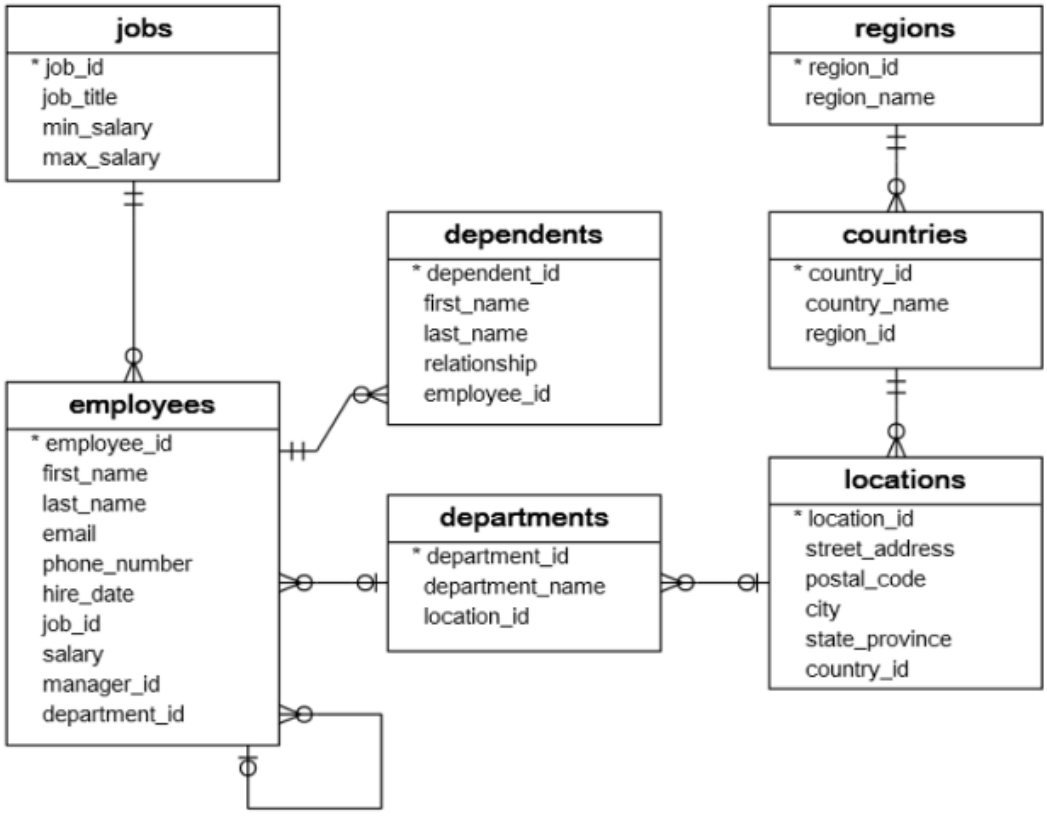
COURSE OUTCOME 1

Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

PROGRAM 1

AIM: Create a database company, and tables Employees, Departments, Dependents, Locations, Countries, and Regions and perform the queries using DDL commands with integrity constraints.

ER Diagram:



1. Create a new database named ‘company’.

```
MariaDB [(none)]> create database company;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> use company;
Database changed
```

2. Create tables with necessary primary keys and foreign keys.

```
MariaDB [company]> create table jobs (job_id int primary key, job_title varchar(50), min_salary int, max_salary int);
Query OK, 0 rows affected (0.018 sec)

MariaDB [company]> create table regions (region_id int primary key, region_name varchar (50));
Query OK, 0 rows affected (0.013 sec)

MariaDB [company]> create table countries (country_id int primary key, country_name varchar (100), region_id int, foreign key(region_
id) references regions(region_id));
Query OK, 0 rows affected (0.029 sec)
```

```
MariaDB [company]> create table locations (location_id int primary key, street_address varchar(100), postal_code varchar(50), city va
rchar(50), state_province varchar(50), country_id int, foreign key(country_id) references countries(country_id));
Query OK, 0 rows affected (0.036 sec)

MariaDB [company]> create table departments (department_id int primary key, department_name varchar(100), location_id int, foreign ke
y(location_id) references locations(location_id));
Query OK, 0 rows affected (0.032 sec)
```

```
MariaDB [company]> create table dependents (dependent_id int primary key, first_name varchar(50), last_name varchar(50), relationship
 varchar (100), employee_id int, foreign key(employee_id) references employees (employee_id));
Query OK, 0 rows affected (0.032 sec)
```

```
MariaDB [company]> create table employees (employee_id int primary key, first_name varchar(50), last_name varchar(50), email varchar(
100), phone_number varchar(15), hire_date date, job_id int, foreign key(job_id) references jobs(job_id), salary int(10), manager_id i
nt, department_id int, foreign key(department_id) references departments(department_id));
Query OK, 0 rows affected (0.030 sec)
```

