SQL LAB-1
RDBMS, MYSQL

NAME: Lingutla Rajitha

ID: AF0366374

QUESTIONS

Lab 1. Create a Database & Table Using MySQL Command-Line Client.

- Create a database with the name StudentManagementSystem.

1) Create a table with named Student with attributes:

- StudentID (Primary Key)
- FirstName
- LastName
- DateOfBirth
- Gender
- Email
- Phone

2) Create a table with name Course with attributes:

- CourseID (Primary Key)
- CourseTitle
- Credits

3) Create a table with named Instructor with attributes:

- InstructorID (Primary Key)
- FirstName
- LastName
- Email

4) Create a table with named Enrollment with attributes:

- EnrollmentID (Primary Key)
- EnrollmentDate
- StudentID(Foreign key)
- CourseID(Foreign Key)

- InstructorID(Foreign key)

5) Create a table with named Score with attributes:

- ScoreID (Primary Key)
- CourseID (Foreign key)
- StudentID (Foreign Key)
- DateOfExam
- CreditObtained

6) Create a table with named Feedback with attributes:

- FeedbackID (Primary Key)
- StudentID (Foreign key)
- Date
- InstructorName
- Feedback

ChatGPT Exercise

- Using ChatGPT generate the Database design

  Scenario: Implementing Database Design

  The database should store emergency contact information for each employee. This information is crucial for situations where immediate contact with family or emergency contacts is necessary. The design should consider privacy and security measures for sensitive contact details.

- Use the chatGPT prompt to formulate the database design for the described scenario.

● Create a database with the name StudentManagementSystem.

Code:

```
mysql> create database StudentManagementSystem;
Query OK, 1 row affected (0.07 sec)
```

Output:

```
mysql> show databases;
+------------------------+
| Database               |
+------------------------+
| information_schema     |
| mysql                  |
| performance_schema     |
| studentmanagementsystem |
| sys                    |
+------------------------+
5 rows in set (0.06 sec)
```

1) Create a table with named Student with attributes:

StudentID (Primary Key), FirstName, LastName, DateOfBirth, Gender, Email, Phone

Code:

```
mysql> Use Studentmanagementsystem;
Database changed
```

```
mysql> create table Student(
    -> StudentID int not null Primary key,
    -> FirstName varchar(25) not null,
    -> LastName varchar(25) not null,
    -> DateOfBirth Date not null,
    -> Gender varchar(10) not null,
    -> Email varchar(25) unique not null,
    -> Phone int not null unique check(Phone=10)
    -> );
Query OK, 0 rows affected (0.13 sec)
```

Output:

```
mysql> desc Student;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| StudentID   | int         | NO   | PRI | NULL    |       |
| FirstName   | varchar(25) | NO   |     | NULL    |       |
| LastName    | varchar(25) | NO   |     | NULL    |       |
| DateOfBirth | date        | NO   |     | NULL    |       |
| Gender      | varchar(10) | NO   |     | NULL    |       |
| Email       | varchar(25) | NO   | UNI | NULL    |       |
| Phone       | int         | NO   | UNI | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
7 rows in set (0.03 sec)
```

2) Create a table with name Course with attributes:

CourseID (Primary Key), CourseTitle, Credits

Code:

```
mysql> Create table Course(
    -> CourseID int not null Primary key,
    -> CourseTitle varchar(60) not null,
    -> Credits int not null);
Query OK, 0 rows affected (0.04 sec)
```

Output:

```
mysql> desc Course;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| CourseID    | int         | NO   | PRI | NULL    |       |
| CourseTitle | varchar(60) | NO   |     | NULL    |       |
| Credits     | int         | NO   |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

3) Create a table with named Instructor with attributes:

InstructorID (Primary Key), FirstName, LastName, Email

Code:

```
mysql> create table Instructor(
    -> InstructorID int not null Primary key,
    -> FirstName varchar(25) not null,
    -> LastName varchar(25) not null,
    -> Email varchar(60) not null unique);
Query OK, 0 rows affected (0.04 sec)
```

Output:

```
mysql> desc Instructor;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| InstructorID | int         | NO   | PRI | NULL    |       |
| FirstName    | varchar(25) | NO   |     | NULL    |       |
| LastName     | varchar(25) | NO   |     | NULL
  |       |
| Email        | varchar(60) | NO   | UNI | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

4) Create a table with named Enrollment with attributes:

EnrollmentID (Primary Key), EnrollmentDate,StudentID(Foreignkey),

CourseID(Foreign Key),InstructorID(Foreign key)

Code:

```
mysql> Create table Enrollment(
    -> EnrollmentID int not null Primary key,
    -> EnrollmentDate Date not null,
    -> StudentID int not null,
    -> CourseID int not null,
    -> InstructorID int not null,
    -> Foreign key (StudentID) References Student(StudentID),
    -> Foreign key (CourseID) References Course(CourseID),
    -> Foreign key (InstructorID) References Instructor(InstructorID));
Query OK, 0 rows affected (0.08 sec)
```

