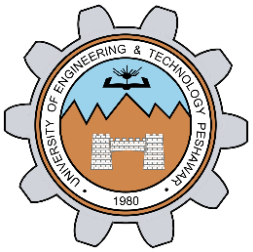
KEY MILESTONE 3: SQL IMPLEMENTATION

KAZ KITCHEN RESERVATION SYSTEM



Spring 2025

CSE-403L Database Management System Lab

Group Members:

KHIZRA HAROON (22PWCSE2121)
AREEJ (22PWCSE2206)
HAFIZA ZARLISHT NOOR (22PWCSE2112)

Class Section: **C**

Submitted to:

Engr. Sumayyea Salahuddin

June 2, 2025

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

KAZ KITCHEN RESERVATION SYSTEM

OVERVIEW

This project implements a restaurant reservation system with essential functionalities like booking reservations, handling table and slot conflicts, managing cancellations, and ensuring data integrity using triggers and constraints in MariaDB.

DATABASE AND TABLES

Database Name

```
MariaDB [(none)]> create database restaurant_db;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> use restaurant_db;
Database changed
```

Table: Customer

```
MariaDB [restaurant_db]> create table Customer (
    -> Customer_ID int auto_increment primary key,
    -> Name varchar(100),
    -> Contact varchar(20),
    -> Email varchar(100),
    -> Password varchar(100));
Query OK, 0 rows affected (0.039 sec)
```

```
MariaDB [restaurant_db]> describe customer;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| Customer_ID | int(11)      | NO   | PRI | NULL    | auto_increment |
| Name        | varchar(100) | YES  |     | NULL    |                |
| Contact     | varchar(20)  | YES  |     | NULL    |                |
| Email       | varchar(100) | YES  |     | NULL    |                |
| Password    | varchar(100) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

Table: TableInfo

```
MariaDB [restaurant_db]> create table TableInfo (
    -> Table_ID int auto_increment primary key,
    -> Capacity int);
Query OK, 0 rows affected (0.062 sec)
```

```
MariaDB [restaurant_db]> describe TableInfo;
+----------+---------+------+-----+---------+----------------+
| Field    | Type    | Null | Key | Default | Extra          |
+----------+---------+------+-----+---------+----------------+
| Table_ID | int(11) | NO   | PRI | NULL    | auto_increment |
| Capacity | int(11) | YES  |     | NULL    |                |
+----------+---------+------+-----+---------+----------------+
2 rows in set (0.046 sec)
```

Table: Time_Slot

```
MariaDB [restaurant_db]> create table Time_Slot (
    -> Slot_ID int auto_increment primary key,
    -> Start_Time TIME,
    -> End_Time TIME);
Query OK, 0 rows affected (0.051 sec)
```

```
MariaDB [restaurant_db]> describe Time_Slot;
+------------+---------+------+-----+---------+----------------+
| Field      | Type    | Null | Key | Default | Extra          |
+------------+---------+------+-----+---------+----------------+
| Slot_ID    | int(11) | NO   | PRI | NULL    | auto_increment |
| Start_Time | time    | YES  |     | NULL    |                |
| End_Time   | time    | YES  |     | NULL    |                |
+------------+---------+------+-----+---------+----------------+
```

Table: Service

```
MariaDB [restaurant_db]> create table Service (
    -> Service_ID int auto_increment primary key,
    -> Name varchar(100),
    -> Description TEXT,
    -> Cost Decimal(10,2));
Query OK, 0 rows affected (0.054 sec)
```

```
MariaDB [restaurant_db]> describe Service;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| Service_ID | int(11)      | NO   | PRI | NULL    | auto_increment |
| Name       | varchar(100) | YES  |     | NULL    |                |
| Description| text         | YES  |     | NULL    |                |
| Cost       | decimal(10,2)| YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
4 rows in set (0.040 sec)
```

Table: Reservation

```
MariaDB [restaurant_db]> create table Reservation (
    -> Reservation_ID int auto_increment primary key,
    -> Customer_ID int,
    -> Table_ID int,
    -> Slot_ID int,
    -> Service_ID int,
    -> ReservationDate DATE,
    -> ReservationStatus ENUM('Confirmed', 'Cancelled'),
    -> NumberOfGuests int,
    -> FOREIGN KEY (Customer_ID) references Customer(Customer_ID),
    -> FOREIGN KEY (Table_ID) references TableInfo(Table_ID),
    -> FOREIGN KEY (Slot_ID) references Time_Slot(Slot_ID),
    -> FOREIGN KEY (Service_ID) references Service(Service_ID));
Query OK, 0 rows affected (0.093 sec)
```

