RESTAURANT MANAGEMENT SYSTEM

PROJECT BY Sparsh Makharia and Kaavya Iyer

1. INTRODUCTION:

In today's fast-paced world, restaurants rely on efficient and organized systems to manage their daily operations. A well-designed Restaurant Management System (RMS) can significantly enhance the overall dining experience, from tracking reservations to handling customer orders and billing. This project aims to develop a Database Management Systemfor a restaurant, enabling efficient management of crucial restaurant operations, including customer information, menu items, table reservations, order processing, billing, and more.

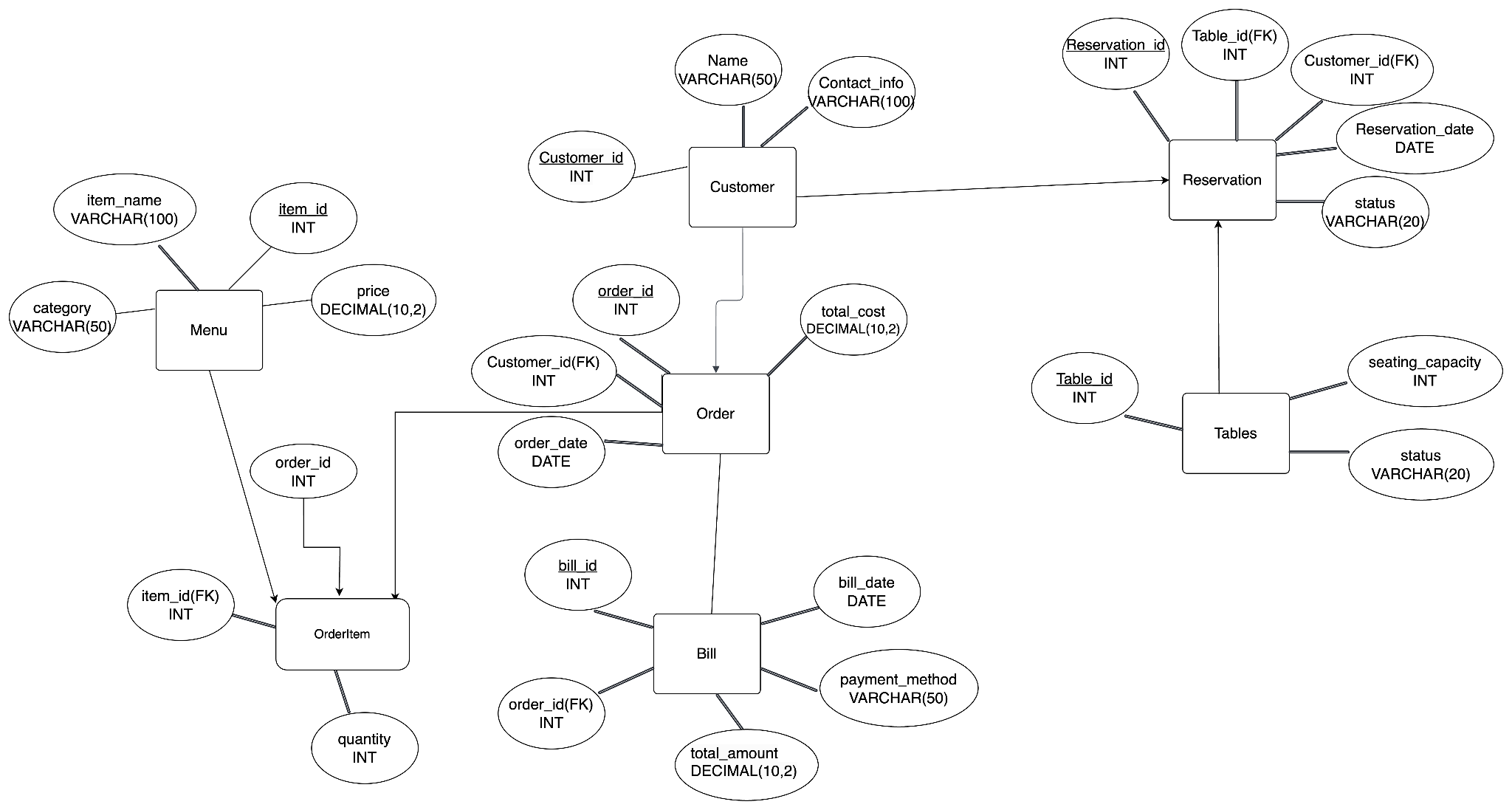
1. OBJECTIVES OF THE SYSTEM:

The key objectives of the DBMS-based restaurant management system include:

* **Efficiency**: Streamlining restaurant operations, reducing manual work and human errors.
* **Data Integrity**: Ensuring that the data stored in the system remains accurate, consistent, and secure.
* **Real-time Access**: Giving restaurant staff real-time access to data such as table availability, menu items, and customer orders.
* **Scalability**: Designing the system to be flexible and scalable to accommodate changes in the restaurant's size or menu.

1. IMPLEMENTATION:

* E-R MODEL



RELATIONS BETWEEN EACH TABLE:

## 1. Customer Table

## One-to-Many with Order: A single customer can place multiple orders, but each order is linked to only one customer. This relationship is established through the customer\_id foreign key in the Order table.

## One-to-Many with Reservation: A single customer can make multiple reservations, but each is associated with only one customer. This relationship is established through the customer\_id foreign key in the Reservation table.

## 2. MenuItem Table

## Many-to-Many with Order (via OrderItem): Each menu item can be included in multiple orders, and each order can contain multiple menu items. This many-to-many relationship is resolved through the OrderItem table,

## 

## 

## 3. Tables Table

## One-to-Many with Reservation: A single table can have multiple reservations over time, but each reservation pertains to only one specific table. This relationship is established through the table\_id foreign key in the Reservation table.

## 4. Order Table

## One-to-Many with OrderItem: Each order can contain multiple order items, but each item belongs to only one order. This relationship is established through the order\_id foreign key in the OrderItem table.

## One-to-One with Bill: Each order generates one bill, and each bill corresponds to only one order. This relationship is established through the order\_id foreign key in the Bill table.

## 5. OrderItem Table

## Many-to-One with Order: Each OrderItem is associated with a single order, but an order can have multiple items.

## Many-to-One with MenuItem: Each order item corresponds to one specific menu item, but a menu item can be included in multiple order items.

## 6. Bill Table

## One-to-One with Order: Each bill is linked to a single order through the order\_id, ensuring that each order has a corresponding bill.

## 7. Reservation Table

## Many-to-One with Customer: Each reservation is made by one customer, but a customer can have multiple reservations.

## Many-to-One with Tables: Each reservation is for one specific table, but a table can be reserved multiple times.

\

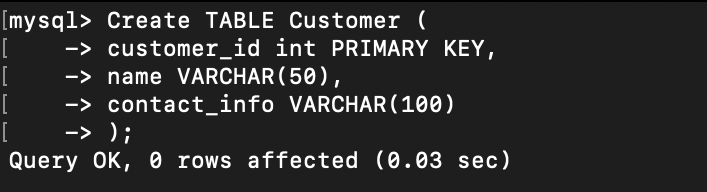
**Database Tables and the information in them:**

We have made 7 tables and added inputs to each table, keeping a primary key and a foreign key to connect the tables. The database consists of the following key tables:

## 1. Customer

This table stores information about the restaurant's customers.

