

SQL LAB-2

RDBMS, MYSQL

NAME: Pravallika Ullikanti

ID: AF0366355

QUESTIONS

Lab 1. Day-2 Lab

(PN: ChatGPT exercise is mandatory)

Lab 1. Use the Database and table from Day 1 lab. Insert 5 records in each table and retrieve data from all tables and display.

- 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)
```

i) A table with named Students with attributes:

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

Code:

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

Insert five records:

```
mysql> INSERT INTO Students (StudentID, FirstName, LastName, DateOfBirth, Gender, Email, Phone)
-> VALUES
-> (481, 'Rajitha', 'Lingutla', '2002-06-21', 'Female', 'rajitha@gmail.com', '1934756782'),
-> (482, 'Bob', 'Stone', '1999-02-20', 'Male', 'bob.stone@gmail.com', '8987654329'),
-> (483, 'Cara', 'Wins', '2001-03-25', 'Female', 'cara.wins@example.com', '1122334458'),
-> (484, 'David', 'Johnson', '2002-04-30', 'Male', 'david.johnson@example.com', '2233445568'),
-> (485, 'Eva', 'Brown', '1998-05-10', 'Female', 'eva.brown@example.com', '3344556674');
Query OK, 5 rows affected (0.05 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

Display the data which is retrieved from the table Students:

```
mysql> select *from Students;
```

StudentID	FirstName	LastName	DateOfBirth	Gender	Email	Phone
481	Rajitha	Lingutla	2002-06-21	Female	rajitha@gmail.com	1934756782
482	Bob	Stone	1999-02-20	Male	bob.stone@gmail.com	8987654329
483	Cara	Wins	2001-03-25	Female	cara.wins@example.com	1122334458
484	David	Johnson	2002-04-30	Male	david.johnson@example.com	2233445568
485	Eva	Brown	1998-05-10	Female	eva.brown@example.com	3344556674

```
5 rows in set (0.00 sec)
```

2) A table with name Course with attributes:

CourseID (Primary Key), CourseTitle, Credits

Insert five records:

```
mysql> INSERT INTO Course (CourseID, CourseTitle, Credits)
-> VALUES
-> (1, 'Introduction to Computer Science', 3),
-> (2, 'Data Structures and Algorithms', 4),
-> (3, 'Database Systems', 3),
-> (4, 'Operating Systems', 4),
-> (5, 'Web Development', 3);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

Display the data which is retrieved from the table Course:

```
mysql> Select *from Course;
```

CourseID	CourseTitle	Credits
1	Introduction to Computer Science	3
2	Data Structures and Algorithms	4
3	Database Systems	3
4	Operating Systems	4
5	Web Development	3

```
5 rows in set (0.00 sec)
```

3) A table with named Instructor with attributes:

InstructorID (Primary Key), FirstName, LastName, Email

Insert five records:

```
mysql> INSERT INTO Instructor (InstructorID, FirstName, LastName, Email)
-> VALUES
-> (1, 'John', 'Doe', 'john.doe@example.com'),
-> (2, 'Jane', 'Smith', 'jane.smith@example.com'),
-> (3, 'Mike', 'Johnson', 'mike.johnson@example.com'),
-> (4, 'Sara', 'Williams', 'sara.williams@example.com'),
-> (5, 'Tom', 'Brown', 'tom.brown@example.com');
Query OK, 5 rows affected (0.05 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

To display the data retrieved from the Instructor table:

```
mysql> Select *from Instructor;
+-----+-----+-----+-----+
| InstructorID | FirstName | LastName | Email |
+-----+-----+-----+-----+
| 1 | John | Doe | john.doe@example.com |
| 2 | Jane | Smith | jane.smith@example.com |
| 3 | Mike | Johnson | mike.johnson@example.com |
| 4 | Sara | Williams | sara.williams@example.com |
| 5 | Tom | Brown | tom.brown@example.com |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

4) A table with named Enrollment with attributes:

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

CourseID(Foreign Key), InstructorID(Foreign key)

Insert five records:

```
mysql> INSERT INTO Enrollment (EnrollmentID, EnrollmentDate, StudentID, CourseID, InstructorID)
-> VALUES
-> (221, '2023-01-01', 481, 1, 1),
-> (222, '2023-01-02', 482, 2, 2),
-> (223, '2023-01-03', 483, 3, 3),
-> (224, '2023-01-04', 484, 4, 4),
-> (225, '2023-01-05', 485, 5, 5);
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

To display the data retrieved from the Enrollment table:

```
mysql> Select *from Enrollment;
+-----+-----+-----+-----+-----+
| EnrollmentID | EnrollmentDate | StudentID | CourseID | InstructorID |
+-----+-----+-----+-----+-----+
| 221 | 2023-01-01 | 481 | 1 | 1 |
| 222 | 2023-01-02 | 482 | 2 | 2 |
| 223 | 2023-01-03 | 483 | 3 | 3 |
| 224 | 2023-01-04 | 484 | 4 | 4 |
| 225 | 2023-01-05 | 485 | 5 | 5 |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

5) A table with named Score with attributes:

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

Insert the five records:

```
mysql> INSERT INTO Score (ScoreID, CourseID, StudentID, DateofExam, CreditObtained)
-> VALUES
-> (21, 1, 481, '2023-01-10', 2.5),
-> (22, 2, 482, '2023-01-11', 3.0),
-> (23, 3, 483, '2023-01-12', 2.8),
-> (24, 4, 484, '2023-01-13', 3.2),
-> (25, 5, 485, '2023-01-14', 2.7);
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

To display the data retrieved from the Score table:

```
mysql> Select *from Score;
+-----+-----+-----+-----+-----+
| ScoreID | CourseID | StudentID | DateofExam | CreditObtained |
+-----+-----+-----+-----+-----+
| 21 | 1 | 481 | 2023-01-10 | 2.50 |
| 22 | 2 | 482 | 2023-01-11 | 3.00 |
| 23 | 3 | 483 | 2023-01-12 | 2.80 |
| 24 | 4 | 484 | 2023-01-13 | 3.20 |
| 25 | 5 | 485 | 2023-01-14 | 2.70 |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

6) A table with named Feedback with attributes:

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

Insert the records:

```
mysql> INSERT INTO Feedback (FeedbackID, StudentID, FeedbackDate, InstructorName, Feedback)
-> VALUES
-> (41, 481, '2023-01-01', 'John Doe', 'Great instructor, very helpful. '),
-> (42, 482, '2023-01-02', 'Jane Smith', 'Explained concepts very clearly. '),
-> (43, 483, '2023-01-03', 'Bob Johnson', 'Enthusiastic and knowledgeable. '),
-> (44, 484, '2023-01-04', 'Alice Williams', 'Made the class fun and engaging. '),
-> (45, 485, '2023-01-05', 'Charlie Brown', 'Very organized and approachable. ');
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

To display the data retrieved from the Feedback table:

```
mysql> Select *from Feedback;
```

FeedbackID	StudentID	FeedbackDate	InstructorName	Feedback
41	481	2023-01-01	John Doe	Great instructor, very helpful.
42	482	2023-01-02	Jane Smith	Explained concepts very clearly.
43	483	2023-01-03	Bob Johnson	Enthusiastic and knowledgeable.
44	484	2023-01-04	Alice Williams	Made the class fun and engaging.
45	485	2023-01-05	Charlie Brown	Very organized and approachable.

```
5 rows in set (0.00 sec)
```

ChatGPT Exercise

Generating a normalized table using ChatGPT as a continuation of our earlier

ChatGPT scenario.

Put the below statement in the message box and see what ChatGPT says:

I have created a database of emergency contact information. Kindly help me by providing normalized tables with understanding.

- 1)Employee (EmployeeID, FirstName, LastName, DateOfBirth, Gender, Phone)
- 2)EmergencyContact (ContactID, EmployeeID, ContactName, Relationship)
- 3)ContactDetails (ContactID, ContactType, ContactValue)

The Employee table stores information about each employee, including their name, date of birth, gender, and phone number. The primary key is EmployeeID, which uniquely identifies each employee.

The EmergencyContact table stores information about each employee's emergency contact, including their name and relationship to the employee. The primary key is ContactID, which is also a foreign key that references the ContactDetails table. The EmployeeID attribute is a foreign key that references the Employee table, indicating which employee this emergency contact belongs to.

The ContactDetails table stores the actual contact information for each emergency contact, including their email address and physical address. The ContactType attribute indicates whether the ContactValue is an email address or a physical address. The primary key is ContactID, which is also a foreign key that references the EmergencyContact table.

Here is an example of how you can create the ContactDetails table:

```
CREATE TABLE ContactDetails (  
    ContactID INT NOT NULL PRIMARY KEY,  
    ContactType VARCHAR(10) NOT NULL,  
    ContactValue VARCHAR(100) NOT NULL,  
    FOREIGN KEY (ContactID) REFERENCES EmergencyContact(ContactID)  
);
```

Sure, I'd be happy to help you normalize your database of emergency contact information. Here's an example of how you might structure the tables.

Employee Table

- EmployeeID (Primary Key, Auto-Increment)
- FirstName
- LastName
- DateOfBirth
- Gender
- Phone

EmergencyContact Table

- ContactID (Primary Key, Auto-Increment)
- EmployeeID (Foreign Key, References Employee(EmployeeID))
- ContactName
- Relationship
- ContactEmail
- ContactAddress

In this design, the Employee table stores information about each employee, and the EmergencyContact table stores information about each employee's emergency contact.

The EmployeeID field in the EmergencyContact table is a foreign key that references the Employee table, indicating which employee this emergency contact belongs to.

By normalizing the database in this way, we avoid redundancy and ensure that each piece of information is stored in only one place. This makes the database easier to maintain, update, and query.