

***SQL MODULE***

***LAB – 2***

***BY***

***Sushritha V***

## Questions

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

and retrieve data from all tables and display.

Table1: Student1

```
mysql> describe student1;
```

Field	Type	Null	Key	Default	Extra
StudentID	int	NO	PRI	NULL	
FirstName	varchar(30)	NO		NULL	
LastName	varchar(30)	NO		NULL	
DateOfBirth	date	NO		NULL	
Gender	varchar(30)	YES		NULL	
Email	varchar(30)	YES		NULL	
Phone	int	YES		NULL	

7 rows in set (0.01 sec)

```
mysql> Insert into student1
-> values
-> (1, 'Sushritha', 'V', '1999-12-30', 'F', 'sushrithasushri30@gmail.com', 886152986),
-> (2, 'Pedda', 'Jagadeesh', '2003-07-06', 'M', 'peddajagadeesh06@gmail.com', 800874870),
-> (3, 'Krishna', 'Teja', '1999-12-23', 'M', 'Tejakrishna23@gmail.com', 990076583),
-> (4, 'Sai', 'Sujan', '2001-11-23', 'M', 'Saisujan23@gmail.com', 994563210),
-> (5, 'Punith', 'Samrat', '1999-06-19', 'M', 'Punithsamrat1906@gmail.com', 815106004);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

```
mysql> select * from student1;
```

StudentID	FirstName	LastName	DateOfBirth	Gender	Email	Phone
1	Sushritha	V	1999-12-30	F	sushrithasushri30@gmail.com	886152986
2	Pedda	Jagadeesh	2003-07-06	M	peddajagadeesh06@gmail.com	800874870
3	Krishna	Teja	1999-12-23	M	Tejakrishna23@gmail.com	990076583
4	Sai	Sujan	2001-11-23	M	Saisujan23@gmail.com	994563210
5	Punith	Samrat	1999-06-19	M	Punithsamrat1906@gmail.com	815106004

5 rows in set (0.00 sec)

Table2: Course

```
mysql> describe course;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| CourseID       | int           | NO   | PRI | NULL     | auto_increment |
| CourseTitle    | varchar(100)  | NO   |     | NULL     |                |
| Credits        | int           | YES  |     | NULL     |                |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

```
mysql> Insert into Course values
-> (1,'computer' ,6),
-> (2,'Civil', 5),
-> (3,'Electrical', 9),
-> (4,'Mechanical', 5),
-> (5,'Electronics', 7);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> select * from course;
+-----+-----+-----+
| CourseID | CourseTitle | Credits |
+-----+-----+-----+
|         1 | computer    |        6 |
|         2 | Civil       |        5 |
|         3 | Electrical  |        9 |
|         4 | Mechanical  |        5 |
|         5 | Electronics |        7 |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

Table3: Instructor

```
mysql> describe instructor;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| InstructorID   | int           | NO   | PRI | NULL     | auto_increment |
| FirstName      | varchar(50)   | NO   |     | NULL     |                |
| LastName       | varchar(50)   | NO   |     | NULL     |                |
| Email          | varchar(100)  | YES  | UNI | NULL     |                |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> Insert into instructor
-> values
-> (1, 'Sushritha', 'V', 'sushrithasushri30@gmail.com'),
-> (2, 'Pedda', 'Jagadeesh', 'peddajagadeesh06@gmail.com'),
-> (3, 'Krishna', 'Teja', 'Tejakrishna23@gmail.com'),
-> (4, 'Sai', 'Sujan', 'Saisujan23@gmail.com'),
-> (5, 'Punith', 'Samrat', 'Punithsamrat1906@gmail.com');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> select * from instructor;
+-----+-----+-----+-----+
| InstructorID | FirstName | LastName | Email |
+-----+-----+-----+-----+
| 1 | Sushritha | V | sushrithasushri30@gmail.com |
| 2 | Pedda | Jagadeesh | peddajagadeesh06@gmail.com |
| 3 | Krishna | Teja | Tejakrishna23@gmail.com |
| 4 | Sai | Sujan | Saisujan23@gmail.com |
| 5 | Punith | Samrat | Punithsamrat1906@gmail.com |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Table4: Enrollment

```
mysql> describe enrollment;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| EnrollmentID | int | NO | PRI | NULL | |
| EnrollmentDate | date | YES | | NULL | |
| StudentID | int | NO | MUL | NULL | |
| CourseID | int | NO | MUL | NULL | |
| InstructorID | int | NO | MUL | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