3. Insert values into each table

```
MariaDB [company]> INSERT INTO `jobs`(`job_id`, `job_title`, `min_salary`, `max_salary`) VALUES ('1','HR','20000','40000');
Query OK, 1 row affected (0.003 sec)

MariaDB [company]> INSERT INTO `jobs`(`job_id`, `job_title`, `min_salary`, `max_salary`) VALUES ('2','manager','30000','50000');
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [company]> INSERT INTO `regions`(`region_id`, `region_name`) VALUES ('111','Asia');
Query OK, 1 row affected (0.004 sec)

MariaDB [company]> INSERT INTO `regions`(`region_id`, `region_name`) VALUES ('112','Europe');
Query OK, 1 row affected (0.003 sec)

MariaDB [company]> INSERT INTO `regions`(`region_id`, `region_name`) VALUES ('113','America');
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [company]> INSERT INTO `countries`(`country_id`, `country_name`, `region_id`) VALUES ('100','India','111');
Query OK, 1 row affected (0.004 sec)

MariaDB [company]> INSERT INTO `countries`(`country_id`, `country_name`, `region_id`) VALUES ('101','Germany','112');
Query OK, 1 row affected (0.006 sec)

MariaDB [company]> INSERT INTO `countries`(`country_id`, `country_name`, `region_id`) VALUES ('102','Chicago','113');
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [company]> INSERT INTO `locations`(`location_id`, `street_address`, `postal_code`, `city`, `state_province`, `country_id`) VA
LUES ('1000','70, High School Road, Sangli Factory Area','400053','Mumbai','Maharashtra','100');
Query OK, 1 row affected (0.004 sec)

MariaDB [company]> INSERT INTO `locations`(`location_id`, `street_address`, `postal_code`, `city`, `state_province`, `country_id`) VA
LUES ('1001','Mollstrasse 16','64289','Hamburg Ottensen','Hamburg','101');
Query OK, 1 row affected (0.004 sec)

MariaDB [company]> INSERT INTO `locations`(`location_id`, `street_address`, `postal_code`, `city`, `state_province`, `country_id`) VA
LUES ('1002','179 Poplar Avenue','92010','Chula Vista','California','102');
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [company]> INSERT INTO `departments`(`department_id`, `department_name`, `location_id`) VALUES ('2000','Sales','1002');
Query OK, 1 row affected (0.004 sec)

MariaDB [company]> INSERT INTO `departments`(`department_id`, `department_name`, `location_id`) VALUES ('2001','Marketing','1001');
Query OK, 1 row affected (0.003 sec)

MariaDB [company]> INSERT INTO `departments`(`department_id`, `department_name`, `location_id`) VALUES ('2002','Development','1000');

Query OK, 1 row affected (0.004 sec)
```

PROGRAM 2

AIM: Create an application to apply Data Manipulation Language (DML) commands to modify the database.

Use database named ‘company’.

```
MariaDB [(none)]> use company;
Database changed
```

1. Write a query to display all the countries.

```
MariaDB [company]> select country_name from countries;
+--------------+
| country_name |
+--------------+
| India        |
| Germany      |
| Chicago      |
+--------------+
3 rows in set (0.001 sec)
```

2. Write a query to display specific columns like email and phone number for all the employees.

```
MariaDB [company]> select email,phone_number from employees;
+-------------------+--------------+
| email             | phone_number |
+-------------------+--------------+
| rohan@gmail.com   | 9999999999   |
| hazel@gmail.com   | 9809097890   |
| jithin@gmail.com  | 9890768909   |
| paul@gmail.com    | 9999999999   |
| riya@gmail.com    | 9656445566   |
| issac@gmail.com   | 9777777777   |
+-------------------+--------------+
6 rows in set (0.001 sec)
```

3. Write a query to display the data of employee whose last name is “Lal”.

```
MariaDB [company]> select * from employees where last_name='Lal';
+-------------+------------+-----------+-----------------+--------------+------------+--------+--------+------------+---------------
+
| employee_id | first_name | last_name | email           | phone_number | hire_date  | job_id | salary | manager_id | department_id
|
+-------------+------------+-----------+-----------------+--------------+------------+--------+--------+------------+---------------
+
|       1113 | Jithin     | Lal       | jithin@gmail.com | 9890768909   | 2016-09-20 |      2 |   5000 |        233 |          2001
|
+-------------+------------+-----------+-----------------+--------------+------------+--------+--------+------------+---------------
+
1 row in set (0.001 sec)
```