* customer\_id (PK): Unique identifier for each customer
* customer\_name: Name of the customer
* Contact\_info: phone number





## 

## 

## 

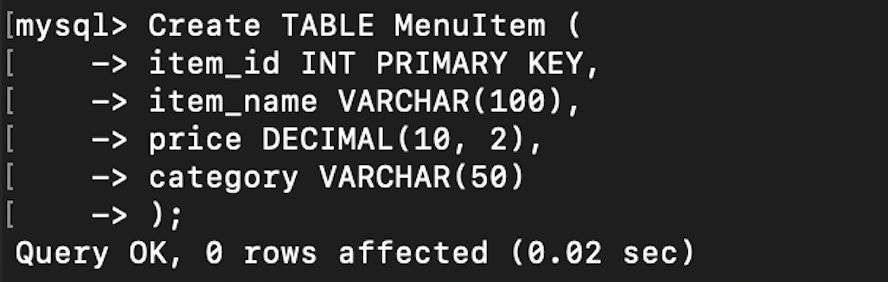
## 

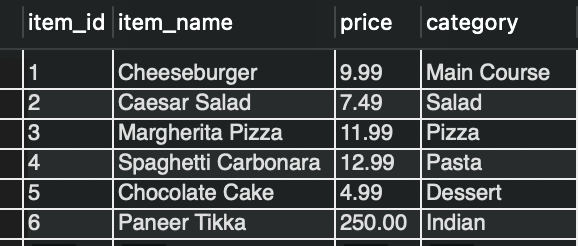
## 

## 2. MenuItem

This table contains details about the food and beverage items available in the restaurant.

* item\_id (PK): Unique identifier for each menu item.
* item\_name: Name of the food item.
* category: Category of the item (e.g., appetizer, main course).
* price: Price of the item.





## 

## 

## 

## 

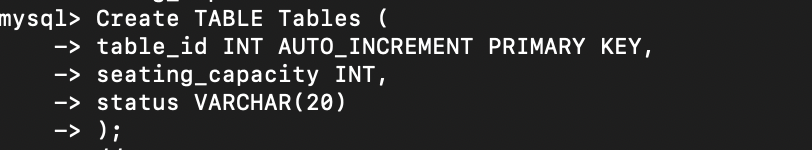
## 

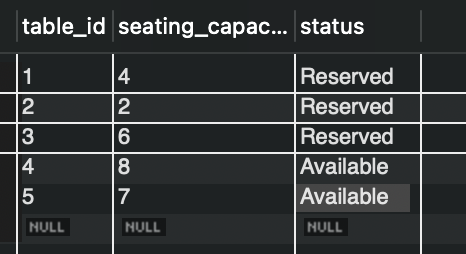
## 

## 3. Tables

This table manages information regarding the tables available in the restaurant.

* table\_id (PK): Unique identifier for each table.
* seating\_capacity: Number of people that can be seated at the table.
* status: Current status (e.g., available, reserved)

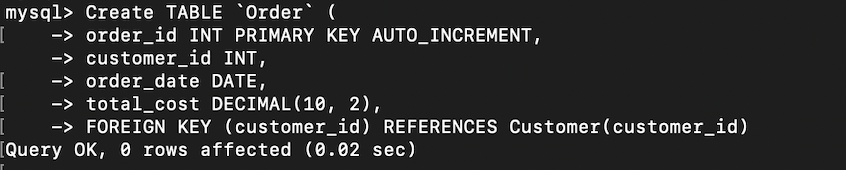


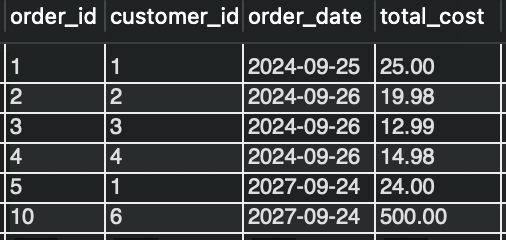


4. Order

This table records all orders placed by customers.

* order\_id (PK): Unique identifier for each order.
* customer\_id (FK): Foreign key linking to the Customer table.
* totalcost: Total cost of the items ordered.
* order\_date: Date and time when the order was placed.





