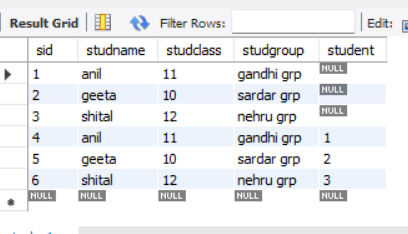
1) Create a table teachers with the following columns: teacher\_id (Primary Key), teacher\_name (NOT NULL), subject (NOT NULL), and email (UNIQUE). • Lab 2: Implement a FOREIGN KEY constraint to relate the teacher\_id from the teachers table with the students table



create database admins;

use admins;

create table teacherss(teacher\_id integer primary key auto\_increment,

teachername varchar(20) not null,

subject varchar(20) not null,

email varchar(20) unique);

create table studs(sid integer primary key auto\_increment,

studname varchar(15),

studclass integer,

studgroup varchar(20),

student integer,

foreign key(student) references teachers(teacher\_id));

insert into teacherss(teachername,subject,email)values

('Vishwamam','science','abc@gmail.com'),

('dixeemam','economics','abcds@gmail.com'),

('haritsir','social','sdfg@gmail.com'),

('Vishalsir','maths','abcd@gmail.com'),

('Vishakhamam','biology','xyz@gmail.com');

insert into studs(studname,studclass,studgroup,student)values

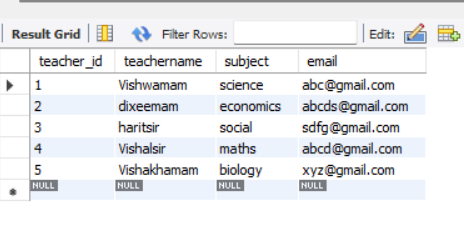
('anil',11,'gandhi grp',1),

('geeta',10,'sardar grp',2),

('shital',12,'nehru grp',3);

select \* from teacherss;

select \* from studs;



2) Lab 1: Write SQL queries to retrieve specific columns (student\_name and age) from the students table. • Lab 2: Write SQL queries to retrieve all students whose age is greater than 10

create database studentinfo;

use studentinfo;

create table student(sid integer primary key auto\_increment,

stud\_name varchar(20),

age integer,

class integer,

check(age>18));

select \* from students;

insert into student(stud\_name,age,class)values

('vishwa',15,12),

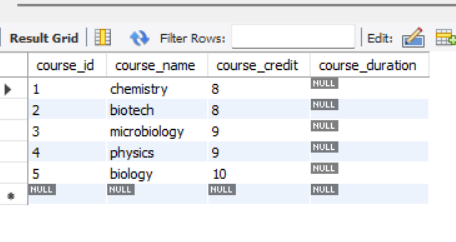
('dixee',19,12),

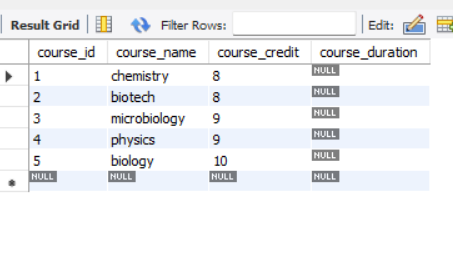
('harit',20,12),

('vishal',19,12),

('vishwas',15,10);

3) Lab 1: Modify the courses table by adding a column course\_duration using the ALTER command. • Lab 2: Drop the course\_credits column from the courses table.





create database university\_db;

use university\_db;

create table courses(course\_id integer primary key auto\_increment,

course\_name varchar(20),

course\_credit integer);

insert into courses(course\_name,course\_credit)values

('chemistry',8),

('biotech',8),

('microbiology',9),

('physics',9),

('biology',10);

select \* from courses;

alter table courses add column course\_duration integer;

select \* from courses;

alter table courses drop column course\_credit;

select \* from courses;

4) Drop the teachers table from the school\_db database. • Lab 2: Drop the students table from the school\_db database and verify that the table has been removed.

Ans\_ create database studentinfo;

use studentinfo;

create table student(sid integer primary key auto\_increment,

stud\_name varchar(20),

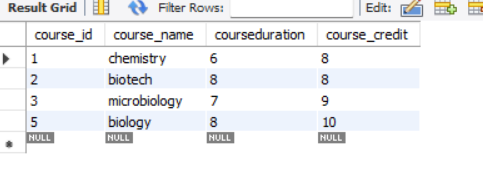
age integer,

class integer,

drop table teacher;

select \* from teacher;

5) Insert three records into the courses table using the INSERT command. • Lab 2: Update the course duration of a specific course using the UPDATE command. • Lab 3: Delete a course with a specific course\_id from the courses table using the DELETE command



create database university\_dbb;

use university\_dbb;

create table courses(course\_id integer primary key auto\_increment,

course\_name varchar(20),

courseduration integer,

course\_credit integer);

insert into courses(course\_name,courseduration,course\_credit)values

('chemistry',6,8),

('biotech',7,8),

('microbiology',7,9),

('physics',8,9),

('biology',8,10);

update courses set courseduration=8 where course\_id=2;

select \* from courses;

delete from courses where course\_id=4;

select \* from courses;

6) Lab 1: Retrieve all courses from the courses table using the SELECT statement. • Lab 2: Sort the courses based on course\_duration in descending order using ORDER BY. • Lab 3: Limit the results of the SELECT query to show only the top two courses using LIMIT.

create database university\_dbb;

use university\_dbb;

create table courses(course\_id integer primary key auto\_increment,

course\_name varchar(20),

courseduration integer,

course\_credit integer);

insert into courses(course\_name,courseduration,course\_credit)values

('chemistry',6,8),

('biotech',7,8),

('microbiology',7,9),

('physics',8,9),

('biology',8,10);

update courses set courseduration=8 where course\_id=2;