```
MariaDB [restaurant_db]> describe Reservation;
+------------------+------------------------------+------+-----+---------+----------------+
| Field            | Type                         | Null | Key | Default | Extra          |
+------------------+------------------------------+------+-----+---------+----------------+
| Reservation_ID   | int(11)                      | NO   | PRI | NULL    | auto_increment |
| Customer_ID      | int(11)                      | YES  | MUL | NULL    |                |
| Table_ID         | int(11)                      | YES  | MUL | NULL    |                |
| Slot_ID          | int(11)                      | YES  | MUL | NULL    |                |
| Service_ID       | int(11)                      | YES  | MUL | NULL    |                |
| ReservationDate  | date                         | NO   |     | NULL    |                |
| ReservationStatus | enum('Confirmed','Cancelled') | YES  |     | NULL    |                |
| NumberOfGuests   | int(11)                      | YES  |     | NULL    |                |
+------------------+------------------------------+------+-----+---------+----------------+
8 rows in set (0.048 sec)
```

Table: Cancellation

```
MariaDB [restaurant_db]> create table Cancellation (
    -> Cancellation_ID int auto_increment primary key,
    -> Reservation_ID int UNIQUE,
    -> FOREIGN KEY (Reservation_ID) REFERENCES Reservation(Reservation_ID));
Query OK, 0 rows affected (0.061 sec)
```

```
MariaDB [restaurant_db]> describe Cancellation;
+-----------------+---------+------+-----+---------+----------------+
| Field           | Type    | Null | Key | Default | Extra          |
+-----------------+---------+------+-----+---------+----------------+
| Cancellation_ID | int(11) | NO   | PRI | NULL    | auto_increment |
| Reservation_ID  | int(11) | YES  | UNI | NULL    |                |
+-----------------+---------+------+-----+---------+----------------+
2 rows in set (0.046 sec)
```

SAMPLE DATA INSERTION

```
MariaDB [restaurant_db]> insert into Customer (Name, Contact, Email, Password) values
    -> ('Khizra Haroon', '03419084519', 'khizra.haroon3@gmail.com','1422'),
    -> ('Areej', '03018977431', 'areej123@gmail.com','abc123'),
    -> ('Zarlisht Noor', '03028867400', 'zarlishtkhan@gmail.com','try456'),
    -> ('Ali', '03312330541', 'alykhn@gmail.com','123456'),
    -> ('Bruno', '03318911221', 'bruno@gmail.com','bruno123');
Query OK, 5 rows affected (0.019 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
MariaDB [restaurant_db]> insert into TableInfo (Capacity) values
    -> (2),
    -> (3),
    -> (6),
    -> (8),
    -> (10);
Query OK, 5 rows affected (0.018 sec)
Records: 5  Duplicates: 0  Warnings: 0

MariaDB [restaurant_db]> insert into Time_Slot (Start_Time, End_Time) values
    -> ('18:00:00','20:00:00'),
    -> ('20:00:00','21:00:00'),
    -> ('21:00:00','22:00:00'),
    -> ('22:00:00','24:00:00'),
    -> ('17:00:00','18:00:00');
Query OK, 5 rows affected (0.016 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
MariaDB [restaurant_db]> insert into Service (Name, Cost) values
    -> ('Birthday Setup', 5000),
    -> ('Anniversary Decor', 5000),
    -> ('Valentine Setup', 4500),
    -> ('Corporate Setup', 5500),
    -> ('Candlelight Dinner', 6000);
Query OK, 5 rows affected (0.017 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
MariaDB [restaurant_db]> insert into Reservation(Customer_ID, Table_ID, Slot_ID, NumberOfGuests, ReservationStatus, Service_ID) values
    -> (1,2,1,2, 'Confirmed', 1),
    -> (2,3,2,4, 'Confirmed', 2),
    -> (3,1,3,2, 'Cancelled', NULL),
    -> (4,4,4,6, 'Confirmed', 3),
    -> (5,5,5,8, 'Confirmed', 5);
Query OK, 5 rows affected (0.043 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
MariaDB [restaurant_db]> ALTER TABLE Reservation MODIFY ReservationDate DATE;
Query OK, 0 rows affected (0.019 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [restaurant_db]> UPDATE Reservation SET ReservationDate = '2025-06-01' WHERE Reservation_ID = 1;
Query OK, 1 row affected (0.018 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [restaurant_db]> UPDATE Reservation SET ReservationDate = '2025-06-02' WHERE Reservation_ID = 2;
Query OK, 1 row affected (0.004 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [restaurant_db]> UPDATE Reservation SET ReservationDate = '2025-06-03' WHERE Reservation_ID = 3;
Query OK, 1 row affected (0.005 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [restaurant_db]> UPDATE Reservation SET ReservationDate = '2025-06-04' WHERE Reservation_ID = 4;
Query OK, 1 row affected (0.005 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [restaurant_db]> UPDATE Reservation SET ReservationDate = '2025-06-05' WHERE Reservation_ID = 5;
Query OK, 1 row affected (0.004 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