```
mysql> insert into enrollment
-> values(1,'2024-05-21',1,1,1);
Query OK, 1 row affected (0.01 sec)

mysql> insert into enrollment
-> values(2,'2024-05-21',2,2,2),
-> (3,'2024-05-22',3,3,3),
-> (4,'2024-05-23',4,4,4),
-> (5,'2024-05-24',5,5,5);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

```
mysql> select * from enrollment;
```

EnrollmentID	EnrollmentDate	StudentID	CourseID	InstructorID
1	2024-05-21	1	1	1
2	2024-05-21	2	2	2
3	2024-05-22	3	3	3
4	2024-05-23	4	4	4
5	2024-05-24	5	5	5

Table5: Score

```
mysql> describe score;
```

Field	Type	Null	Key	Default	Extra
ScoreID	int	NO	PRI	NULL	auto_increment
CourseID	int	NO	MUL	NULL	
StudentID	int	NO	MUL	NULL	
DateOfExam	date	YES		NULL	
CreditObtained	int	YES		NULL	

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

```
mysql> insert into score
-> values(1,1,1,'2024-05-21',2),
-> (2,2,2,'2024-05-21',4),
-> (3,3,3,'2024-05-21',6),
-> (4,4,4,'2024-05-21',8),
-> (5,5,5,'2024-05-21',10);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> select * from score;
```

ScoreID	CourseID	StudentID	DateOfExam	CreditObtained
1	1	1	2024-05-21	2
2	2	2	2024-05-21	4
3	3	3	2024-05-21	6
4	4	4	2024-05-21	8
5	5	5	2024-05-21	10

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

Table6: feedback

```
mysql> describe feedback;
```

Field	Type	Null	Key	Default	Extra
FeedbackID	int	NO	PRI	NULL	auto_increment
StudentID	int	NO	MUL	NULL	
Date	date	YES		NULL	
InstructorName	varchar(100)	YES		NULL	
Feedback	text	YES		NULL	

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

```
mysql> insert into feedback
-> values
-> (1, 1, '2024-05-22', 'Nagaraju J N', 'BESR'),
-> (2, 2, '2024-05-22', 'Nagaraju J N', 'OUTSTANDING'),
-> (3, 3, '2024-05-22', 'Nagaraju J N', 'GREAT'),
-> (4, 4, '2024-05-22', 'Nagaraju J N', 'IMPRESSIVE'),
-> (5, 5, '2024-05-22', 'Nagaraju J N', 'GOOD');
Query OK, 5 rows affected (0.02 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> select * from feedback;
```

FeedbackID	StudentID	Date	InstructorName	Feedback
1	1	2024-05-22	Nagaraju J N	BESR
2	2	2024-05-22	Nagaraju J N	OUTSTANDING
3	3	2024-05-22	Nagaraju J N	GREAT
4	4	2024-05-22	Nagaraju J N	IMPRESSIVE
5	5	2024-05-22	Nagaraju J N	GOOD

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

ChatGPT Question:

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.

I got following information from ChatGPT

Creating a normalized database for emergency contact information involves organizing the data into multiple related tables to reduce redundancy and improve data integrity. Here's an outline of what the normalized tables could look like along with explanations for each.

## Tables:

### 1. Employee Table:

- **Columns:**

- EmployeeID (INT, Primary Key): Unique identifier for each employee.
- FirstName (VARCHAR (50)): First name of the employee.
- LastName (VARCHAR (50)): Last name of the employee.
- Other relevant employee information (e.g., Department, Job Title).

