-- create database employee;

-- use employee;

-- create table tab ( ID int not null, Name varchar (30), Salary int );

-- alter table tab add ( age int );

-- alter table tab drop column age;

-- alter table tab modify Name varchar(50);

-- alter table tab modify name int;

-- alter table tab rename column ID to Id;

-- alter table tab rename tab1;

-- -- changing name without alter command check it below

-- rename table tab1 to tab;

-- drop table tab;

-- drop database employee;

-- -- data ya database ko define karte he

-- -- alter command is used to define structure

-- Data types define the kind of data types

-- a)numeric data type ->

-- 1) int -> used for integer

-- 2) float -> used for decimal

-- a)) precision -> kitne digit ayenge (total including \_\_.\_\_\_)

-- b)) scale -> decimal ke bad string (6-9)

-- 3) double ->excedding (9)

-- 4) numeric -> exact value milege toh use karenge --> exact decimal number

-- b) character or string data type

-- 1) char

-- 2) varchar

-- c) date and time data type

-- 1) date-> yyyy-mm-dd

-- 2) time-> hour:min:sec

-- kisi bhe database me data ko kaise insert karte he -> insert into

-- create database dbs;

-- use dbs;

-- create table student(roll\_no int not null unique , name varchar (30) );

-- insert into student value('1', 'ankit');

-- insert into student value( '2', 'bose');

-- or

-- sql constraints

-- 1) not null

-- 2) unique -> duplicate value nahi ayege -- data redundancy reduce ho jayege

-- 3) primary key -> iske jo bhe conditions hongi wo attribute par apply jo jayege

-- 4) foreign key -> connect two tables

-- 5) default -> kisi ke ander default value store karwani hoti he toh use hota he

-- 6) check -> to check conditions before adding data

-- create database dbs;

-- use dbs;

-- create table fg1 (roll\_no int, name varchar(30), primary key ( roll\_no));

-- create table course1 ( course\_id int primary key, course\_name varchar (30));

-- drop table student;

-- create table student ( roll\_no int primary key, name varchar(30), course\_id int , foreign key ( course\_id) references course1( course\_id));

-- how to connect two tables

-- create table gg ( roll\_no int , name varchar(30), course\_name varchar(30) default 'dbms');

-- insert into gg( roll\_no, name) values ('1', 'ankit');

-- insert into gg values ('1', 'ankit', 'oops');

-- -- default value ayege jab tab hm specify nahi karenge

-- create table gg1 ( roll\_no int ,age int check(age>=18));

-- name varchar(30), course\_name varchar(30) default 'dbms');

-- insert into gg1 values ('1', '19');

-- insert into gg1 values ('2', '20');

-- insert into gg1 values ('2', '8');

-- sql -> 1)ddl command, 2)dql command->data query language --> used for quering and retreiving the data from the table a) select-> used for retreiving the data

-- b) order by-> sort the data ----> ascending order (asc) and for (desc) descending order

-- c) distinct --> attribute me se duplicate remove kar dega

select \* from gg1;

-- -- will retrieve complete data what is have entered

-- select age from gg1;

-- used to show particular attribute data or column

-- need to column data

-- select age, roll\_no from gg1;

-- for particular row

-- select \* from gg1 where roll\_no='2';

-- --

-- select age from gg1 where roll\_no='2';

-- for age it will retrievie roll-no

--

-- select \* from gg1 order by age asc;

--

--

create database practice;

use practice;

create table tab ( ID int not null, Name varchar (30), Salary int );

insert into tab values('1', 'a', '100');

insert into tab values('5', 'b', '10000');

insert into tab values('10', 'c', '1001');

insert into tab values('9', 'd', '1001');

insert into tab values('9', 'daniel', '1001');

insert into tab values('889', 'aman', '1001');

insert into tab values('92', 'amu', '1001');

insert into tab values('93', 'ammen', '1001');

insert into tab values('95', 'dinish', '1001');

insert into tab values('915', 'annbcdefg', '1001');

insert into tab values('9522', 'abcecdacb', '1001');

select \* from tab order by id desc, salary asc;

-- pehle id ko sort karega then salary if their are same id

select distinct salary from tab;