## 

## 

## 

## 

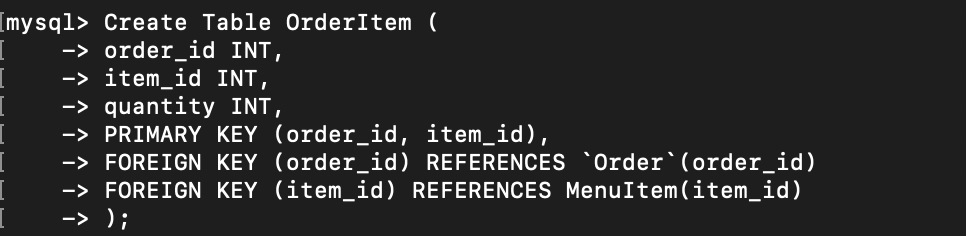
## 

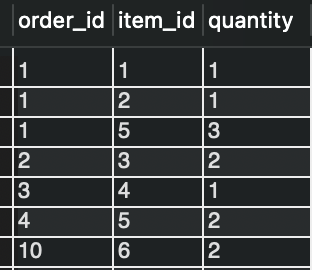
## 

## 5. OrderItem

This table links customers' orders with specific menu items.

* order\_id (FK): Foreign key linking to the Order table.
* item\_id (FK): Foreign key linking to the MenuItem table.
* quantity: Number of units ordered.

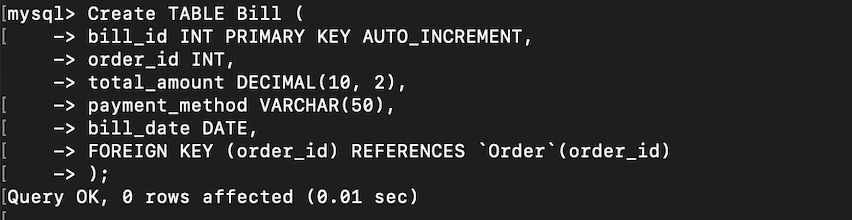


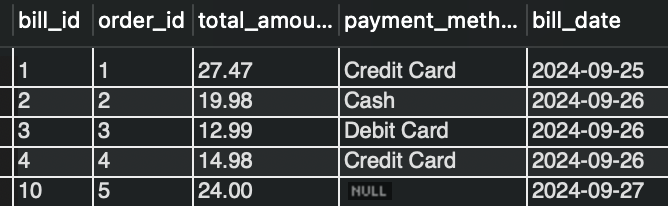


## 6. Bill

This table manages billing information related to orders.

* bill\_id (PK): Unique identifier for each bill.
* order\_id (FK): Foreign key linking to the Order table.
* total\_payment: Total amount due for payment.
* payment\_method: Method of payment used by the customer (e.g., cash, credit card, mobile payment).
* bill\_date: Date and time when the bill was generated.



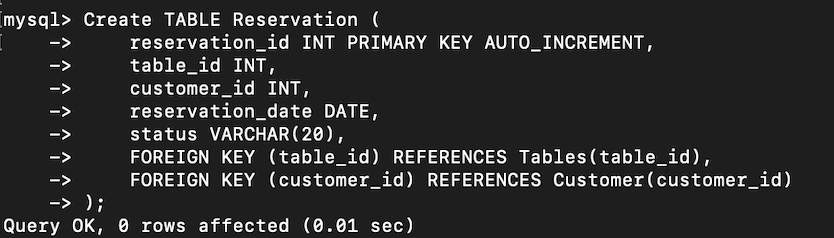


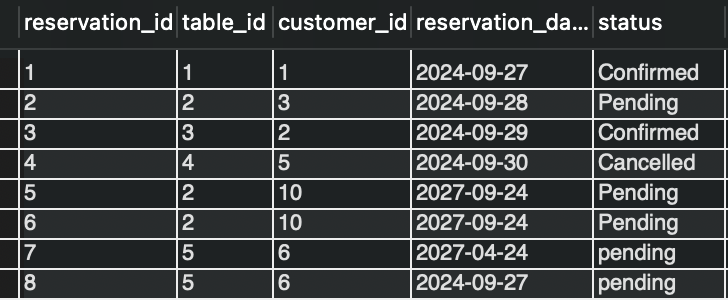
## 

## 7. Reservation

This table handles reservations made by customers in advance.

