple
 $\rightarrow \text{select floor (4.1)}$
 from dual;
 $\underline{\text{floor (4.1)}}$
 $\underline{4}$ ↳ (4) looks like 4
 $\rightarrow \text{select floor (4.8)}$
 from dual;
 $\underline{\text{floor (4.8)}}$
 $\underline{4}$ ↳ (4) looks like 4
 $\rightarrow \text{select from (4.999)}$
 from dual;
 $\underline{\text{floor (4.999)}}$
 $\underline{4}$ ↳ (4) looks like 4

- $\text{ceil}()$:- [ceil every time will give high value]

→ Select ceil (4.2) from dual; → signal

CEIL (4.2)

5

→ select ceil (4.99) from dual; ← (ceil) aligned table

CEIL (4.99)

5

→ select ceil (-4.5) from dual; ← (ceil) aligned table

CEIL (-4.5)

-4

→ $\text{ground}()$:- [floor and ceil combine ground]

→ select ground (4.999) from dual; ← (ground) aligned, signed table

Round (4.999)

5

→ select ground (-4.2) from dual; ← (ground) aligned, signed table

Round (-4.2)

-4

→ select ground (-4.5) from dual; ← (ground) aligned, signed table

Round (-4.5)

-5

char functions

→ length :-

→ Select length ('hello')
from dual;

O.P : 5

→ Select length ('hellow --')
from dual;

O.P : 8

→ select ename , length (ename)
from emp;

→ select job , length (job)
from emp; [because strings has blank part.]

→ select empno , length (empno)
from emp;

→ Upper () :-

It convert lower to upper case . if provide

Query is already upper (it convert it to upper)

→ select upper ('Hellow')
from dual;

O.P : HELLOW

→ select upper ('HEllow')
from dual;

O.P : HELLOW

(S.P) 100% true
about project

(S.P) 100%

(PP.P) 100% includes
about project

(PP.P) 100%

(S.P.) 100% true
about project

(S.P.) 100%

(PPP.P) 100% true
about project

(PPP.P) 100%

It convert lower to upper case . if provide

(S.P.) 100%

(S.P.) 100% true

about project

(S.P.) 100%

- lower() :-
- Select Lower ('HOLLOW')
 - From dual;
 - O.p :- hollow ('HOLLOW')
 - Select Lower ('hollow')
 - From dual;
 - O.p :- hollow ('hollow')
- Trim() :-
- It remove the extra space starting and ending
 - ending And starting letter will be same trim
 - work On starting and pending letter remove.
 - select trim ('-hi hellow ---').
 - From dual;
 - TRIM ('HI' 'HEL')
 - hi hellow ('O' 'nove')
 - select trim ('h' from 'heell')
 - 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;
- select ename, replace (ename, 'e')
from emp;
- select replace ('Madhusua', 'M')
from dual;
O.p :- adhuwa
- select replace ('Guna', 'a')
from dual;
O.p :- gun
- SubString :-
syn 1 :- substr (source, pos);
syn 2 :- substr (source, pos, length);
→ select substr ('Good morning', 10);
from dual; O.p :-
- SUB
ing
- RE
lection

- select substr ('Good morning', 1, 8) from dual;
 : Subs correct
Sub
String
- syntax :-
~~= =~~
- select substr ('good morning', 6, 2) from dual;
 : Subs correct
good
morning
 08 : 30
- O.P :- MD
 (SS-Von-FO, SS-Von-FO) Oracle - column tools
- select substr ('good morning', 6, 4) from dual;
 : Subs correct
good
morning
 paper - 01-9-0
- O.P :- morin
 (SS-Von-FO, SS-Von-FO) Oracle - column tools
- select ename substr ('ename', 1, 4) from emp;
 : Subs correct
ename
ename
 ename
- ename subs
 SMITH SSMITH
 ALLEN ALLN
- Oracle - 01-9-0
- 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 is (10-10-2022) Subs correct
 Select current_date is (10-10-2022) to Current_D from dual;
 from dual;
- 07-Nov-22
 08-10-2022
 08-10-2022

→ `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 :- 10.2056713

→ 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 () :- date converted to date

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.
- Multiflow function:
- select max(sal) from emp
 where job in ('salesman', 'manager');
- select * from emp
 where max(sal);
- Error : group function is not allowed here.
- Write a query to fetch max salary?
 Select max(sal)
 from emp
- write a query to fetch 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);
88-10-150 → 90
- 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 Salesman
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');

→ $\text{avg}()$ [U cannot find the one person details]

→ all the stations and "null" person will be
written a query to fetch average sal?

select avg (sal)
from emp;

→ write a query to fetch average salary
among the salesman?

