

SQL MODULE

LAB – 3

BY

Sushritha V

Questions

Task 1: Update the Student table with the following information:

Change the email to 'jane_Smith@example.com'

Where FirstName is 'Jane' and LastName is 'Smith';

Update the Instructor with the following information:

Change the email to 'rogerwhite@example.com'

Where FirstName of the instructor is 'Roger' and LastName is 'White';

```
mysql> update student1
-> set firstname='Jane', lastname='Smith', email='jane_Smith@example.com'
-> where studentid=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> select * from student1;
+-----+-----+-----+-----+-----+-----+-----+
| StudentID | FirstName | LastName | DateOfBirth | Gender | Email | Phone |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | Jane | Smith | 1999-12-30 | F | jane_Smith@example.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)
```

Instructor table:

```
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> update instructor
  -> set firstname='Roger', lastname='white',email='rogerwhite@example.com'
  -> where instructorid=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from instructor;
+-----+-----+-----+-----+
| InstructorID | FirstName | LastName | Email |
+-----+-----+-----+-----+
| 1 | Roger | white | rogerwhite@example.com |
| 2 | Pedda | Jagadeesh | peddajagadeesh06@gmail.com |
| 3 | Krishna | Teja | Tejakrishna23@gmail.com |
| 4 | Sai | Sujana | Saisujana23@gmail.com |
| 5 | Punith | Samrat | Punithsamrat1906@gmail.com |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Task2: Delete record from the student table on following condition:

Delete student/students records from the student table where last name is Smith.

: List the student whose first name starts with J.

```
mysql> select * from student1
  -> where firstname like 'j%';
+-----+-----+-----+-----+-----+-----+-----+
| StudentID | FirstName | LastName | DateOfBirth | Gender | Email | Phone |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | Jane | Smith | 1999-12-30 | F | jane_Smith@example.com | 886152986 |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Lab 2. Database Schema:

Consider a simple database with one table: Employee

Employee Table:

- Columns: emp_id (Primary Key), first_name, last_name, age, email

```
mysql> use student_management_system;
Database changed
mysql> create table employee
  -> (emp_id int primary key,
  -> firstname varchar(30) not null,
  -> lastname varchar(30) not null,
  -> age int not null,
  -> email varchar(30) not null unique);
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> describe employee;
```

Field	Type	Null	Key	Default	Extra
emp_id	int	NO	PRI	NULL	
firstname	varchar(30)	NO		NULL	
lastname	varchar(30)	NO		NULL	
age	int	NO		NULL	
email	varchar(30)	NO	UNI	NULL	

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

Task 1: Insert Data

Write an SQL INSERT statement to insert data into the Employee table

```
mysql> insert into employee
-> values
-> (1, 'sukanya', 'N', 20, 'sukanyammu21@gmail.com'),
-> (2, 'Monika', 'S', 23, 'monika19@gmail.com'),
-> (3, 'Dayana', 'S R', 23, 'puttidayana13@gmail.com');
Query OK, 3 rows affected (0.01 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

Task 2: Retrieving Data

Write an SQL SELECT statement to retrieve the first_name and last_name of all employees from the Employee table.

```
mysql> select * from employee;
```

emp_id	firstname	lastname	age	email
1	sukanya	N	20	sukanyammu21@gmail.com
2	Monika	S	23	monika19@gmail.com
3	Dayana	S R	23	puttidayana13@gmail.com

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

Task 3: Filtering Data

Write an SQL SELECT statement to retrieve the first_name, last_name, and age of employees who are older than 30 years.

```
mysql> select first_name, last_name from employee
       -> where age>30;
Empty set (0.00 sec)
```

Updating Data

Write an SQL UPDATE statement to increase the age of employees by 1 year for all employees older than 25.

```
mysql> update employee
       -> set age=age+1
       -> where age>25;
Query OK, 0 rows affected (0.01 sec)
Rows matched: 0  Changed: 0  Warnings: 0

mysql> ^C
mysql> select * from employee;
+-----+-----+-----+-----+-----+
| emp_id | first_name | last_name | age | email |
+-----+-----+-----+-----+-----+
|      1 | sukanya   | N        | 20 | sukanyammu21@gmail.com |
|      2 | Monika    | S        | 23 | monika19@gmail.com |
|      3 | Dayana    | S R      | 23 | puttidayana13@gmail.com |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

ChatGPT Exercise

Using ChatGPT generates SQL queries to update the Employee salary.

Scenario:

Due to a pricing adjustment, the company decided to increase the salary of all employees by 10%. Create an SQL update query to apply this change selectively to employees with a specific job title, say 'Manager'

sol: -

To selectively update the salary of all employees with the job title 'Manager' by 10%, you can use the following SQL `UPDATE` statement:

```
```SQL
```

```
UPDATE employees
```

```
SET salary = salary * 1.10
```

```
WHERE job title = 'Manager';
```

```
```
```

Explanation:

1. **UPDATE employees**: This specifies the table that you want to update.
2. **SET salary = salary * 1.10**: This updates the `salary` column by increasing its value by 10%.
3. **WHERE job title = 'Manager'**: This condition ensures that only the rows where the `job title` is 'Manager' are updated

Make sure to run this query in a transaction or test environment first to ensure it behaves as expected.

Conclusion:

In conclusion, creating a normalized database for emergency contact information involves structuring the data into related tables to minimize redundancy and maintain data integrity. By separating data into Individuals, Contacts, and Addresses tables, we ensure each piece of information is stored efficiently and can be easily maintained.

Additionally, to address specific business requirements, such as updating employee salaries selectively, we can use precise SQL queries. For example, increasing the salary of all employees with the job title 'Manager' by 10% can be achieved with a targeted UPDATE statement, ensuring that only the relevant records are modified.

By following these best practices in database normalization and SQL query formulation, we can create robust, maintainable, and scalable database systems.

