To alter the table

To add new column at the beginning of the table. We may add not null constraint while adding column only if the table is empty, Otherwise add column, update the values, and then use modify column to add not null constraint	Alter table mytable Add lastname varchar(30) not null first Alter table mytable Add mname varchar(30) after fname
To delete the column	Alter table mytable Drop column lname
To modify the column definition	Alter table mytable Modify mname varchar(50) not null
To change the name of the column	Alter table mytable Change fname firstname varchar(30)
To add constraint in the table	Alter table mytable Add primary key(myid) Alter table mytable Add constraint f_fname foreign key(fname) references mytab2(firstname)
To drop the constraint	Alter table mytable Drop primary key Alter table mytable Drop foreign key f_fname
To rename the table	alter table mytable rename to mytab;
To find name of constraint	Show create table myproduct
To find name of constraint	select COLUMN_NAME, CONSTRAINT_NAME, REFERENCED_COLUMN_NAME, REFERENCED_TABLE_NAME from information_schema.KEY_COLUMN_USAGE where TABLE_NAME = 'student_marks and and table_schema='acts0923'

To drop the table

-----to delete all the rows and the table structure also

drop table <table name>

To truncate table

-----to delete all the rows from the table, but it will keep empty table

truncate table mytable;

DML statements

All DML statements works only on one table at a time

1. Insert
2. Update

   ```
   Update <table name>
   Set col1=val1,col2=val2,col3=val3
   Where <condition>

   Update emp
   Set sal=sal*1.10,deptno=20
   Where sal>2000;
   ```

3. Delete

------to delete all the rows from the table but empty table will remain

   Delete from <table name>   ---→ in both mysql and oracle

   Delete <tablename>   -→ only in oracle

------to delete rows based on condition

   ```
   Delete from <table name>
   Where <condition>

   Delete from mytable
   Where lname='joshi';
   ```

Truncate	delete
It is DDL statement	It is DML statement
Since all DDL statements are auto commit, Rollback is not possible	But rollback is possible if auto commit is off
Where clause cannot be used in truncate	Where clause can be used

Tcl--→ transaction control statements
Commit-→ make the changes permanent.
Rollback--→ undo the changes in the table
Savepoint-→ adding markers in between multiple dml statements.

To commit the changes either use commit, or if you execute any DDL statement, it will autocommit the changes

To set auto commit off
Set autocommit=0;

To set auto commit off
Set autocommit=1;

If you have a table mytab, it contains 10 rows.
Insert ---1
Insert----1
Insert ----1
---execute any DDL
Insert----2
Savepoint A
Update---2
Savepoint B
Delete ----10
Rollback to B

Nested Query
If we write one query inside another query then it is called as nested query
Nested queries can be used in where or from clause also
It can be used in create table and all DML statement also

1. Simple nested query
    a. If inner query(child query) is independent query, then it is called as simple nested query
    b. Inner query will get executed only once.
2. co. related nested query.
    a. If inner query (child query) is dependent on parent query, then it is called as co-related query
    b. Inner query will get executed once for each row in the parent table
3.

--------to display all employees with sal>2000

Select * from emp where sal>2000;

--------to display all employees with sal>miller's salary

Select *

From emp

Where sal>(Select sal from emp where ename='Miller');

---to find all employees who works under same mgr as Smith.

```
Select *

From emp

Where mgr=(select mgr from emp where ename=’SMITH’)

-------to find all employees who works in same department as miller or Jones work

Select *

From emp

Where deptno in  (select deptno from emp where ename in (‘miller’,’jones’));


------to find all employees whose salary is > both miller and Blake salary

Select *

From emp

Where sal > all (select sal from emp where ename in (‘SMITH’,’BLAKE’))


------to find all employees whose salary is > any one , miller or Blake salary

Select *

From emp

Where sal > any (select sal from emp where ename in (‘SMITH’,’BLAKE’))


------find all employees whose sal >= smith’s sal and < blakes salary

Select *

From emp

Where sal between (select sal from emp where ename=’SMITH’) and (select sal from emp where
ename=’BALKE’)


----find all employees who works in accounting department.

Select * from emp

Where deptno=(select deptno from dept where dname=’ACCOUNTING’);

-----find all employees with salary > avg salary of dept ‘Sales’

Select * from emp

Where sal >(select avg(sal) from emp where deptno=(select deptno from dept where
dname=’SALES’))
```

```
-------to create a table myemp, add all the rows with sal> 2000 from emp table

Create table myemp

As

(select * from emp where sal>2000)

---to add only limited columns

Create table myemp

As

(select empno,ename,sal from emp where sal>2000)

-------to create a empty table myemp with all the columns same as emp

Create table myemp

As

(select * from emp where 1=2)


------update smiths salary = millers salary

Update emp

Set sal=(select sal from (select * from emp) e where e.ename=’MILLER’)

Where ename=’SMITH’


----update sal of all employees who works in Accounting dept  to ‘blake’ sal

Update emp

Set sal=(select sal from (select * from emp) e where e.ename=’blake’)

Where deptno=(select deptno from dept where dname=’accounting’)


---delete all employees who are working in smiths department

Delete from emp

Where deptno=(select deptno from (select * from emp) e where ename=’SMITH’)

-----find all employees with salary > avg salary of their own department

select * from emp e

where sal>(select avg(sal) from emp m where m.deptno=e.deptno);


-----find all employees with salary < avg salary of all employees working under same manager.
```

select * from emp e

where sal<(select avg(sal) from emp m where m.mgr=e.mgr);

exists	Returns true if inner query returns 1 or more rows
Not exists	Returns true if inner query returns 0 rows

------to display all departments in which no employees are there

select * from dept d

-> where not exists(select * from emp e where e.deptno=d.deptno);

------to display all employees who are not working as mgr for any one

select * from emp e

-> where not exists(select * from emp m where m.mgr=e.empno);

Joins

When the o/p needs columns from more than one table then use joins

Types of joins

1. Cross join
2. Inner join-→ cross join with join condition
   a. Equi join-→ if join condition uses = sign then it is called as equi join
   b. Non equi join---→ if join condition does not uses = sign then it is called as non equi join
   c. Self join--→ if the table joins itself then it is self join
3. Outer join
   a. Right join
   b. Left join
   c. Full outer join

--------to display all employees and their department names

select empno,ename,e.deptno,d.deptno,dname
 	
from emp e , dept d
 	
where e.deptno=d.deptno;
	select empno,ename,e.deptno,d.deptno,dname
	
	from emp e inner join dept d
	
	on e.deptno=d.deptno;

--------to display all employees and their department names for all employees with sal> 2000

select empno,ename,sal,e.deptno,d.deptno,dname from emp e , dept d where e.deptno=d.deptno and sal>2000;	select empno,ename,e.deptno,d.deptno,dname from emp e inner join dept d on e.deptno=d.deptno;

--------display empno,name,sal, and grade of the employee

select empno,ename,sal,grade,losal,hisal from emp e,salgrade s where e.sal between s.losal and s.hisal;	select empno,ename,sal,grade,losal,hisal from emp e inner join salgrade s on e.sal between s.losal and s.hisal;

------display all employeedetails with manager name

select e.empno,e.ename,e.mgr,m.empno,m.ename from emp e, emp m where e.mgr=m.empno;	select e.empno,e.ename,e.mgr,m.empno,m.ename from emp e inner join emp m on e.mgr=m.empno;