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

Sum function:

→ select sum (sal)
from emp;

→ select max (sal) + sum (sal)
from emp;

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

→ max (sal) and min (sal)
will give us the highest and lowest value
among all the records.

- Count ~~is~~ will return count of rows
~~in table~~
How many "rows" are Available in the table or column.
(Ans) pro false
- Select count (comm)
from emp;
Count (comm)
4
(Ans) pro false
- select Count (mvr)
from emp;
Count (mvr)
13
(Ans) mvr true
- select count (EmpNo)
from emp;
Count (EmpNo)
14
(Ans) mvr + (Ans) xrm false
- select + Count(*)
from emp;
(Ans) mvr + (Ans) xrm false
- single row function Can be used in select clause, where Clause group by and having clause and etc
(Ans) mvr + (Ans) xrm false
- multi row function Can be used in select group by having but multirow function cannot be used in the were clause.
(Ans) mvr + (Ans) xrm false

Sub Query :- It is a query which is used inside another query.

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

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

to fetch all the details of query is as follows

Syntax :- select * from table name order by column name
where condition (Sub Query);

Syntax :- select * from (Sub Query) where condition.

→ Write a query to fetch all the details of the employee who is one getting maximum salary?

Select *

from emp

where sal = (select max(sal) from emp)

from emp;

→ Select ename, sal from

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

→ Write a Query to fetch all the details of the employee 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 (join dept) on deptno = deptno

where deptno = (select deptno

from dept

where dname = 'SALES');

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

(a) xam task > loc specify
Select *
from emp
where deptNo = (select deptNo
from dept
where loc = 'NEWYORK');

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

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

→ write a Query to fetch all the details of employee
whose 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 employee 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 salary. `sal < max(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 column it will arrange the based on previous data.

(don't still not specify)

Syntax:

Select

from table name

where condition

Order by column name

(asc/desc)

column name 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 ended.

→ Select ename, sal
from emp
where ename in ('SMITH', 'ALLEN', 'WARD', 'RAM',
'FORD')

Order by ename asc, sal desc;

→ Rownum-

rownum

→ Rownum is One of "virtual" Column.

→ Every time start with an a first row.

→ it will work "<, <=".

Syntax select

from tablename
where condition.

→ Select * from

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

• Before selecting for bracketed spec : parent

- 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	clerk ✓ → Duplicate
3000	clerk ✓
1000	manager ✓
1000	salseman ✓ → Duplicate

What a Gluey to fetch this highest salary?

select sal

from (select sal)

from (select distinct sal

from emp

Order by sal desc

where $\text{rownum} \leq 3$

Order by sal asc)

where $\text{Townum} \leq 1$;

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

Select *

from emp

where $\text{sal} = (\text{select } \text{sal}$

from Select sol

from (select distinct sal

from empirical to abstract or general to specific

Order by sal desc) ep odm sefopgcs 94

where $\pi_{own} = 3/4$ and $\pi_{other} = 1/4$

Order by sal asc)

where $\text{rownorm} <= 1$;

Digitized by srujanika@gmail.com

→ Write a Query to find highest sal?

Select sal

from (select sal

from (select distinct sal

from emp

Order by sal desc)

where rownum <= 1

Order by sal asc)

where rownum <= 1;

↳ false

(↳ false) more.

(↳ false) more.

↳ more.

(see loc pd pd)

↳ minwage speaks,

(see loc pd pd)

↳ minwage speaks

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

Select *

from emp

where sal > (select avg(sal)

from emp);

↳ false

↳ more.

(↳ false) more.

(↳ false) more.

(↳ false) more.

