**Joins :-**

**=====**

--Join is used to receive data from multiple tables or by using we can combine records from multiple tables.

there r following types of joins:

1. Inner Join

2. Outer Join

Left Outer join

Right Outer Join

Full Outer Join

3. self join

4. Cross Join (Cartesian product)

Note: when we try to get the data from more than one table without using joining condition, then it is called cross join, in this case every record of the first table will be mapped with every record of the second table.

--with the cross join we don't get the meaningfull data, in order to get the meaningfull data we need to use other types of joins.

**INNER JOIN:**

============

--here we need to apply joining condition on the common data from both table

--if ambiguity is there in column name(both table having the same col name) then we need to use alias support.

--this inner join returns the matching record from the DB tables based on common column.

> select \* from dept INNER JOIN emp ON dept.did = emp.deptid;

**Q/- give the emp details who is working in 'marketing' dept.**

>select eid,ename,address,salary from dept INNER JOIN emp ON dept.did = emp.deptid AND dept.dname='marketing';

with alias support:

-----------------------

>select e.eid,e.ename,e.address,e.salary,d.dname,d.location from dept d INNER JOIN emp e ON d.did = e.deptid AND d.dname='marketing';

another syntax of INNER JOIN (without using INNER JOIN command).

----------------------------------------------------------------------------------------

>select e.eid,e.ename,e.address,e.salary,d.dname,d.location from dept d,emp e where d.did = e.deptid AND d.dname='marketing';

Left Outer Join:

--------------------

--to get the unmatched records from the left table use left outer join (it shows the details of left table and null value for the right table).

Right Outer Join:

--------------------

--to get the unmatched records from the right table use right outer join (it shows the details of right table and null value for the left table).

full outer join:

-----------------

--it display the null values both side for all the unmatched records.

Note: Full Outer join is not supported by the mysql DB.

--in order to use full outer join in mysql, then we should use union of left join and right join.

select d.did,d.dname,e.ename,e.address from dept d LEFT JOIN emp e ON d.did = e.deptid UNION

select d.did,d.dname,e.ename,e.address from dept d RIGHT JOIN emp e ON d.did = e.deptid;

Self Join:

========

--here we use joining a table to itself.

--here joining condition col must belongs to same datatype.

Note:- if we want to compare two table same col value then we use INNER JOIN whereas if we want to compare 2 diff col values within a single table then we must use self join.

\*\*\*whenever a table contains hirarical data then only we allow to use self join.

ex:

emp ----> manager

student ---> monitor

when we use self join, we must take the support of alias.

create table emp5

(

eid int primary key,

ename varchar(12),

salary int,

mgr int

);

mysql> insert into emp5 values(100,'Ram',7800,null); // RAM does not have any manager

mysql> insert into emp5 values(110,'Ravi',7200,100); // Ravi manager is RAM

mysql> insert into emp5 values(112,'amit',7500,100); // amit manager is RAM

mysql> insert into emp5 values(114,'sunil',7000,110); // sunil manager is RAVI

Q/- display the emp name and their manager name.

--here we need to use SELF JOIN

> select e1.ename EMPLOYEE, e2.ename MANAGER from emp5 e1,emp5 e2 where e1.mgr = e2.eid;

subqueries:

===========

--a query inside another query is called subquery or nested query.

--sub queries r used to retrieve the data from single or multiple tables based on more than one step process.

--here outer query is called parent query and inner query is called child query.

--child query will execute first then only parent query will be executed.

Child query :- it provides values/data to the parent query.

Parent query : it recieves the values/data from the child query.

--in child query we can not use order by clause, but parent query can use.

--group by clause can be used in both queries.

subqueries we can categories into following categories:

------------------------------------------------------------------

1.single row and single col SQ(scalar value SQ)

2.multiple row single col SQ

3.multiple col SQ

1.single row and single col SQ(scalar value SQ):

---------------------------------------------------------

--here child query will return only a single value.

--here mostly same col name which is in the where clause of the parent query , will be there inside the select clause of child query.

**Q/- WAQ to display emp details who is working in marketing dept.**

using JOIN:-

>select eid,ename,address from dept INNER JOIN emp ON dept.did = emp.deptid AND dname='marketing';

using SQ:-

>select eid,ename,address from emp where deptid = (select did from dept where dname = 'marketing');

**Q/- WAQ to display emp details who is working with chandan;**

> select \* from emp where deptid = (select deptid from emp where ename='chandan');

**Q/- WAQ to display emp details who are getting more sal than avg sal from emp table.**

> select \* from emp where salary > (select avg(salary) from emp);

**Q/ WAQ to display second higest salary employee details.**

first highest salary emp:

> select \* from emp where salary = (select max(salary) from emp);

second highest salary emp:-

>select \* from emp where salary = (select max(salary) from emp where salary < (select max(salary) from emp));

**Q/- WAQ to display details of emp who is working under RAVI;**

select \* from emp5 where mgr = (select eid from emp5 where ename='ravi');

2.Multiple row,single col SQ:

=======================

--in multi-row, single col SQ, child query will return multiple rows and single col to the parent query.

--in this case in parent query we should use one of following operators:

IN

ANY

ALL

ex:

>select \* from emp where salary IN (select salary from emp where eid > 100);

IN : it check equal to any number in the list (using OR)

ANY : it compare any value in the list

All : it compares all values in the list

ex: --

salary > any(----); here it checks salary should be greater than any of 4 values in the list

salary > all(----); here it checks salary should be greater than all of 4 values in the list

>any(10,20,30,40)

ex:

< any(): less than any :- less than maximum

> any(): greater than any : - greater than equal to minimum

= any(): equal to any :- it is equal to IN operator.

5 < all(10,20,30,40)

< all(): less than all :- less than minimum

> all(): greater than all : - greater than maximum

= all(): equal to all :- it is meaningless (becoz one value can not be equal to 3 or 4 value).

ex:

>select \* from emp where salary = ANY(select salary from emp where eid > 100);

Q/- WAQ to display the emp who is getting max salary in each dept ?

>select \* from emp where salary IN(select max(salary) from emp group by deptid);

3.multicolumn subquery:

---------------------------------

--if we try to compare multiple col values of the child query with the multiple col values of the parent query then we use this type of SQ.

syn:

select \* from tab\_name where (col1,col2,....) IN (select col1, col2,.... from table where condition)

Q/- WAQ to display the emp whose salary and did matches with the salary and did of a RAM.

>select \* from emp where (salary,deptid) IN (select salary,deptid from emp where ename='RAM');

SQ in DML:

==========

SQ in insert :

---------------

>create table x1(id int, name varchar(12));

> insert into x1 (select eid,ename from emp);

>insert into x1 values(500, (select ename from emp where eid=100));

SQ in update:

---------------

here SQ is allowed inside where clause or Set clause.

ex:

> update x1 set name= 'ramesh' where id=(select eid from emp where ename='ram');

> update x1 set name= (select ename from emp where eid=100) where id=(select eid from emp where ename='ram');

SQ in delete:

> delete from x1 where id = (select eid from emp where ename='ram');