* reservation\_id (PK): Unique identifier for each reservation.
* table\_id(FK): Foreign key linking the reservation to a specific table reserved.
* customer\_id (FK): Foreign key linking to the Customer table.
* table\_id (FK): Foreign key linking to the Tables table.
* reservation\_date: Date and time when the reservation is made.
* status: Current status of the reservation (e.g., pending, confirmed, cancelled)





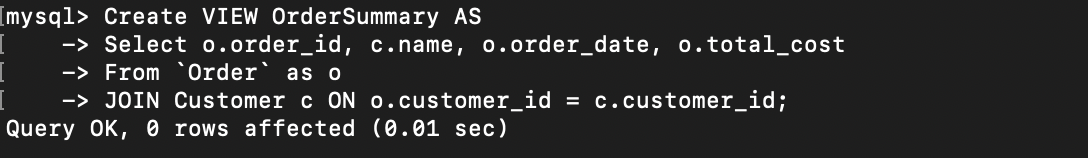
1. REQUIREMENTS:

* VIEWS, PROCEDURES AND TRIGGERS MADE:

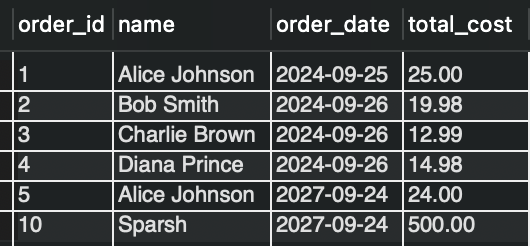
VIEWS

* View #1:

It displays customer orders with menu items. The OrderSummary view significantly enhances restaurant operations by providing a streamlined and efficient way to access key order information.

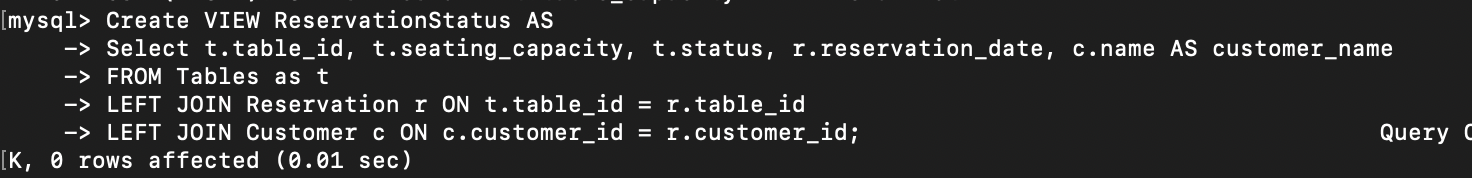


Output-

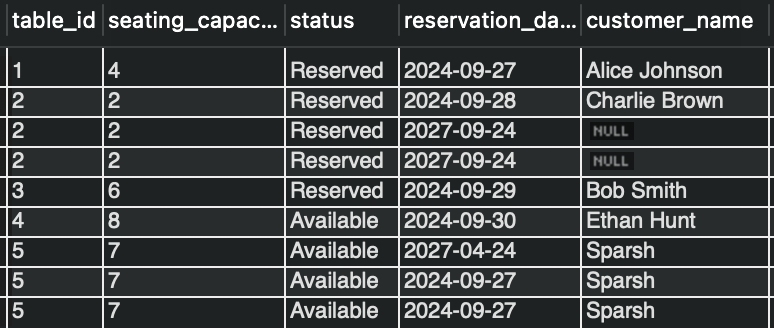


* View #2:

The Reservation Status view helps restaurants by showing all reservation details in one place, making it easier for staff to manage tables and serve customers quickly. This leads to better customer service and helps the restaurant run more smoothly.