- **Example Data:**

EmployeeID	FirstName	LastName	Department	Job Title
1	John	Smith	Marketing	Manager
2	Jane	Doe	Engineering	Developer
3	Michael	Lee	Finance	Analyst

### 2. Emergency Contact Table:

- **Columns:**

- EmergencyContactID (INT, Primary Key): Unique identifier for each emergency contact.
- FirstName (VARCHAR (50)): First name of the emergency contact.
- LastName (VARCHAR (50)): Last name of the emergency contact.
- RelationshipToEmployee (VARCHAR (50)): Relationship to the employee (e.g., Spouse, Parent, Child, Sibling, Friend).

--	--	--	--	--

Emergency ContactID	First Name	Last Name	Relationship To Employee	HashedPhoneNumber
1	Alice	Johnson	Spouse	(Hashed value of actual phone number)
2	David	Miller	Parent	(Hashed value of actual phone number)
3	Emily	Brown	Sibling	(Hashed value of actual phone number)

### Assumptions:

1. Each individual may have multiple emergency contacts.
2. Each contact might be connected to multiple individuals (e.g., in the case of family members).
3. Contacts may have various types of relationships with the individuals.
4. Address information might be necessary for contacts.

### First Normal Form (1NF)

1. **Individuals:** To store the personal details of the individuals.
2. **Contacts:** To store the details of the emergency contacts.
3. **Individual Contacts:** To store the relationships between individuals and their contacts.

### Tables and Structures

#### 1. Individuals

- **Table Name: Individuals**
- **Columns:**
  - **IndividualID** (Primary Key, unique identifier for each individual)
  - **FirstName**
  - **LastName**
  - **DateOfBirth**
  - **Phone Number**
  - **Email**
  - **Address**

#### 2. Contacts



- **Table Name: Contacts**
- **Columns:**
  - **ContactID** (Primary Key, unique identifier for each contact)
  - **FirstName**
  - **LastName**
  - **Phone Number**
  - **Email**
  - **Relationship Type** (e.g., parent, sibling, friend)
  - **Address**

### 3. Individual Contacts

- **Table Name: Individual Contacts**
- **Columns:**
  - **IndividualContactID** (Primary Key, unique identifier for each relationship entry)
  - **IndividualID** (Foreign Key, references **Individuals (IndividualID)**)
  - **ContactID** (Foreign Key, references **Contacts (ContactID)**)
  - **Relationship Type** (e.g., primary contact, secondary contact)
- **Example Data Entries**
- **Individuals Table**

Individual ID	FirstName	LastName	DateOf Birth	Phone Number	Email	Address
1	Sushi	V	1980-01-01	123-456-7890	<a href="mailto:sushisushri22@example.com">sushisushri22@example.com</a>	123 Elm St, City, ST

### Contacts Table

Contact ID	FirstName	LastName	Phone Number	Email	Relationship Type	Address
1	Punith	Samrat	234-567-8901	<a href="mailto:puni1906@example.com">puni1906@example.com</a>	Mother	456 Oak St, City, ST
2	Mike	Smith	345-678-9012	<a href="mailto:mike.smith@example.com">mike.smith@example.com</a>	Friend	789 Pine St,

						City, ST
--	--	--	--	--	--	-------------

### Individual Contacts Table

IndividualContactID	IndividualID	ContactID	Relationship Type
1	1	1	Primary
2	1	2	Secondary

### Explanation

- **Individuals Table:** Contains information about the individuals for whom the emergency contacts are being recorded.
- **Contacts Table:** Stores contact information of the emergency contacts.
- **Individual Contacts Table:** Manages the many-to-many relationship between individuals and their contacts, allowing everyone to have multiple contacts and each contact to be associated with multiple individuals if needed. It also specifies the nature of each relationship.

By following this normalization process, you reduce redundancy (e.g., not repeating contact details for everyone) and maintain data integrity, ensuring that updates to contact information need only be made in one place.