TRIGGERS USED

Trigger 1: Prevent Double Booking

This checks if a confirmed reservation already exists for the same table and time on the same date. If yes, it throws an error and blocks the insertion.

```
MariaDB [restaurant_db]> DELIMITER $$
MariaDB [restaurant_db]> create trigger prevent_double_booking
    -> BEFORE INSERT ON reservation
    -> FOR EACH ROW
    -> BEGIN
    ->     IF EXISTS (
    ->         SELECT 1 FROM reservation
    ->         WHERE Table_ID = NEW.Table_ID
    ->           AND Slot_ID = NEW.Slot_ID
    ->           AND ReservationDate = NEW.ReservationDate
    ->           AND ReservationStatus = 'Confirmed'
    ->     ) THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'This table is already booked for the same time slot and date.';
    -> END IF;
    -> END$$
Query OK, 0 rows affected (0.056 sec)
```

Testing:

```
MariaDB [restaurant_db]> insert into Reservation (Customer_ID, Table_ID, Slot_ID, ReservationDate, ReservationStatus, NumberOfGuests)
    -> values (2,3,2, '2025-06-02' , 'Confirmed', 11);
ERROR 1644 (45000): This table is already booked for the selected time slot on this date.
```

Trigger 2: Auto-Update Status on Cancellation

When a new row is added to Cancellation, this updates the related reservation status.

```
MariaDB [restaurant_db]> DELIMTER $$
    -> CREATE TRIGGER update_status_on_cancellation
    -> AFTER INSERT ON Cancellation
    -> FOR EACH ROW
    -> UPDATE Reservation
    -> SET ReservationStatus = 'Cancelled'
    -> WHERE Reservation_ID = NEW.Reservation_ID;
    -> WHERE Reservation_ID = NEW.Reservation_ID;
    -> END$$
Query OK, 0 rows affected (0.037 sec)
```

Testing:

```
MariaDB [restaurant_db]> insert into Cancellation (Reservation_I
    -> VALUES (1);
Query OK, 1 row affected (0.021 sec)
```

```
MariaDB [restaurant_db]> select * from Reservation;
+----------------+-------------+----------+---------+------------+-----------------+-------------------+----------------+
| Reservation_ID | Customer_ID | Table_ID | Slot_ID | Service_ID | ReservationDate | ReservationStatus | NumberOfGuests |
+----------------+-------------+----------+---------+------------+-----------------+-------------------+----------------+
|              1 |           1 |        2 |       1 |          1 | 2025-06-01      | Cancelled         |              2 |
|              2 |           2 |        3 |       2 |          2 | 2025-06-02      | Confirmed         |              4 |
|              3 |           3 |        1 |       3 |       NULL | 2025-06-03      | Cancelled         |              2 |
|              4 |           4 |        4 |       4 |          3 | 2025-06-04      | Confirmed         |              6 |
|              5 |           5 |        5 |       5 |          5 | 2025-06-05      | Confirmed         |              8 |
|              7 |           1 |        1 |       1 |       NULL | 2025-06-06      | Cancelled         |              5 |
+----------------+-------------+----------+---------+------------+-----------------+-------------------+----------------+
```