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

Select *

from emp

where dept=20 and sal > (select avg(sal)

from emp);

(see loc pd pd)

(↳ minwage speaks,

(see loc pd pd)

→ write a query to fetch all the details of the Allen's manager? (also named as Select * from emp to get details of all 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 employee 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 employee name of the salesman who are earning salary more than the highest salary of clerks →

Select ename from sales - qms mark

from emp - qms mark

where job = "SALESMAN" and sal > (Select max(sal)

sal to. without problem of from emp ←

{ Note here, where job = 'CLERK' };

→ Write a query to fetch all the details of the employees whose getting salary greater than word ? qms mark

(Note) of some marks
Select *

from emp

where sal > (Select (sal) from to store ←

from emp select sal

where ename = 'HARO');

marked qms false

qms mark

Xem baba) = lca bao 'HARO') = do que

qms mark

(Note) of some marks

from emp

where ename =

→ 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';

↳ query is good

(Q1)

↳ didn't get the output of query is blank

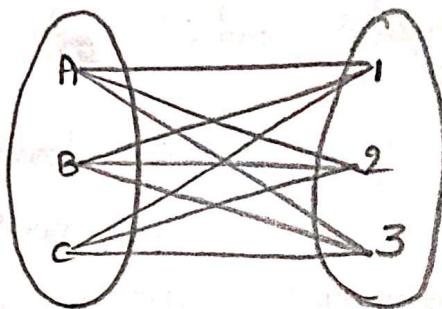
→ wrong Select table.col, table.col...

From table 1 Cross join Table 2;

↳ query is good

↳ second, it works well

↳ both are same & good



A₁ 1A
A₂ 2A
A₃ 3A

B₁, B₂, B₃ (1)

B₂, B₃ 2B

B₃ 3B

C₁, C₂, C₃ (2)

C₂, C₃ 3C

C₃

- 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 through in fullstack development?

Select s.sname, c.*

From student s , Course c

where s.sname = 'abc';

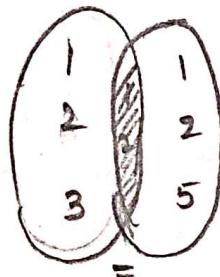
→ Write a query to display student name who
are done with a job Python?

Select s.sname, c cname
from stu s, Cpu c

where c cname = 'JAVA'

→ Inner Join :-

→ Equi join :- To establish set No group or strict



→ To establish set implicitly of group or strict

→ Connection is Compulsory of group or strict

→ Equi join done by common column exist

→ if the connection between two tables we will

do the equi join b tsk, e quite more

→ where condition is mandatory. e specia

→ Equi join done by '=' symbol.

→ Syntax :- Select table1.col1, table2.col2... from

from table1, table2 and print b

where table1.commoncol = table2.commoncol;

b task, e quite more

(if common bne obgkts = onlqkts gkts)

(inner join)

- 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, COURS 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, COURS c
where s.CID = c.CID;
- Write a Query to fetch all the details of
employee and their current working location?
- Select e.* , d.location
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.location
from emp e, dept d
where e.deptNo = d.deptNo, and eename In
('SMITH', 'ALLEN');

→ display all the details of the salesman and manager and display their current working location?

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

→ Inner join :-

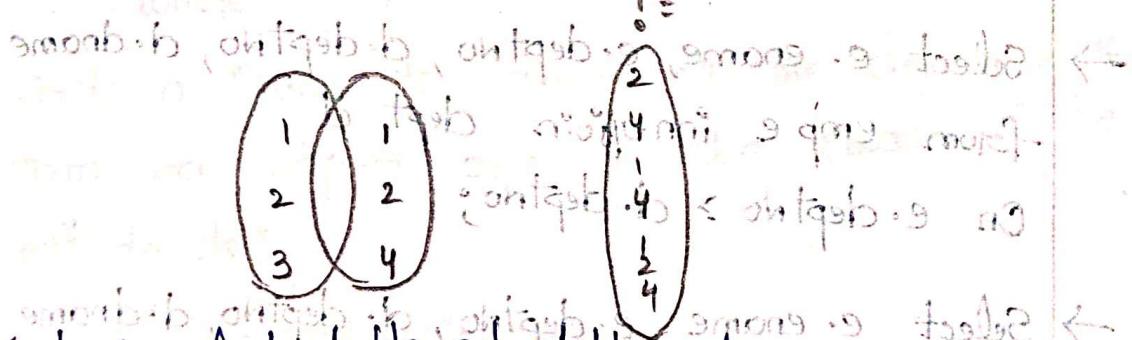
→ Combination of equal joins.

→ Every time compare one record to another record

→ where condition is mandatory from both

→ Connection is compulsory to fetch data.

→ =, !=, <, >, <=, >= on tables as



Syntax :- select table.col, table.col ...

