Restaurant Reservations Database Report

Github: https://github.com/BHUAY/Final-Project-Restaurant-Portal

This report outlines the development process of the "restaurant_reservations" database, designed to manage customer information, reservations, and dining preferences for a restaurant. It covers the project's requirements identification, design approach, implementation strategies, and the results, along with findings from the initial deployment.

1. Project Requirements

The "restaurant_reservations" database project was initiated to address the need for a comprehensive system that includes the following entities:

- Customers: Managing customer data, including names and contact information.
- Reservations: Recording reservation details such as reservation time, number of guests, and special requests.
- DiningPreferences: Storing customers' dining preferences, including favorite tables and dietary restrictions.

2. Design Approach

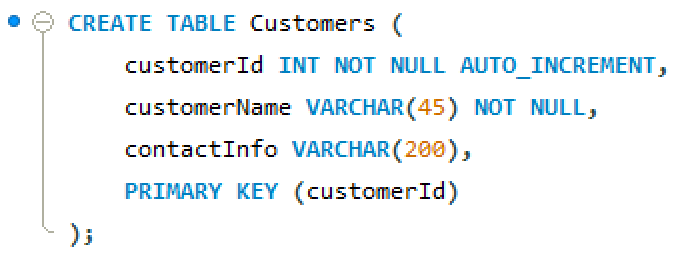
The design strategy involved creating normalized tables for each major entity to minimize redundancy and ensure data integrity. Primary and foreign keys were used to establish and enforce relationships between the entities, supporting a robust relational model for efficient data retrieval and management.

3. Implementation Strategies

The implementation phase was executed using SQL scripts with the following strategies:

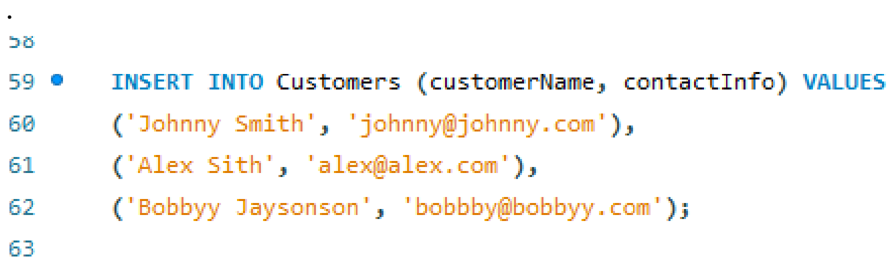
- Table Creation: Utilizing CREATE TABLE statements to define the structure of each entity, ensuring proper data types and constraints.

- Data Integrity: Assigning primary keys for unique identification and foreign keys for relational integrity among tables.



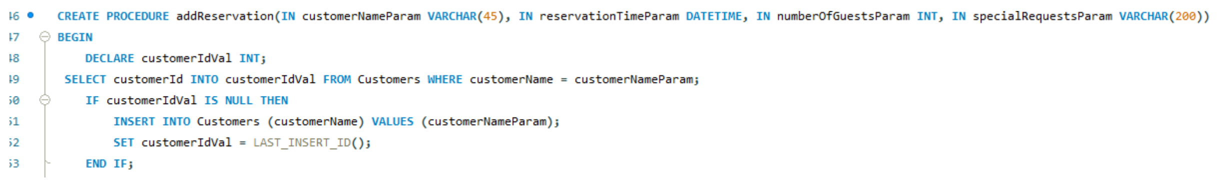
```sql
CREATE TABLE Customers (
    customerId INT NOT NULL AUTO_INCREMENT,
    customerName VARCHAR(45) NOT NULL,
    contactInfo VARCHAR(200),
    PRIMARY KEY (customerId)
);
```

- Initial Data Population: Employing INSERT INTO statements to populate the tables with sample data representing realistic scenarios



```sql
INSERT INTO Customers (customerName, contactInfo) VALUES
('Johnny Smith', 'johnny@johnny.com'),
('Alex Sith', 'alex@alex.com'),
('Bobbyy Jaysonson', 'bobbby@bobbyy.com');
```

-
- Data Retrieval: Implementing SELECT statements with JOIN clauses to retrieve and integrate data across multiple tables.



```sql
CREATE PROCEDURE addReservation(IN customerNameParam VARCHAR(45), IN reservationTimeParam DATETIME, IN numberOfGuestsParam INT, IN specialRequestsParam VARCHAR(200))
BEGIN
    DECLARE customerIdVal INT;
 SELECT customerId INTO customerIdVal FROM Customers WHERE customerName = customerNameParam;
    IF customerIdVal IS NULL THEN
        INSERT INTO Customers (customerName) VALUES (customerNameParam);
        SET customerIdVal = LAST_INSERT_ID();
    END IF;
```

4. Results and Findings

The successful execution of the SQL script resulted in a fully functional "restaurant_reservations" database capable of managing customer information, reservations, and dining preferences effectively. Key observations include:

- The relational structure effectively represents the relationships between different entities.
- Data integrity constraints ensure consistency and accuracy of stored data.
- Normalized tables reduce data redundancy and improve data retrieval efficiency.

Areas for improvement include identifying and addressing data relationship gaps and optimizing financial and administrative processes based on database insights.

5 . Conclusion

The "restaurant_reservations" database project demonstrates a comprehensive approach to managing restaurant reservations and customer preferences. Its success sets a foundation for efficient restaurant operations and customer service, offering insights for continuous improvement and adaptation to evolving business needs.