4. Write a query to find the hire date for employees whose last name is “John” or “Sivan”.

```
MariaDB [company]> select first_name,last_name,hire_date from employees where last_name IN ('John','Sivan');
+------------+-----------+------------+
| first_name | last_name | hire_date  |
+------------+-----------+------------+
| Jay        | Sivan     | 2017-01-16 |
| Paul       | John      | 2018-03-13 |
+------------+-----------+------------+
```

5. Write a query to display name of the employees who work as clerks.

```
MariaDB [company]> select e.first_name,e.last_name,j.job_title from employees e join jobs j on e.job_id=j.job_id where j.job_title='C
lerks';
+------------+-----------+-----------+
| first_name | last_name | job_title |
+------------+-----------+-----------+
| Jay        | Sivan     | Clerks    |
+------------+-----------+-----------+
1 row in set (0.001 sec)
```

6. Write a query to get all the employees who work for department 3.

```
MariaDB [company]> select e.first_name,e.last_name,d.department_id from employees e join departments d on e.department_id=d.departmen
t_id where d.department_id='3';
+------------+-----------+---------------+
| first_name | last_name | department_id |
+------------+-----------+---------------+
| Riya       | Manuel    |             3 |
| Issac      | Abraham   |             3 |
| Satya      | Paul      |             3 |
+------------+-----------+---------------+
3 rows in set (0.001 sec)
```

7. Write a query to display the departments in the descending order.

```
MariaDB [company]> select department_id,department_name from departments order by department_id desc;
+---------------+--------------------+
| department_id | department_name    |
+---------------+--------------------+
|          2002 | Development        |
|          2001 | Marketing          |
|          2000 | Sales              |
|            10 | Ordering           |
|             8 | Shipping           |
|             6 | Automatic Testing  |
|             3 | Manual Testing     |
+---------------+--------------------+
7 rows in set (0.001 sec)
```

8. Write a query to display all the employees whose last name starts with “K”.

```
MariaDB [company]> select first_name,last_name from employees where last_name like 'K%';
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| Kiran      | Koshy     |
+------------+-----------+
1 row in set (0.001 sec)
```

9. Display name of the employees whose hire dates are between 2015 and 2019.

```
MariaDB [company]> select first_name,last_name,hire_date from employees where hire_date between '2015-01-01' and '2019-12-31';
+------------+-----------+------------+
| first_name | last_name | hire_date  |
+------------+-----------+------------+
| Rohan      | Matthew   | 2016-07-22 |
| Jithin     | Lal       | 2016-09-20 |
| Jay        | Sivan     | 2017-01-16 |
| Paul       | John      | 2018-03-13 |
| Riya       | Manuel    | 2018-08-03 |
| Issac      | Abraham   | 2018-07-14 |
+------------+-----------+------------+
6 rows in set (0.001 sec)
```

10. Write a query to display jobs where the maximum salary is less than 5000.

```
MariaDB [company]> select * from jobs where max_salary<5000;
+--------+-----------+------------+------------+
| job_id | job_title | min_salary | max_salary |
+--------+-----------+------------+------------+
|      2 | manager   |       3000 |       4000 |
+--------+-----------+------------+------------+
1 row in set (0.000 sec)
```

11. Write a query to display the name of the employees who has dependents.

```
MariaDB [company]> select distinct e.first_name,e.last_name,d.relationship from employees e join dependents d on e.employee_id=d.empl
oyee_id;
+------------+-----------+--------------+
| first_name | last_name | relationship |
+------------+-----------+--------------+
| Hazel      | Ann       | Child        |
| Rohan      | Matthew   | Spouse       |
| Riya       | Manuel    | Spouse       |
+------------+-----------+--------------+
3 rows in set (0.001 sec)
```

12. Write a query to display name of the employees who were hired in 2017.

```
MariaDB [company]> select first_name,last_name,hire_date from employees where year(hire_date)='2017';
+------------+-----------+------------+
| first_name | last_name | hire_date  |
+------------+-----------+------------+
| Jay        | Sivan     | 2017-01-16 |
| Kiran      | Koshy     | 2017-04-15 |
+------------+-----------+------------+
2 rows in set (0.001 sec)
```