From table1 innerjoin table2 morit

on table1.commoncol = table2.commoncol;

To display all the list of = in strict

optional :- select table1.col, table2.col ...

From table1 join table2

On table1.commoncol = table2.commoncol;

on table1.col = table2.col;

→ 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 salesman and manager and display the current location?
- Select e.* , d.loc
 from emp e join dept d on e.deptno = d.deptno
 where e.jobs in ('SALESMAN', 'MANAGER');
- Write a Query to fetch all the details of the smith and allen and display their department name and their loc. & year of entry
- 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 courses still he left to do?
- Select s.* , c.cname
 from student s join course c on s.cid = c.cid
 where s.sname = 'RAM';

- Self join
- As join it self is known as self join.
- self join is possible when there are common columns in the table.

Syntax

```
Select table.col, table1.col, ...
from table1, table2
where table1.commoncol = table2.commoncol;
```

- 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.mgr = e2.empno and e1.ename = 'SMITH'
```

- Select * from table1, table2

```
from Emp e1, emp e2
```

```
where e1.mgr = e2.empno and e1.ename = 'SMITH'
```

- Select e1.ename, e1.sal, e1.job, e2.ename, e2.sal

```
from emp e1, emp e2
```

```
where e1.mgr = e2.empno and e1.ename = 'SMITH'
```

- Select e1.ename, e1.ename || ' manager is ' || e2.ename

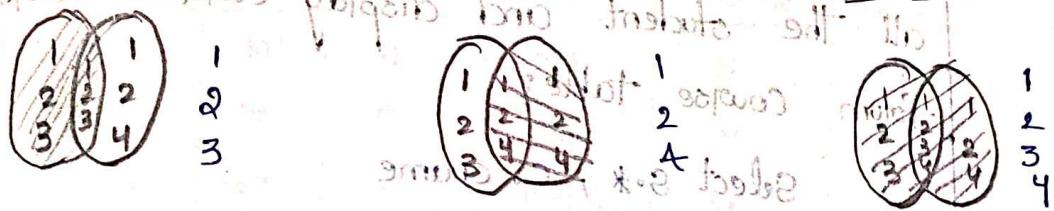
```
from emp e1, emp e2
```

