

SQL LAB – 2

INSERT, SELECT

NAME: Keerthana K R

ID: AF0363623

QUESTIONS

Use the Database and table from Day 1 lab.
Insert 5 records in each table.

Database: studentmanagementsystem

Tables:

1. student
2. course
3. instructor
4. enrolment
5. score
6. feedback

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. Insert 5 records in student table.

Code:

```
mysql> insert into student
-> values(1, 'Alice', 'Johnson', '1995-03-18', 'Female', 'alice.johnson@gmail.com', 1234567890),
-> (2, 'Bob', 'Smith', '1998-07-22', 'Male', 'bob.smith@gmail.com', 2345678901),
-> (3, 'Carol', 'Taylor', '2000-11-11', 'Female', 'carol.taylor@yahoo.com', 3456789012),
-> (4, 'David', 'Brown', '1997-02-05', 'Male', 'david.brown@gmail.com', 4567890123),
-> (5, 'Eva', 'Davis', '1999-05-09', 'Female', 'eva.davis@yahoo.com', 5678901234);
Query OK, 5 rows affected (0.11 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

Output:

```
mysql> select * from student;
```

StudentID	FirstName	LastName	DateOfBirth	Gender	Email	Phone
1	Alice	Johnson	1995-03-18	Female	alice.johnson@gmail.com	1234567890
2	Bob	Smith	1998-07-22	Male	bob.smith@gmail.com	2345678901
3	Carol	Taylor	2000-11-11	Female	carol.taylor@yahoo.com	3456789012
4	David	Brown	1997-02-05	Male	david.brown@gmail.com	4567890123
5	Eva	Davis	1999-05-09	Female	eva.davis@yahoo.com	5678901234

5 rows in set (0.00 sec)

2. Insert 5 records in course table.

Code:

```
mysql> insert into course
-> values(101, 'Introduction to Computer Science', 3),
-> (102, 'Principles of Economics', 4),
-> (103, 'Calculus I', 4),
-> (104, 'World History', 3),
-> (105, 'Organic Chemistry', 4);
Query OK, 5 rows affected (0.13 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

Output:

```
mysql> select * from course;
+-----+-----+-----+
| CourseID | CourseTitle | Credits |
+-----+-----+-----+
| 101 | Introduction to Computer Science | 3 |
| 102 | Principles of Economics | 4 |
| 103 | Calculus I | 4 |
| 104 | World History | 3 |
| 105 | Organic Chemistry | 4 |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

3. Insert 5 records in instructor table.

Code:

```
mysql> insert into instructor
-> values
-> (1, 'Alice', 'Johnson','alice.johnson@gmail.com'),
-> (2, 'Bob', 'Smith','bob.smith@gmail.com'),
-> (3, 'Carol', 'Taylor','carol.taylor@yahoo.com'),
-> (4, 'David', 'Brown','david.brown@gmail.com'),
-> (5, 'Eva', 'Davis','eva.davis@yahoo.com');
Query OK, 5 rows affected (0.11 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

Output:

```
mysql> select * from instructor;
+-----+-----+-----+-----+
| InstructorID | FirstName | LastName | Email |
+-----+-----+-----+-----+
|          1 | Alice    | Johnson  | alice.johnson@gmail.com |
|          2 | Bob      | Smith    | bob.smith@gmail.com     |
|          3 | Carol    | Taylor   | carol.taylor@yahoo.com  |
|          4 | David    | Brown    | david.brown@gmail.com    |
|          5 | Eva      | Davis    | eva.davis@yahoo.com     |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

4. Insert 5 records in enrollment table.

Code:

```
mysql> insert into enrollment
-> values(1002,'2024-01-24',2,102,2),
-> (1003,'2024-01-01',3,103,3),
-> (1004,'2024-01-12',4,104,4),
-> (1005,'2024-01-21',5,105,5);
Query OK, 4 rows affected (0.11 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

Output:

```
mysql> select * from enrollment;
+-----+-----+-----+-----+-----+
| EnrollmentID | EnrollmentDate | StudentID | CourseID | InstructorID |
+-----+-----+-----+-----+-----+
| 1001 | 2024-01-24 | 1 | 101 | 1 |
| 1002 | 2024-01-24 | 2 | 102 | 2 |
| 1003 | 2024-01-01 | 3 | 103 | 3 |
| 1004 | 2024-01-12 | 4 | 104 | 4 |
| 1005 | 2024-01-21 | 5 | 105 | 5 |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

5. Insert 5 records in score table.

Code:

```
mysql> insert into score
-> values(111,'2024-03-12',5,101,1),
-> (112,'2024-03-12',4,102,2),
-> (113,'2024-03-12',4,103,3),
-> (114,'2024-03-12',3,104,4),
-> (115,'2024-03-12',5,105,5);
Query OK, 5 rows affected (0.14 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

Output;

```
mysql> select * from score;
+-----+-----+-----+-----+-----+
| ScoreID | DateOfExam | Creditobtained | CourseID | StudentID |
+-----+-----+-----+-----+-----+
|      111 | 2024-03-12 |           5 |      101 |          1 |
|      112 | 2024-03-12 |           4 |      102 |          2 |
|      113 | 2024-03-12 |           4 |      103 |          3 |
|      114 | 2024-03-12 |           3 |      104 |          4 |
|      115 | 2024-03-12 |           5 |      105 |          5 |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

6. Insert 5 records in feedback table.

Code:

```
mysql> insert into feedback
-> values(1, '2024-05-01', 'Dr. Alan', 'Very engaging lectures and thorough understanding of the material.', 1),
-> (2, '2024-05-02', 'Ms. Brenda', 'Needs more clarity in explaining complex topics.', 2),
-> (3, '2024-05-03', 'Prof. Charles', 'Excellent feedback on assignments.', 3),
-> (4, '2024-05-04', 'Dr. Diana', 'Classes are well-organized, but could include more interactive sessions.', 4),
-> (5, '2024-05-05', 'Mr. Eric', 'Helpful office hours and patient explanation of difficult concepts.', 5);
Query OK, 5 rows affected (0.14 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

Output:

```
mysql> select * from feedback;
+-----+-----+-----+-----+
| FeedbackID | Date       | InstructorName | Feedback |
+-----+-----+-----+-----+
| 1 | 2024-05-01 | Dr. Alan      | Very engaging lectures and thorough understanding of the material. |
| 2 | 2024-05-02 | Ms. Brenda    | Needs more clarity in explaining complex topics. |
| 3 | 2024-05-03 | Prof. Charles | Excellent feedback on assignments. |
| 4 | 2024-05-04 | Dr. Diana     | Classes are well-organized, but could include more interactive sessions. |
| 5 | 2024-05-05 | Mr. Eric      | Helpful office hours and patient explanation of difficult concepts. |
+-----+-----+-----+-----+
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. Unnormalized Structure

Start by looking at how data might be stored without normalization:

- EmployeeID
- FirstName
- LastName
- DateOfBirth
- Email
- Phone
- ContactName
- ContactRelationship
- ContactPhone
- ContactEmail
- ContactAddress

2. First Normal Form (1NF)

Split the data into two tables where each table has a primary key, and all attributes contain only atomic (indivisible) data:

Employee Table

- EmployeeID (Primary Key)
- FirstName
- LastName
- DateOfBirth
- Email
- Phone

EmergencyContact Table

- ContactID (Primary Key)
- EmployeeID (Foreign Key)
- ContactName
- Relationship
- Phone
- Email
- Address

In this structure:

- Each table has a primary key.
- Repeating groups are removed (multiple contacts are referenced by EmployeeID instead of repeating employee details).

3. Second Normal Form (2NF)

Achieved by ensuring that all non-key attributes are fully functional dependent on the primary key.

- Employee Table and EmergencyContact Table from 1NF are already in 2NF because all non-key fields are dependent on the primary key alone.

4. Third Normal Form (3NF)

A table is in 3NF if it is in 2NF and all the attributes in a table are only dependent on the primary key.

In the tables we've defined, there isn't any transitive dependency (where one non-key attribute depends on another non-key attribute). Thus, these tables also adhere to 3NF.

Normalized Tables Defined

Employee Table

Stores only employee-specific information, minimizing repeated data and isolating personal information.

```
CREATE TABLE Employee (  
    EmployeeID INT PRIMARY KEY,  
    FirstName VARCHAR(50),  
    LastName VARCHAR(50),  
    DateOfBirth DATE,  
    Email VARCHAR(100),  
    Phone VARCHAR(15)  
);
```

EmergencyContact Table

Stores emergency contact data linked to employees via a foreign key. Each entry can be related to one employee, but employees can have multiple contacts.

```
CREATE TABLE EmergencyContact (  
    ContactID INT PRIMARY KEY,  
    EmployeeID INT,  
    ContactName VARCHAR(50),  
    Relationship VARCHAR(50),  
    Phone VARCHAR(15),  
    Email VARCHAR(100),  
    Address VARCHAR(255),  
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)  
);
```