13. Write a query to insert an employee “Satya Paul” in department 6.

```
MariaDB [company]> insert into employees values ('2233','Satya','Paul','satya@gmail.com','9022390089','2025/03/03','1','3000','230','
3');
Query OK, 1 row affected (0.023 sec)
```

```
MariaDB [company]> select * from employees;
+-------------+------------+-----------+-----------------+--------------+------------+--------+--------+------------+---------------
+
| employee_id | first_name | last_name | email           | phone_number | hire_date  | job_id | salary | manager_id | department_id
|
+-------------+------------+-----------+-----------------+--------------+------------+--------+--------+------------+---------------
+
|        1111 | Rohan      | Matthew   | rohan@gmail.com | 9999999999   | 2016-07-22 |      1 |  35000 |        230 |          2000
|
|        1112 | Hazel      | Ann       | hazel@gmail.com | 9809097890   | 2020-09-20 |      1 |  30000 |        239 |          2000
|
|        1113 | Jithin     | Lal       | jithin@gmail.com | 9890768909   | 2016-09-20 |      2 |   5000 |        233 |          2001
|
|        1114 | Jay        | Sivan     | jay@gmail.com   | 9777777700   | 2017-01-16 |     19 |   4000 |        239 |          2001
|
|        1119 | Paul       | John      | paul@gmail.com  | 9999999999   | 2018-03-13 |      1 |   8000 |        233 |          2000
|
|        2221 | Riya       | Manuel    | riya@gmail.com  | 9656445566   | 2018-08-03 |      2 |   8000 |        239 |             3
|
|        2222 | Issac      | Abraham   | issac@gmail.com | 9777777777   | 2018-07-14 |      2 |  10000 |        299 |             3
|
|        2233 | Satya      | Paul      | satya@gmail.com | 9022390089   | 2025-03-03 |      1 |   3000 |        230 |             3
|
+-------------+------------+-----------+-----------------+--------------+------------+--------+--------+------------+---------------
+
8 rows in set (0.001 sec)
```

14. Write a query to delete the shipping department.

```
MariaDB [company]> select * from departments where department_name = 'shipping';
+---------------+-----------------+-------------+
| department_id | department_name | location_id |
+---------------+-----------------+-------------+
|             8 | Shipping        |        1000 |
+---------------+-----------------+-------------+
1 row in set (0.001 sec)
```

```
MariaDB [company]> delete from departments where department_name = 'shipping';
Query OK, 1 row affected (0.005 sec)

MariaDB [company]> select * from departments where department_name = 'shipping';
Empty set (0.001 sec)
```

15. Display names of all departments, its city, country, and region names with a single query.

```
MariaDB [company]> select d.department_name,l.city,c.country_name,r.region_name from departments d join locations l on d.location_id=
l.location_id join countries c on l.country_id=c.country_id join regions r on c.region_id=r.region_id;
+------------------+------------------+--------------+-------------+
| department_name  | city             | country_name | region_name |
+------------------+------------------+--------------+-------------+
| Manual Testing   | Hamburg Ottensen | Germany      | Europe      |
| Automatic Testing | Mumbai          | India        | Asia        |
| Ordering         | Hamburg Ottensen | Germany      | Europe      |
| Sales            | Chula Vista      | Chicago      | America     |
| Marketing        | Hamburg Ottensen | Germany      | Europe      |
| Development      | Mumbai           | India        | Asia        |
+------------------+------------------+--------------+-------------+
6 rows in set (0.001 sec)
```

PROGRAM 3

AIM: Apply DCL and TCL commands to impose restrictions on databases.

1. Connect as ‘root’

```
Setting environment for using XAMPP for Windows.
hanna@LAPTOP-HANNA c:\xampp
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 105
Server version: 10.4.32-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

   a) Create a user s2 with a password

```
MariaDB [company]> create user s2 identified by 'abcd';
Query OK, 0 rows affected (0.025 sec)
```

   b) Grant SELECT and UPDATE privileges on the employee table to ‘s2’

```
MariaDB [company]> grant select, update on employees to s2;
Query OK, 0 rows affected (0.003 sec)
```

2. Connect as ‘s2’ user and start a transaction