```
where e1.mgr = e2.empno and e1.ename = 'SMITH'
```

→ select e₁.ename || ' manager is ' || e₂.ename
from emp e₁, emp e₂
where e₁.mgr = e₂.empno;

Outer join

left join set ||| right join prevQ. n shan
full outer join



3i

left join :-

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

Syntax :- Select table.col, table.col

from table1 leftjoin table2

On table1.commoncol = Table2.commoncol;

right join :-

right join :-

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

Syntax :- Select table.col, table.col

from table1 rightjoin table2

On table1.commoncol = Table2.commoncol;

right join :-

full Outer Join

full Outer join fetches "Common data" and also fetches all the ~~other~~ 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 either student course(s) currently and display all the course details display which are available in the institute?

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

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

Select s.* , c.*
From stu.s full Outer join cour c
On s.sid = c.cid;

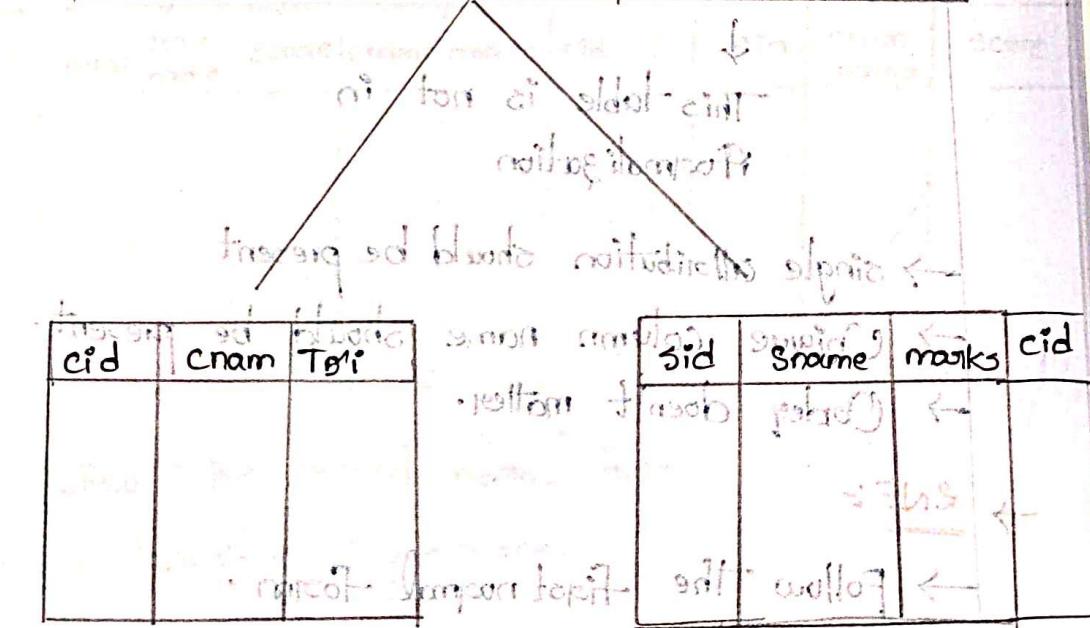
→ **Normalization :-** [we will follow the normalization rules to create a table]

(i) insertion Anomaly

(ii) updation Anomaly

(iii) deletion Anomaly

Eid	sid	Sname	Cname	Tname	Marks
1	101	A	Java	Opentxt	60
2	102	B	Python	X	70
2	103	C	Python	Y	80
2	104	D	Python	Y	90
3	105	E	Java	Z	95
3	106	F	Java	Z	75



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 is good

Normalization 3 types :-

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



1NF

sid	sname	marks	cname	sid	sname	marks	c ₁	c ₂
101	A	60	C++, C	101	A	60	C	C++
102	B	70	java	102	B	80	70	java
103	C	80	python	103	C	80	80	python
104	D	90	sql	104	D	85	85	sql

This table is not in normalization

→ single attribution should be present.

→ Unique column name should be present.

→ Order doesn'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.]

↑ not depending primary key

sid	sname	marks	tname	cid	cid	cname	tname
1	John	80	IT	1	1	Computer	IT
2	Jack	90	IT	2	2	Computer	IT
3	Jill	70	IT	3	3	Computer	IT
4	Tom	60	IT	4	4	Computer	IT

→ 3NF:
→ good match, both no depending attribute.

↑ not depending each other

sid	sname	exam name	score	cname	marks	eid	eid	exam name	score
1	John	English	80	Computer	80	1	1	English	80
2	Jack	Maths	90	Computer	90	2	2	Maths	90
3	Jill	Science	70	Computer	70	3	3	Science	70
4	Tom	IT	60	Computer	60	4	4	IT	60

- Follow the second normal form → false
 → No transitive dependence. I will do project

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

[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 succeeded

→

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.emp; Error at line 1
Table or view does not exist

→ Select * from scott.emp;

- insert into scott.emp (emp_no) values (101);
Error : insufficient privileges.
Grant select, insert on jobs to scott;
Grant succeeded
Conn SCOTT
Enter password : * * * * *
Connected
select * from hr.jobs;
insert into hr.jobs (job_id, job_title)
values ('te101', '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 does not exist
- revoke all on emp from hr;
revoke succeed
- One user to another user :-
- * syntax :-
- (1) Create User username identified by password;
 - (2) grant Connect to newuser; work on it;
 - (3) grant all privilege to 'newuser';
 - (4) drop user username cascade;
- Create user mno identified by mnaz;
Error : insufficient privileges
- Conn system
Enter password : ****
Connected
- Create user mno identified by mnaz;
User created
- Select * from all users;
- Conn mno
Enter password
Error :- User mno lacks create session privilege logo
defined. You are no longer connected to

- Conn system
Enter password : ****
Connected
- grant Connect to mno; all to root@localhost
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 virtual-table.
- view is the resultant set of stored Query.
- Take advantages of the table & benefit from it.
- To restrict the data access [security].
- To make Complex Query easier.
- To provide data dependency.
- To present different views of same data.
- 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_name where column_name = value~~
- Index :-
- Create index indexname On TableName (columnname);
- drop index indexname;
- Definition :-