Trigger 3: To Prevent Cancellations After the Time Slot

According to Business Rule: “A CANCELLATION must occur before the reservation time.” So, this trigger enforces timely cancellations.

```
MariaDB [restaurant_db]> CREATE TRIGGER prevent_late_cancellation
    -> BEFORE INSERT ON Cancellation
    -> FOR EACH ROW
    -> BEGIN
    -> DECLARE reserved_time DATETIME;
    -> SELECT ts.Start_Time
    ->     INTO reserved_time
    ->     FROM Reservation r
    ->     JOIN Time_Slot ts ON r.Slot_ID = ts.Slot_ID
    ->     WHERE r.Reservation_ID = NEW.Reservation_ID;
    ->
    ->     IF NOW() > reserved_time THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Cannot cancel a reservation after the reserved time.';
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.022 sec)
```

Testing:

```
MariaDB [restaurant_db]> INSERT INTO Payment (Reservation_ID, Amount, Payment_Date)
    -> VALUES (9, 5000, NOW());
Query OK, 1 row affected, 1 warning (0.016 sec)

MariaDB [restaurant_db]> INSERT INTO Cancellation (Reservation_ID, Refund_amount)
    -> VALUES (9, 5000);
ERROR 1644 (45000): Cannot cancel a reservation after the reserved time.
```

VIEWING CONFIRMED UPCOMING RESERVATIONS

```
MariaDB [restaurant_db]> CREATE VIEW Confirmed_Reservations AS
    -> SELECT r.Reservation_ID, c.Name AS Customer, t.Capacity, ts.Start_Time, s.Name AS Service
    -> FROM Reservation r
    -> JOIN Customer c ON r.Customer_ID = c.Customer_ID
    -> JOIN TableInfo t ON r.Table_ID = t.Table_ID
    -> JOIN Time_Slot ts ON r.Slot_ID = ts.Slot_ID
    -> LEFT JOIN Service s ON r.Service_ID = s.Service_ID
    -> WHERE r.ReservationStatus = 'Confirmed';
Query OK, 0 rows affected (0.025 sec)
```

ON DELETE CASCADE FOREIGN KEY CONSTRAINTS

This ensures related rows get deleted when a parent is deleted (like Cancellation when Reservation is deleted).

```
MariaDB [restaurant_db]> ALTER TABLE Cancellation
    -> ADD CONSTRAINT fk_res_cancel
    -> FOREIGN KEY (Reservation_ID) REFERENCES Reservation(Reservation_ID)
    -> ON DELETE CASCADE;
Query OK, 2 rows affected (0.182 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

SAMPLE QUERIES

1. **SHOW TABLES;**
   Shows all the tables in the database.

```
MariaDB [restaurant_db]> show tables;
+--------------------------+
| Tables_in_restaurant_db |
+--------------------------+
| cancellation             |
| customer                 |
| reservation              |
| service                  |
| tableinfo                |
| time_slot                |
+--------------------------+
6 rows in set (0.001 sec)
```

2. **SELECT * FROM Customer;**
   Fetches all rows and all columns from the Customer table.

```
MariaDB [restaurant_db]> select * from Customer;
+-------------+--------------+-------------+-------------------------+----------+
| Customer_ID | Name         | Contact     | Email                   | Password |
+-------------+--------------+-------------+-------------------------+----------+
|           1 | Khizra Haroon | 03419084519 | khizra.haroon3@gmail.com | 1422     |
|           2 | Areej        | 03018977431 | areej123@gmail.com       | abc123   |
|           3 | Zarlisht Noor | 03028867400 | zarlishtkhan@gmail.com   | try456   |
|           4 | Ali          | 03312330541 | alykhn@gmail.com         | 123456   |
|           5 | Bruno        | 03318911221 | bruno@gmail.com          | bruno123 |
+-------------+--------------+-------------+-------------------------+----------+
5 rows in set (0.001 sec)
```

3. **SELECT * FROM Reservation;**
   Fetches all rows and all columns from the Reservation table.

```
MariaDB [restaurant_db]> select * from reservation;
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
| Reservation_ID | Customer_ID | Table_ID | Slot_ID | Service_ID | ReservationDate | ReservationStatus | NumberOfGuests |
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
|              1 |           1 |        2 |       1 |          1 | 2025-06-01      | Cancelled        |              2 |
|              2 |           2 |        3 |       2 |          2 | 2025-06-02      | Confirmed        |              4 |
|              3 |           3 |        1 |       3 |       NULL | 2025-06-03      | Cancelled        |              2 |
|              4 |           4 |        4 |       4 |          3 | 2025-06-04      | Confirmed        |              6 |
|              5 |           5 |        5 |       5 |          5 | 2025-06-05      | Confirmed        |              8 |
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
5 rows in set (0.000 sec)
```

4. **SELECT * FROM TableInfo;**
   Fetches all rows and all columns from the TableInfo table.