```
hanna@LAPTOP-HANNA c:\xampp
# mysql -u s2 -p
Enter password: ****
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 107
Server version: 10.4.32-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

   a) Select the database ‘company’

```
MariaDB [(none)]> use company;
Database changed
```

b) attempt to update employee salary

```
MariaDB [company]> select * from employees;
+-------------+------------+-----------+------------------+--------------+------------+--------+--------+------------+---------------+
| employee_id | first_name | last_name | email            | phone_number | hire_date  | job_id | salary | manager_id | department_id |
+-------------+------------+-----------+------------------+--------------+------------+--------+--------+------------+---------------+
|        1111 | Rohan      | Matthew   | rohan@gmail.com  | 9999999999   | 2016-07-22 |      1 |  35000 |        230 |          2000 |
|        1112 | Hazel      | Ann       | hazel@gmail.com  | 9809097890   | 2020-09-20 |      1 |  30000 |        239 |          2000 |
|        1113 | Jithin     | Lal       | jithin@gmail.com | 9890768909   | 2016-09-20 |      2 |   5000 |        233 |          2001 |
|        1114 | Jay        | Sivan     | jay@gmail.com    | 9777777700   | 2017-01-16 |     19 |   4000 |        239 |          2001 |
|        1119 | Paul       | John      | paul@gmail.com   | 9999999999   | 2018-03-13 |      1 |   8000 |        233 |          2000 |
|        2221 | Riya       | Manuel    | riya@gmail.com   | 9656445566   | 2018-08-03 |      2 |   8000 |        239 |             3 |
|        2222 | Issac      | Abraham   | issac@gmail.com  | 9777777777   | 2018-07-14 |      2 |  10000 |        299 |             3 |
|        2233 | Satya      | Paul      | satya@gmail.com  | 9022390089   | 2025-03-03 |      1 |   3000 |        230 |             3 |
+-------------+------------+-----------+------------------+--------------+------------+--------+--------+------------+---------------+
8 rows in set (0.001 sec)

MariaDB [company]> update employees set salary='2000' where employee_id=1111;
Query OK, 1 row affected (0.025 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

c) check the update record.

```
MariaDB [company]> select employee_id,salary from employees;
+-------------+--------+
| employee_id | salary |
+-------------+--------+
|        1111 |   2000 |
|        1112 |  30000 |
|        1113 |   5000 |
|        1114 |   4000 |
|        1119 |   8000 |
|        2221 |   8000 |
|        2222 |  10000 |
|        2233 |   3000 |
+-------------+--------+
8 rows in set (0.001 sec)
```

3. Set a savepoint ‘A’.

```
MariaDB [company]> begin;
Query OK, 0 rows affected (0.000 sec)

MariaDB [company]> savepoint A;
Query OK, 0 rows affected (0.001 sec)
```

a) attempt to update employee salary

```
MariaDB [company]> select employee_id,salary from employees;
+-------------+--------+
| employee_id | salary |
+-------------+--------+
|        1111 |   2000 |
|        1112 |  30000 |
|        1113 |   5000 |
|        1114 |   4000 |
|        1119 |   8000 |
|        2221 |   8000 |
|        2222 |  10000 |
|        2233 |   3000 |
+-------------+--------+
8 rows in set (0.001 sec)

MariaDB [company]> update employees set salary='20000' where employee_id=1111;
Query OK, 1 row affected (0.025 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

b) check the update record

```
MariaDB [company]> select employee_id,salary from employees;
+-------------+--------+
| employee_id | salary |
+-------------+--------+
|        1111 |  20000 |
|        1112 |  30000 |
|        1113 |   5000 |
|        1114 |   4000 |
|        1119 |   8000 |
|        2221 |   8000 |
|        2222 |  10000 |
|        2233 |   3000 |
+-------------+--------+
8 rows in set (0.001 sec)
```

4. Roll back to savepoint ‘A’

```
MariaDB [company]> rollback to A;
Query OK, 0 rows affected (0.028 sec)
```

a) check employee’s salary after the rollback (should be unchanged)

```
MariaDB [company]> select employee_id,salary from employees;
+-------------+--------+
| employee_id | salary |
+-------------+--------+
|        1111 |   2000 |
|        1112 |  30000 |
|        1113 |   5000 |
|        1114 |   4000 |
|        1119 |   8000 |
|        2221 |   8000 |
|        2222 |  10000 |
|        2233 |   3000 |
+-------------+--------+
8 rows in set (0.001 sec)
```

b) attempt to delete empid = 1

```
MariaDB [company]> delete from employees where employee_id=1111;
ERROR 1142 (42000): DELETE command denied to user 's2'@'localhost' for table `company`.`employees`
```

c) commit the transaction.

```
MariaDB [company]> commit;
Query OK, 0 rows affected (0.001 sec)
```