- 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) clustered index
- (2) non-clustered index
- clustered index :-
- in a clustered index we can Create One index in a table
- clustered index faster than the non-clustered index
- in clustered index we will store data in the "Sorted Order".
- non-clustered index :-
- in a non-clustered index we can't create One or more index in a table.
- non clustered index is slower than the clustered index.

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

→ select *
from emp
where deptNo = 10;

- Set "deletes rows" part before of rows
- (1) Unionall : It will fetch unique value
 - (2) Unionall : It will fetch the data all duplicate values also.
 - (3) Intersect : It will fetch the common column
 - (4) minus : It will not fetch the common data.

→ Select deptNo from emp
Union all select deptNo from dept;

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

→ select deptNo from emp
Intersect select deptNo from dept;

→ Select by condition, deptNo
minus
select deptNo from emp;

40
60
deptNo
group by

Set definition
for query & for example

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 multisrow 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;

→ select job, count (*)

from emp

group by job;

05 0288

06 0002

01 0002

02 0002

03 0002

04 0002

07 0002

08 0002

09 0002

10 0002

11 0002

12 0002

13 0002

14 0002

15 0002

16 0002

17 0002

18 0002

19 0002

20 0002

21 0002

22 0002

23 0002

24 0002

25 0002

26 0002

27 0002

28 0002

29 0002

30 0002

31 0002

32 0002

33 0002

34 0002

35 0002

36 0002

37 0002

38 0002

39 0002

40 0002

41 0002

42 0002

43 0002

44 0002

45 0002

46 0002

47 0002

48 0002

49 0002

50 0002

51 0002

52 0002

53 0002

54 0002

55 0002

56 0002

57 0002

58 0002

59 0002

60 0002

61 0002

62 0002

63 0002

64 0002

65 0002

66 0002

67 0002

68 0002

69 0002

70 0002

71 0002

72 0002

73 0002

74 0002

75 0002

76 0002

77 0002

78 0002

79 0002

80 0002

81 0002

82 0002

83 0002

84 0002

85 0002

86 0002

87 0002

88 0002

89 0002

90 0002

91 0002

92 0002

93 0002

94 0002

95 0002

96 0002

97 0002

98 0002

99 0002

100 0002

101 0002

102 0002

103 0002

104 0002

105 0002

106 0002

107 0002

108 0002

109 0002

110 0002

111 0002

112 0002

113 0002

114 0002

115 0002

116 0002

117 0002

118 0002

119 0002

120 0002

121 0002

122 0002

123 0002

124 0002

125 0002

126 0002

127 0002

128 0002

129 0002

130 0002

131 0002

132 0002

133 0002

134 0002

135 0002

136 0002

137 0002

138 0002

139 0002

140 0002

141 0002

142 0002

143 0002

144 0002

145 0002

146 0002

147 0002

148 0002

149 0002

150 0002

151 0002

152 0002

153 0002

154 0002

155 0002

156 0002

157 0002

158 0002

159 0002

160 0002

161 0002

162 0002

163 0002

164 0002

165 0002

166 0002

167 0002

168 0002

169 0002

170 0002

171 0002

172 0002

173 0002

174 0002

175 0002

176 0002

177 0002

178 0002

179 0002

180 0002

181 0002

182 0002

183 0002

184 0002

185 0002

186 0002

187 0002

188 0002

189 0002

190 0002

191 0002

192 0002

193 0002

194 0002

195 0002

196 0002

197 0002

198 0002

199 0002

200 0002

201 0002

202 0002

203 0002

204 0002

205 0002

206 0002

207 0002

208 0002

209 0002

210 0002

211 0002

212 0002

213 0002

214 0002

215 0002

216 0002

217 0002

218 0002

219 0002

220 0002

221 0002

222 0002

223 0002

224 0002

225 0002

226 0002

227 0002

228 0002

229 0002

230 0002

231 0002

232 0002

233 0002

234 0002

235 0002

236 0002

237 0002

238 0002

239 0002

240 0002

241 0002

242 0002

243 0002

244 0002

→ 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

5

Manager

1

6

Analyst

1

7

Represen

1

8

Analyst

1

- select deptNo, job, count(*)
from emp
group by deptNo, job
having count(*) >= 3.
- select deptNo, job, count(*)
from emp
having count(*) >= 3;
- eggs
- select deptNo, job, count(*)
from emp
where job in ('SALESMAN', 'MANAGER', 'CLERK')
group by deptNo, job
Order by job asc;
- select dept No ,job, count(*)
from emp
where job in ('SALESMAN', 'MANAGER', 'CLERK')
group by deptNo, job
having count(*) >= 2
Order by job asc;

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

Select deptno, count(*)
from emp
group by deptno
having count(*) >= 3;

COPPS