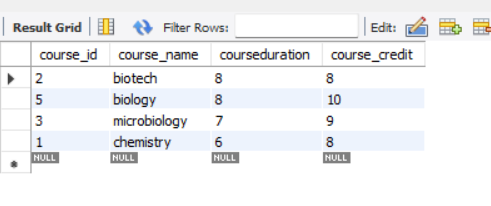
select \* from courses;

delete from courses where course\_id=4;

select \* from courses;

select \* from courses;

select \* from courses order by courseduration desc;



create database university\_dbb;

use university\_dbb;

create table courses(course\_id integer primary key auto\_increment,

course\_name varchar(20),

courseduration integer,

course\_credit integer);

insert into courses(course\_name,courseduration,course\_credit)values

('chemistry',6,8),

('biotech',7,8),

('microbiology',7,9),

('physics',8,9),

('biology',8,10);

update courses set courseduration=8 where course\_id=2;

select \* from courses;

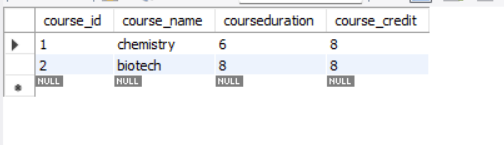
delete from courses where course\_id=4;

select \* from courses;

select \* from courses;

select \* from courses order by courseduration desc;

select \* from courses where course\_id limit 2;



7) Lab 1: Create two new users user1 and user2 and grant user1 permission to SELECT from the courses table. • Lab 2: Revoke the INSERT permission from user1 and give it to user2.

Need to revise again

8) Lab 1: Insert a few rows into the courses table and use COMMIT to save the changes. • Lab 2: Insert additional rows, then use ROLLBACK to undo the last insert operation. • Lab 3: Create a SAVEPOINT before updating the courses table, and use it to roll back specific changes.

create database vv;

use vv;

create table courses(course\_id integer primary key auto\_increment,

course\_name varchar(20),

courseduration integer,

course\_credit integer);

insert into courses(course\_name,courseduration,course\_credit)values

('chemistry',6,8),

('biotech',7,8),

('microbiology',7,9),

('physics',8,9),

('biology',8,10);

select \* from courses;

insert into courses(course\_name,courseduration,course\_credit) values

('sanskrit',8,9),

('drawing',5,8);

commit;

select \*from courses;

insert into courses(course\_name,courseduration,course\_credit) values

('civil',8,9),

('archiotect',5,8);

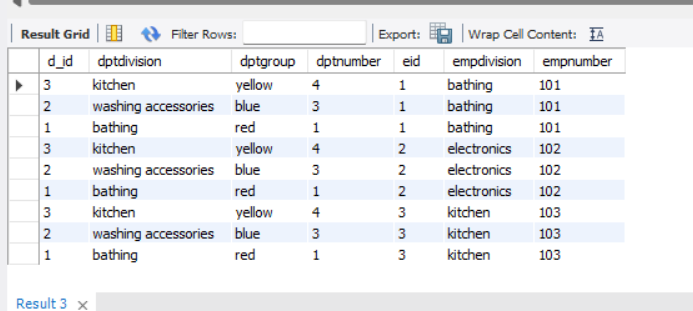
commit;

select \* from courses;

rollback;

select \* from courses;

Lab 1: Create two tables: departments and employees. Perform an INNER JOIN to display employees along with their respective departments. • Lab 2: Use a LEFT JOIN to show all departments, even those without employees.

create database shop;

use shop;

create table departments(d\_id integer primary key auto\_increment,

dptdivision varchar(20),

dptgroup varchar(20),

dptnumber integer);

create table employees(eid integer primary key auto\_increment,

empdivision varchar(20),

empnumber integer);

insert into departments(dptdivision,dptgroup,dptnumber) values

('bathing','red',1),

('washing accessories','blue',3),

('kitchen','yellow',4);

insert into employees(empdivision,empnumber)values

('bathing',101),

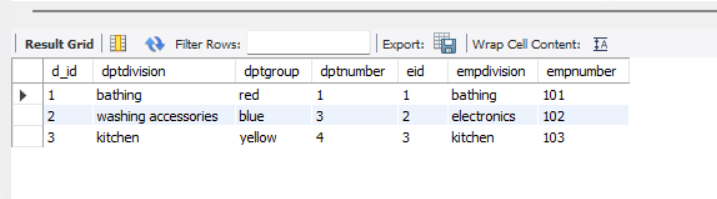
('electronics',102),

('kitchen',103);

select \* from departments;

select \* from employees;

select \* from departments join employees;



select \* from departments left join employees on d\_id=eid;

Lab 1: Group employees by department and count the number of employees in each department using GROUP BY. • Lab 2: Use the AVG aggregate function to find the average salary of employees in each department.

create database shop;

use shop;

create table departments(d\_id integer primary key auto\_increment,

dptdivision varchar(20),

dptgroup varchar(20),

dptnumber integer);

create table employees(eid integer primary key auto\_increment,

empdivision varchar(20),

empnumber integer);

insert into departments(dptdivision,dptgroup,dptnumber) values

('bathing','red',1),

('washing accessories','blue',3),

('kitchen','yellow',4);

insert into employees(empdivision,empnumber)values

('bathing',101),

('electronics',102),

('kitchen',103);

select \* from departments;

select \* from employees;

select \* from departments join employees;

select count(eid)empnumber from employees group by empnumber;