5. Connect as ‘root’ user

```
hanna@LAPTOP-HANNA c:\xampp
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 136
Server version: 10.4.32-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

a) use database company

```
MariaDB [(none)]> use company;
Database changed
```

b) revoke update privilege on a table from ‘s2’ user

```
MariaDB [company]> revoke update on employees from s2;
Query OK, 0 rows affected (0.003 sec)
```

6. Connect as ‘s2’ user

```
hanna@LAPTOP-HANNA c:\xampp
# mysql -u s2 -p
Enter password: ****
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 137
Server version: 10.4.32-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

a) select database ‘company’

```
MariaDB [(none)]> use company;
Database changed
```

b) attempt to update an employee’s salary

```
MariaDB [company]> update employees set salary='20000' where employee_id=1111;
ERROR 1142 (42000): UPDATE command denied to user 's2'@'localhost' for table `company`.`employees`
```

PROGRAM 4

AIM: Create an application to use joins for query optimization

1. Create ‘University’ database with tables for students, courses, and grades.

```
MariaDB [(none)]> create database university;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> use university;
Database changed
MariaDB [university]> create table students(stud_id int primary key, stud_name varchar(20), major varchar(25));
Query OK, 0 rows affected (0.011 sec)
```

```
MariaDB [university]> create table courses(course_id int primary key, course_name varchar(30));
Query OK, 0 rows affected (0.013 sec)
```

```
MariaDB [university]> create table grades(grade_id int primary key, stud_id int, course_id int,grade varchar(3), foreign key(stud_id) references students(stud_id), foreign
key(course_id) references courses(course_id));
Query OK, 0 rows affected (0.030 sec)
```

2. Insert values into each table.

```
MariaDB [university]> insert into students values(1, 'Alice Johnson', 'Computer Science'),(2, 'Bob Smith', 'Mathematics'),(3, 'Charlie Davis', 'Computer Science'),(4, 'Dian
a Green', 'Physics'),(5, 'Edward Williams', 'Computer Science'),(6, 'Fiona White', 'Mathematics');
Query OK, 6 rows affected (0.004 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

```
MariaDB [university]> insert into courses values(101, 'Math 101'),(102, 'CS 101'),(103, 'Physics 101'),(104,'Math 102'),(105,'CS 102');
Query OK, 5 rows affected (0.005 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
MariaDB [university]> insert into grades values(1,1,102,85),(2,1,105,90),(3,2,101,78),(4,2,104,82),(5,3,102,88),(6,3,105,92),(7,4,103,75),(8,5,102,80),(9,5,105,87),(10,6,10
1,95),(11,6,104,80);
Query OK, 11 rows affected (0.002 sec)
Records: 11  Duplicates: 0  Warnings: 0
```

3. Find the names of students who scored an 'A' grade in the 'Math' course using ‘IN’ operator and subquery.

```
MariaDB [university]> SELECT stud_name FROM students WHERE student_id IN (SELECT student_id FROM grades WHERE course_id = (SELECT course_id FROM courses WHERE course_name =
 'Math 101') AND grade >= 90);
+-------------+
| stud_name   |
+-------------+
| Fiona White |
+-------------+
1 row in set (0.002 sec)
```

4. List the courses that have at least one student who major in 'Computer Science' using subquery and ‘IN’ operator.

```
MariaDB [university]> SELECT DISTINCT course_name FROM courses WHERE course_id IN (SELECT course_id FROM grades WHERE student_id IN (SELECT student_id FROM students WHERE m
ajor = 'Computer Science'));
+-------------+
| course_name |
+-------------+
| CS 101      |
| CS 102      |
+-------------+
2 rows in set (0.001 sec)
```

5. Find the average grade for a specific student in all their courses using subquery and ‘AVG’ function.

```
MariaDB [university]> SELECT AVG(grade) FROM grades WHERE student_id = (SELECT student_id FROM students WHERE stud_name = 'Bob Smith');
+------------+
| AVG(grade) |
+------------+
|         80 |
+------------+
1 row in set (0.001 sec)
```

6. Count the number of students who have taken each course using ‘DISTINCT’ and ‘COUNT’ function.

```
MariaDB [university]> SELECT course_name, (SELECT COUNT(DISTINCT student_id) FROM grades WHERE course_id = courses.course_id) AS num_students FROM courses;
+-------------+--------------+
| course_name | num_students |
+-------------+--------------+
| Math 101    |            2 |
| CS 101      |            3 |
| Physics 101 |            1 |
| Math 102    |            2 |
| CS 102      |            3 |
+-------------+--------------+
5 rows in set (0.001 sec)
```

7. Find the students who have taken all available courses using ‘GROUP BY’, ‘DISTINCT’, ‘COUNT’, ‘JOIN’, ‘HAVING’.

