EMP & DEPT table as previous.

4. Cartesian join (cross join).
- join without a Where condition.
- select dname, ename from emp, dept; ← fast

Output:-

DNME	ENAME
TRN	Arun
TRN	Ali
TRN	Kiran
TRN	Jack
TRN	Thomas
EXP	Arun
EXP	Ali
EXP	kiran
EXP	Jack
EXP	Thomas
MKTG	Arun
MKTG	Ali
MKTG	kiran
MKTG	Jack
MKTG	Thomas
MKTG.	

- dept → driving table
- emp → driven table.
- every row of driving table is combined with each and every row of driven table.
- cross product of two tables hence it is also known as cross join.
- lesser the i/p, o/p beth the server HD & RAM, faster it will execute.

- select dname, ename from dept, emp; ← slow.
  i/p beth server ram and server HD.
  (greater the input/output beth the server HD & RAM slower it will execute)

Uses:-

1. Used for printing purposes.

e.g. ~~In University, you h~~

In students table you have all the students name, in subject table you have all the subject name. In university when you are printing the Markshees ts, every student name is combined with each and every subject name; you will required a cartesian join.

- In EMP table, EMP NO is parent column & MGR is child column.

Snehal Sawant

⑤ self join: V.Imp (interview)

- joining a table to itself.
- used when parent column and child column both are present in same table
- based on recursion:

- Select a.ename, b.ename from emp b, emp a where a.mgr = b.empno

Output:

H.D.

EMP table. (internal working)

server RAM.

A

ENAME	MGR
Arun	4
Ali	1
kiran	1
Jack	1
Thomas	4

B

ENAME	EMPNO
Arun	1
Ali	2
kiran	3
Jack	4
Thomas	5

output:-

A.ENAME	B.ENAME
Arun	Jack
Ali	Arun
kirun	Arun
~~Jack~~	Jack.
Thomas. →	

- slowest join.
- The temp. tables will discarded from server ram after execution of select statement

Snehal Sawant

- select dname, ename from emp e, dept d where d.dept no = e.deptno; (Do not use)
  - When you specify an alias for tablename, a copy of table is brought into server RAM
  - Do not specify an alias for tablename unnecessarily because not only will you select statement be slow, but you will slow down the select statements of

other users.
- specify an alias for tablename only if you are writing a self-join.

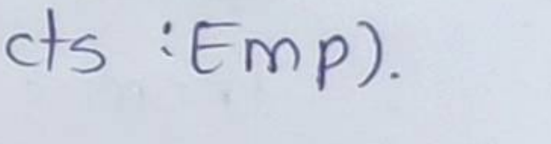
- Cartesian join is the fastest join because there is no WHERE clause, and hence no searching is involved.

for joining 3 tables.

DEPTHEAD. ((create new table).

DEPTNO	DHEAD.
1	Arun
2	Jack.

- Select dname, ename, dhead from emp. dept, depthead (5) (3) (2) rows.
  where depthead.deptno = dept.deptno
  and dept.deptno = emp.deptno;



Types of Relationships:-

one : one (Dept: Depthead) or (depthead : Dept)
One : Many (dept: emp) and (dept head : emp).
Many : one (emp : dept) and (emp : depthead)
Many : Many. (emp : projects) or (projects : Emp).

Projects.

Project NO.	Client Name	Proj-DTLS
P1	Deloitte	CGS
P2	morgan stanly.	AMS
P3	BNP Parilsos	Macrosoft
P4	ICICIbank.	PPS
P5	AMFI	website Dep.

Projects - Emp. # Intersection table.

Projno	Empno
P1	1
P1	2
P2	1
P2	3
P2	4
P3	2
P3	5
.	.
:	:

- depends on set theory.
- intersection table is required for many:many rel^n.

- select clientname, proj_dtls, ename from projects_emp, emp, projects
  where projects.projno = projects_emp.projno
  and emp.empno = projects_emp.empno
  order by 1, 2, 3;

Sub-queries: (V.V. imp).

- also known as Nested queries. (Query within Query (select with in select).

- select ename, min(sal) from emp; ← Error in oracle works in sql but meaning less output.
- select ename from emp ← Error in sql & oracle
  where sal = min(sal);

Display the ENAME who is receving sal = min(sal):-

- select ename from emp
  where sal =
  (select min(sal) from emp);

← Main query, (Parent), (Outer)
← sub-query (child), (inner)

- select ename from emp
  where sal =
  select min(sal) from emp
  where deptno = (select ----));

} ex.