-- for distinct data

-- -- select distinct \* from tab;

-- between command -> range ke bich me data find retrieve karenga

select \* from tab where Name between 'a' and 'd';

-- in -> set of values de di jae agar same value find karne ho to uus data ko retrieve kar dega

select \* from tab where Salary in ('100','10000');

-- like -> used for searching specific patten

select \* from tab where Name like 'b%';

-- jinka first letter a se start hoga

select \* from tab;

-- \_a% -> for first letter unknown, %a\_ -> for last letter ,% for unknown lenght, \_ for known letter

--

select \* from tab where Name like '\_m%';

-- %i% for kisi bhe position par aasakta word me

select \* from tab where Name like '%i%';

select \* from tab where Name like '\_m\_n%';

select \* from tab where Name like '\_man%';

select \* from tab where Name like '%a%b%c%';

-- try %ab%c%

-- \_\_\_\_a \_\_\_\_\_b\_\_\_\_c\_\_\_ right

-- \_\_\_b \_\_\_\_c\_\_\_\_a\_\_\_\_\_\_\_\_\_\_ wrong order will be matched

-- aggregate functions ->is used to perform calculations on data table

select count(\*) from tab;

select \* from tab;

-- a) count() -> kitne rows he, without null values kitna data he. return the number the rows

select count() from tab;

-- without null values show karenga data in specified column

select count(distinct salary) from tab;

select count(distinct salary) from tab where salary>'100';

-- b) sum() -> age ka sum, marks ka sum calculate karna ho

select avg(distinct salary) from tab where salary>'100';

-- c) avg()-> sum by count

-- d)min()-> find min data

-- e)max()-> find max data

select max(salary) from tab;

select min(salary) from tab;

select \* from employees\_100;

select salary - ( select avg(salary) from tab) from tab;

select max(salary) from tab where salary<(select max(salary) from tab) ;

select age , count(\*), avg(salary) from employees\_100 where age>30 group by age order by age ;

select age , count(\*), avg(salary) from employees\_100 where age>30 group by age having count(age)>2 order by age;

select age , count(\*), avg(salary) , sum(salary) as bonus from employees\_100 group by age having count(age)>1 order by count(\*);

select age , count(\*), avg(salary) , sum(salary) as bonus from employees\_100 group by age having avg(salary)>10000 and age > 40 order by count(\*);

select age , count(\*), avg(salary) , sum(salary) as bonus from employees\_100 group by age having avg(salary)>10000 and count(age)> 3 order by count(\*) limit 5;

select salary from employees\_100 order by salary desc limit 5 offset 2;

select \* from employees\_100 order by salary desc limit 5;

set sql\_safe\_updates=0;

update employees\_100 set salary=1000000000 , age= 40 where id=1;

delete from employees\_100 where id=2;

use dbs;

select roll\_no from gg union all select roll\_no from gg1;

select roll\_no from gg INTERSECT select roll\_no from gg1;

create database parrot;

use parrot;

create table student ( Student\_ID int not null, Name varchar (30), Course\_Id int );

insert into student value('1', 'john', '101');

insert into student value('2', 'alice', '102');

insert into student value('3', 'bob', '103');

insert into student value('4', 'carol', '101');

create table course ( Course\_Id int , Course\_Name varchar (30) );

alter table course rename column Course\_ID to Course\_Id;

insert into course value( '101', 'cse');

insert into course value( '102','physics');

insert into course value( '104','electronic');

select \* from student inner join course on student.Course\_Id = course.Course\_Id order by Student\_Id;

select name, Course\_Name from student inner join course on student.Course\_Id = course.Course\_Id order by Student\_Id;

select \* from student inner join course on student.Course\_Id = course.Course\_Id order by Student\_Id;

select \* from student left join course on student.Course\_Id = course.Course\_Id order by Student\_Id;

select Name, Course\_Name from student right join course on student.Course\_Id = course.Course\_Id order by Student\_Id;

-- select Name, Course\_Name from student full join course on student.Course\_Id = course.Course\_Id order by Student\_Id;

select \* from student left join course on student.Course\_Id = course.Course\_Id union select \* from student right join course on student.Course\_Id = course.Course\_Id;