```
MariaDB [university]> SELECT s.student_id, s.stud_name FROM students s JOIN grades g ON s.student_id = g.student_id GROUP BY s.student_id HAVING COUNT(DISTINCT g.course_id)
 = (SELECT COUNT(*) FROM courses);
Empty set (0.001 sec)
```

8. Retrieve all students who are majoring in "Computer Science" using ‘WHERE’ clause.

```
MariaDB [university]> SELECT * FROM students WHERE Major = 'Computer Science';
+------------+-----------------+------------------+
| student_id | stud_name       | major            |
+------------+-----------------+------------------+
|          1 | Alice Johnson   | Computer Science |
|          3 | Charlie Davis   | Computer Science |
|          5 | Edward Williams | Computer Science |
+------------+-----------------+------------------+
3 rows in set (0.001 sec)
```

9. Find all students whose name starts with "A" using ‘LIKE’ operator.

```
MariaDB [university]> SELECT * FROM students WHERE stud_name LIKE 'A%';
+------------+---------------+------------------+
| student_id | stud_name     | major            |
+------------+---------------+------------------+
|          1 | Alice Johnson | Computer Science |
+------------+---------------+------------------+
1 row in set (0.001 sec)
```

10. Find the average grade for each course using ‘AVG’ function, ‘JOIN’, ‘GROUP BY’.

```
MariaDB [university]> SELECT c.course_id, c.course_name, AVG(g.grade) FROM courses c JOIN grades g ON c.course_id = g.course_id GROUP BY c.course_id, c.course_name;
+-----------+-------------+------------------+
| course_id | course_name | AVG(g.grade)     |
+-----------+-------------+------------------+
|       101 | Math 101    |             86.5 |
|       102 | CS 101      | 84.33333333333333 |
|       103 | Physics 101 |               75 |
|       104 | Math 102    |               81 |
|       105 | CS 102      | 89.66666666666667 |
+-----------+-------------+------------------+
5 rows in set (0.001 sec)
```

11. Find all courses with an average grade greater than 85 using ‘HAVING’ clause and ‘AVG’ function.

```
MariaDB [university]> SELECT c.course_id, c.course_name, AVG(g.grade) FROM courses c JOIN grades g ON c.course_id = g.course_id GROUP BY c.course_id, c.course_name HAVING A
VG(g.grade) > 85;
+-----------+-------------+------------------+
| course_id | course_name | AVG(g.grade)     |
+-----------+-------------+------------------+
|       101 | Math 101    |             86.5 |
|       105 | CS 102      | 89.66666666666667 |
+-----------+-------------+------------------+
2 rows in set (0.002 sec)
```

12. List all students ordered by their names in ascending order using ‘ORDER BY’ clause.

```
MariaDB [university]> SELECT * FROM students ORDER BY stud_name ASC;
+------------+------------------+--------------------+
| student_id | stud_name        | major              |
+------------+------------------+--------------------+
|          1 | Alice Johnson    | Computer Science   |
|          2 | Bob Smith        | Mathematics        |
|          3 | Charlie Davis    | Computer Science   |
|          4 | Diana Green      | Physics            |
|          5 | Edward Williams  | Computer Science   |
|          6 | Fiona White      | Mathematics        |
+------------+------------------+--------------------+
6 rows in set (0.001 sec)
```

13. Find the names of all students along with the courses they are enrolled in using ‘INNER JOIN’.

```
MariaDB [university]> SELECT s.stud_name, c.course_name FROM students s INNER JOIN grades g ON s.student_id = g.student_id INNER JOIN courses c ON g.course_id = c.course_id
;
+----------------+-------------+
| stud_name      | course_name |
+----------------+-------------+
| Bob Smith      | Math 101    |
| Fiona White    | Math 101    |
| Alice Johnson  | CS 101      |
| Charlie Davis  | CS 101      |
| Edward Williams | CS 101     |
| Diana Green    | Physics 101 |
| Bob Smith      | Math 102    |
| Fiona White    | Math 102    |
| Alice Johnson  | CS 102      |
| Charlie Davis  | CS 102      |
| Edward Williams | CS 102     |
+----------------+-------------+
11 rows in set (0.001 sec)
```

14. List all students and the courses they are enrolled in, including students who are not enrolled in any courses using ‘LEFT JOIN’.