Snehal Sawant

- Max upto 255 levels for sub-queries (this limit of SQL can be exceeded with the help of views).

- Join is faster than sub query.
  because when you write a join you solve the problem using 1 select statement.
  when you write sub-queries you required 2 or more select statements; the more the no. of select statements the slower it will be.

Find second largest sal:-

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

Display the rows who belong to the same dept no as Thomas:-

```
select deptno from emp.
select * from emp
where deptno =
(select deptno from emp
where ename = 'Thomas');
```

Display all the rows who are doing the same Job as 'kiran':-

```
select * from emp
where job =
(select job from emp
where ename = 'kiran');
```

Using sub-queries with DML commands:-

In oracle:-

```
delete from emp
where deptno =
(select dept no from emp
where ename = 'Thomas');
```

- update emp set sal =10000
  where job =
  (select job from emp
  where ename ='kirun');

- Aborer 2 commdnds will not work in MySQL:-
- In MySQL, you cannot update or delete from a table from which you are currently selecting:

Solution for MySQL:-

- delete from emp
  where deptno =
  (select tempp.deptno from
  (select deptno from emp
  where ename ='Thomas') as tempp);

- update emp set sal =1000
  where job =
  (select tempp.job from
  (select job from emp
  where ename ='kirun') as tempp);

Multirow sub-queries:

- sub-queries returns multiple rows
- any is special operator it will perform logical OR.

Display all the rows that are receiving a sal equal to any of the manager:

- select * from emp
  where sal = any
  (select sal from emp
  where job ='M');

- select * from emp
  where sal in
  (select sal from emp
  where job ='M');

- To exclude the manager:-

- select * from emp
  where job != 'M' ~~sal~~ and sal in
  (select sal from emp
  where job = 'M');

- select * from emp
  where sal in
  (select min(sal) from emp
  where job = 'M');

- select * from emp
  where sal >=
  (select min(sal) from emp
  where job = 'M');

To make it work faster :-

1. Join is faster than sub-query; therefore use join wherever possible.
2. Try to reduce the number of levels for sub-quer.
3. Try to reduce the number of rows returned by sub-query.

In → logical OR.

Assumption 3rd row sal is 13000 :-

- select * from emp
  where sal > all
  (select sal from emp
  where job = 'M');

- select * from emp
  where sal >
  (select max(sal) from emp
  where job = 'M');

Snehal Sawant

Assumption 3rd row sal is 3000:-

#Using sub-query in the having clause.

In Oracle:-

#Display the Dname having maximum sum(SAL)

- select deptno, sum(sal) from emp
  group by deptno;
  ~~having sum(sal)~~

@IP:-

DeptNO	sum(SAL)
1	18000
2	17000

- select sum(sal) from emp
  group by deptno;

O/P -

SUM(SAL)
18000
17000

- select max(sum(sal)) from emp
  group by deptno;

O/P -

Max(sum(SAL))
18000

- select deptno, sum(sal) from emp
  group by deptno
  having sum(sal)=
  (select max(sum(sal))from emp
  group by deptno);

O/P:-

DEPTNO	sum(SAL)
1	18000

- select dname, sum(sal) from emp, dept
  where dept.deptno = emp.deptno
  group by dname
  having sum(sal) =
  (select max(sum(sal)) from emp
  group by deptno);

O/P

DNAME	SUM(SA
TRN	1800

Snehal Sawant

In MySQL:-

Display the DNAME that is having max(sum(sal));

- select sum(sal) from emp
  groupp by deptno;

o/p

SUM(SAL)
18000
17000

- select max(sum_sal) from
  (select sum(sal) sum_sal fromemp
  group by deptno)as temppi;

o/p

Max(sum_sal)
18000

- select deptno, sum(sal) from emp
  groupby deptno
  having sum(sal) =
  (select max(sum_sal) from
  (select sum(sal) sum_sal from emp
  group by deptno) as temppi;

o/p

DeptNO	sum(SAL)
1	18000

- select dname, sum(sal) from emp, dept
  where dept.deptno = emp.deptno group by dname
  having sum(sal) = (select max(sum_sal) from
  (select sum(sal) as sum_sal from emp
  group by deptno) as temppi;

output:-

dname	sum(sal)
TRN	18000

955 - 1 to 12
exe - 1 to 6

Snehal Sawant