```
MariaDB [restaurant_db]> select * from TableInfo;
+----------+----------+
| Table_ID | Capacity |
+----------+----------+
|        1 |        2 |
|        2 |        3 |
|        3 |        6 |
|        4 |        8 |
|        5 |       10 |
+----------+----------+
5 rows in set (0.001 sec)
```

5. **SELECT * FROM Time_Slot;**
   Fetches all rows and all columns from the Time_Slot table.

```
MariaDB [restaurant_db]> select * from Time_Slot;
+---------+------------+----------+
| Slot_ID | Start_Time | End_Time |
+---------+------------+----------+
|       1 | 18:00:00   | 20:00:00 |
|       2 | 20:00:00   | 21:00:00 |
|       3 | 21:00:00   | 22:00:00 |
|       4 | 22:00:00   | 24:00:00 |
|       5 | 17:00:00   | 18:00:00 |
|       6 | 17:00:00   | 18:00:00 |
+---------+------------+----------+
```

6. **SELECT * FROM Service;**
   Fetches all rows and all columns from the Service table.

```
MariaDB [restaurant_db]> select * from Service;
+-----------+------------------+-------------+---------+
| Service_ID | Name             | Description | Cost    |
+-----------+------------------+-------------+---------+
|         1 | Birthday Setup   | NULL        | 5000.00 |
|         2 | Anniversary Decor | NULL        | 5000.00 |
|         3 | Valentine Setup  | NULL        | 4500.00 |
|         4 | Corporate Setup  | NULL        | 5500.00 |
|         5 | Candlelight Dinner | NULL      | 6000.00 |
+-----------+------------------+-------------+---------+
5 rows in set (0.000 sec)
```

7. **SELECT * FROM Cancellation;**
   Fetches all rows and all columns from the Cancellation table.

```
MariaDB [restaurant_db]> select * from cancellation;
+-----------------+----------------+
| Cancellation_ID | Reservation_ID |
+-----------------+----------------+
|               2 |              1 |
|               1 |              3 |
+-----------------+----------------+
2 rows in set (0.000 sec)
```

8. View All Confirmed Reservations:

```
MariaDB [restaurant_db]> SELECT * FROM Reservation WHERE ReservationStatus = 'Confirmed';
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
| Reservation_ID | Customer_ID | Table_ID | Slot_ID | Service_ID | ReservationDate | ReservationStatus | NumberOfGuests |
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
|              2 |           2 |        3 |       2 |          2 | 2025-06-02      | Confirmed         |              4 |
|              4 |           4 |        4 |       4 |          3 | 2025-06-04      | Confirmed         |              6 |
|              5 |           5 |        5 |       5 |          5 | 2025-06-05      | Confirmed         |              8 |
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
3 rows in set (0.001 sec)
```

9. Show Upcoming Reservations:

```
MariaDB [restaurant_db]> SELECT *
    -> FROM Reservation
    -> WHERE ReservationStatus = 'Confirmed'
    ->   AND ReservationDate >= CURDATE()
    -> ORDER BY ReservationDate, Slot_ID;
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
| Reservation_ID | Customer_ID | Table_ID | Slot_ID | Service_ID | ReservationDate | ReservationStatus | NumberOfGuests |
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
|              2 |           2 |        3 |       2 |          2 | 2025-06-02      | Confirmed         |              4 |
|              4 |           4 |        4 |       4 |          3 | 2025-06-04      | Confirmed         |              6 |
|              5 |           5 |        5 |       5 |          5 | 2025-06-05      | Confirmed         |              8 |
+----------------+-------------+----------+---------+------------+-----------------+------------------+----------------+
3 rows in set (0.001 sec)
```

CONCLUSION

The SQL implementation phase successfully established a well-structured database for the Restaurant Reservation System. It includes the creation of essential tables along with clearly defined relationships and constraints to ensure data integrity.

Advanced functionalities were implemented using triggers, including mechanisms to:

- Prevent double booking of tables for the same time and date,
- Automatically update the reservation status upon cancellation,
- Enforce rules for timely cancellations.

The database supports efficient querying of reservation status, upcoming bookings, and customer details, ensuring smooth back-end support for the overall system. This solid foundation ensures consistency, reliability, and future scalability of the restaurant reservation application.