```
MariaDB [university]> SELECT s.stud_name, c.course_name FROM students s LEFT JOIN grades g ON s.student_id = g.student_id LEFT JOIN courses c ON g.course_id = c.course_id;
+----------------+-------------+
| stud_name      | course_name |
+----------------+-------------+
| Alice Johnson  | CS 101      |
| Alice Johnson  | CS 102      |
| Bob Smith      | Math 101    |
| Bob Smith      | Math 102    |
| Charlie Davis  | CS 101      |
| Charlie Davis  | CS 102      |
| Diana Green    | Physics 101 |
| Edward Williams | CS 101     |
| Edward Williams | CS 102     |
| Fiona White    | Math 101    |
| Fiona White    | Math 102    |
+----------------+-------------+
11 rows in set (0.001 sec)
```

15. Find students who have received a grade greater than the average grade in their respective courses using ‘ANY’ and comparison operators

```
MariaDB [university]> SELECT s.student_name, c.course_name, g.grade
    -> FROM students s
    -> JOIN grades g ON s.student_id = g.student_id
    -> JOIN courses c ON g.course_id = c.course_id
    -> WHERE g.grade > ANY (
    ->     SELECT AVG(grade)
    ->     FROM grades
    ->     WHERE course_id = g.course_id
    -> );
+---------------+-------------+-------+
| student_name  | course_name | grade |
+---------------+-------------+-------+
| Fiona White   | Math 101    | 95    |
| Alice Johnson | CS 101      | 85    |
| Charlie Davis | CS 101      | 88    |
| Bob Smith     | Math 102    | 82    |
| Alice Johnson | CS 102      | 90    |
| Charlie Davis | CS 102      | 92    |
+---------------+-------------+-------+
6 rows in set (0.002 sec)
```

16. Find students who have received grades greater than or equal to all grades in the "Math 101" course using ‘ALL’ and comparison operators.

```
MariaDB [university]> SELECT s.student_name, g.grade
    -> FROM students s
    -> JOIN grades g ON s.student_id = g.student_id
    -> WHERE g.grade >= ALL (
    ->     SELECT grade
    ->     FROM grades
    ->     WHERE course_id = (SELECT course_id FROM courses WHERE course_name = 'Math 101')
    -> );
+-------------+-------+
| student_name | grade |
+-------------+-------+
| Fiona White  | 95    |
+-------------+-------+
1 row in set (0.001 sec)
```

17. Find all students who have received a grade in at least one course using ‘EXISTS’.

```
MariaDB [university]> SELECT s.student_name
    -> FROM students s
    -> WHERE EXISTS (
    ->     SELECT 1
    ->     FROM grades g
    ->     WHERE g.student_id = s.student_id
    -> );
+-----------------+
| student_name    |
+-----------------+
| Alice Johnson   |
| Bob Smith       |
| Charlie Davis   |
| Diana Green     |
| Edward Williams |
| Fiona White     |
+-----------------+
6 rows in set (0.001 sec)
```

18. List all unique courses that students are enrolled in using ‘DISTINCT’ and ‘INNER JOIN’.

```
MariaDB [university]> SELECT DISTINCT c.course_name FROM courses c INNER JOIN grades g ON c.course_id = g.course_id;
+-------------+
| course_name |
+-------------+
| Math 101    |
| CS 101      |
| Physics 101 |
| Math 102    |
| CS 102      |
+-------------+
5 rows in set (0.001 sec)
```

19. Find all students who are majoring in "Computer Science" and have received a grade higher than 80 in any course using ‘DISTINCT’ and ‘AND’ operator.

```
MariaDB [university]> SELECT DISTINCT s.stud_name FROM students s INNER JOIN grades g ON s.student_id = g.student_id WHERE s.Major = 'Computer Science' AND g.grade > 80;
+----------------+
| stud_name      |
+----------------+
| Alice Johnson  |
| Charlie Davis  |
| Edward Williams |
+----------------+
3 rows in set (0.002 sec)
```

20. Find students who have received grades between 70 and 85 in any course using ‘DISTINCT’ and ‘BETWEEN’ operator.

```
MariaDB [university]> SELECT DISTINCT s.stud_name FROM students s INNER JOIN grades g ON s.student_id = g.student_id WHERE g.grade BETWEEN 70 AND 85;
+----------------+
| stud_name      |
+----------------+
| Alice Johnson  |
| Bob Smith      |
| Diana Green    |
| Edward Williams |
| Fiona White    |
+----------------+
5 rows in set (0.001 sec)
```