Output:

```
mysql> desc Enrollment;
+----------------+------+------+-----+---------+-------+
| Field          | Type | Null | Key | Default | Extra |
+----------------+------+------+-----+---------+-------+
| EnrollmentID   | int  | NO   | PRI | NULL    |       |
| EnrollmentDate | date | NO   |     | NULL    |       |
| StudentID      | int  | NO   | MUL | NULL    |       |
| CourseID       | int  | NO   | MUL | NULL    |       |
| InstructorID   | int  | NO   | MUL | NULL    |       |
+----------------+------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

5) Create a table with named Score with attributes:

ScoreID (Primary Key), CourseID (Foreign key), StudentID (Foreign Key), DateOfExam, CreditObtained

Code:

```
mysql> create table Score(
    -> ScoreID int not null Primary key,
    -> CourseID int not null,
    -> StudentID int not null,
    -> DateofExam Date not null,
    -> CreditObtained Decimal(5,2) not null,
    -> Foreign key (CourseID) References Course(CourseID),
    -> Foreign key (StudentID) References Student (StudentID)
    -> );
Query OK, 0 rows affected (0.06 sec)
```

Output:

```
mysql> desc Score;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| ScoreID        | int          | NO   | PRI | NULL    |       |
| CourseID       | int          | NO   | MUL | NULL    |       |
| StudentID      | int          | NO   | MUL | NULL    |       |
| DateofExam     | date         | NO   |     | NULL    |       |
| CreditObtained | decimal(5,2) | NO   |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

6) Create a table with named Feedback with attributes:

FeedbackID (Primary Key), StudentID (Foreign key), Date, InstructorName, Feedback

Code:

```
mysql> create table Feedback(
    -> FeedbackID int not null Primary key,
    -> StudentID int not null,
    -> FeedbackDate Date not null,
    -> InstructorName varchar(25) not null,
    -> Feedback Text not null,
    -> Foreign key (StudentID) References Student (StudentID));
Query OK, 0 rows affected (0.04 sec)
```

Output:

```
mysql> desc Feedback;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| FeedbackID     | int         | NO   | PRI | NULL    |       |
| StudentID      | int         | NO   | MUL | NULL    |       |
| FeedbackDate   | date        | NO   |     | NULL    |       |
| InstructorName | varchar(25) | NO   |     | NULL    |       |
| Feedback       | text        | NO   |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

ChatGPT Exercise

- Using ChatGPT generate the Database design

  Scenario: Implementing Database Design

  The database should store emergency contact information for each employee. This information is crucial for situations where immediate contact with family or emergency contacts is necessary. The design should consider privacy and security measures for sensitive contact details.

Code:

```
mysql> create database EmployeeInfo;
Query OK, 1 row affected (0.07 sec)

mysql> Use EmployeeInfo;
Database changed
```

```
mysql> create table Employee(
    -> EmployeeID int not null Primary key Auto_Increment,
    -> FirstName varchar(20) not null,
    -> LastName varchar(20) not null,
    -> DateofBirth Date not null,
    -> Gender varchar(20) not null unique,
    -> Phone varchar(10) not null);
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> create table EmergencyContact(
    -> ContactID int not null Primary key Auto_Increment,
    -> EmployeeID int not null,
    -> ContactName varchar(25) not null,
    -> Relationship varchar(15) not null,
    -> ContactEmail varchar(60),
    -> ContactAdress varchar(100),
    -> Foreign key (EmployeeID) References Employee(EmployeeID));
Query OK, 0 rows affected (0.04 sec)
```

Output:

```
mysql> desc Employee;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| EmployeeID  | int         | NO   | PRI | NULL    | auto_increment |
| FirstName   | varchar(20) | NO   |     | NULL    |                |
| LastName    | varchar(20) | NO   |     | NULL    |                |
| DateofBirth | date        | NO   |     | NULL    |                |
| Gender      | varchar(20) | NO   | UNI | NULL    |                |
| Phone       | varchar(10) | NO   |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
6 rows in set (0.03 sec)

mysql> desc EmergencyContact;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| ContactID     | int         | NO   | PRI | NULL    | auto_increment |
| EmployeeID    | int         | NO   | MUL | NULL    |                |
| ContactName   | varchar(25) | NO   |     | NULL    |                |
| Relationship  | varchar(15) | NO   |     | NULL    |                |
| ContactEmail  | varchar(60) | YES  |     | NULL    |                |
| ContactAdress | varchar(100)| YES  |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

- Use the chatGPT prompt to formulate the database design for the described scenario.

For the scenario of storing emergency contact information for each employee, the database design should consider the following tables:

1) Employee Table: This table stores general information about employees.

   - EmployeeID (Primary Key): Unique identifier for each employee.
   - FirstName: First name of the employee.
   - LastName: Last name of the employee.
   - DateOfBirth: Date of birth of the employee.
   - Gender: Gender of the employee.
   - Email: Email address of the employee.
   - Phone: Phone number of the employee.

2) EmergencyContact Table: This table stores emergency contact information for each employee.

   - ContactID (Primary Key): Unique identifier for each emergency contact.
   - EmployeeID (Foreign Key): References the Employee table to establish a relationship.
   - ContactName: Name of the emergency contact person.
   - Relationship: Relationship of the emergency contact person with the employee (e.g., spouse, parent, sibling).
   - Phone: Phone number of the emergency contact person.
   - Email: Email address of the emergency contact person.

With this design, each employee can have multiple emergency contacts associated with them, allowing for efficient storage and retrieval of essential contact information during emergencies.