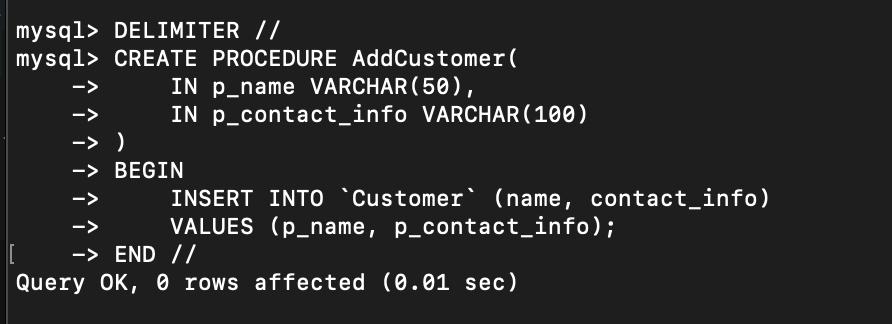
Output:



PROCEDURES

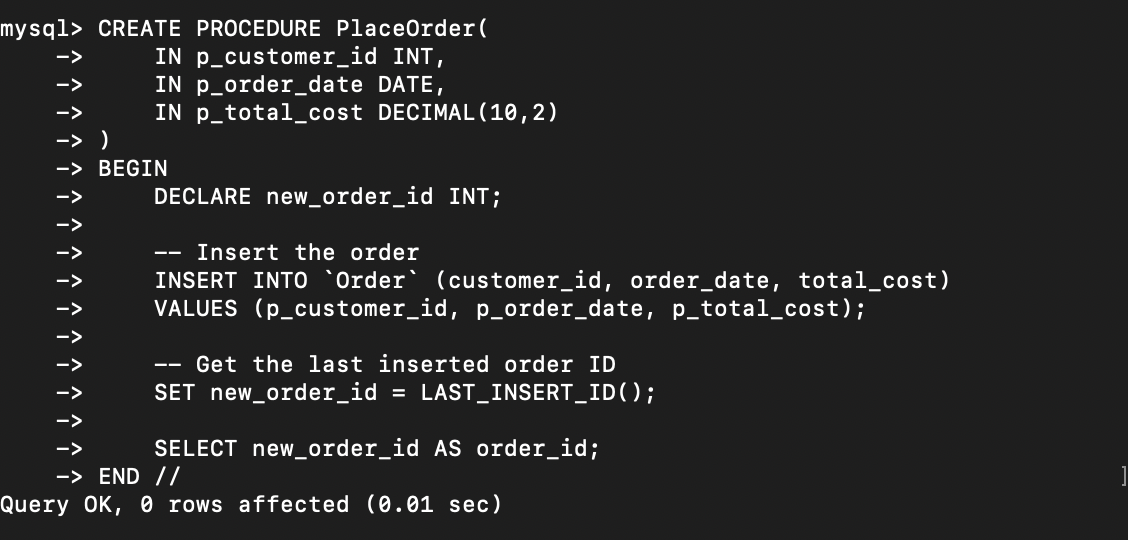
## 1. Procedure to Add a New Customer

This procedure allows you to add a new customer to the Customer table.



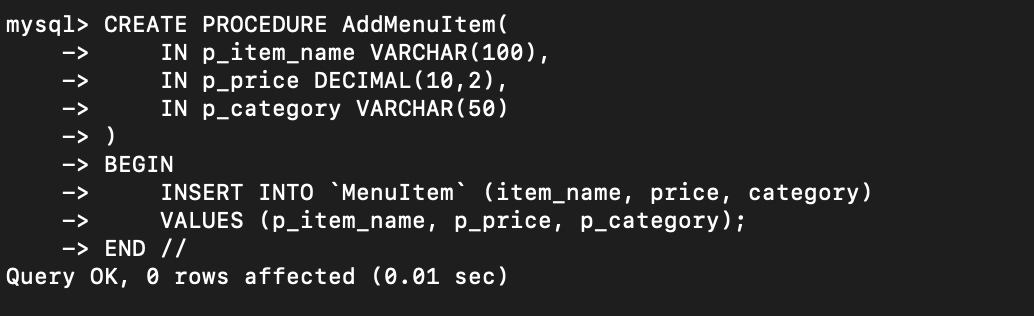
## 2. Procedure to Place an Order

This procedure allows you to place a new order for a customer.



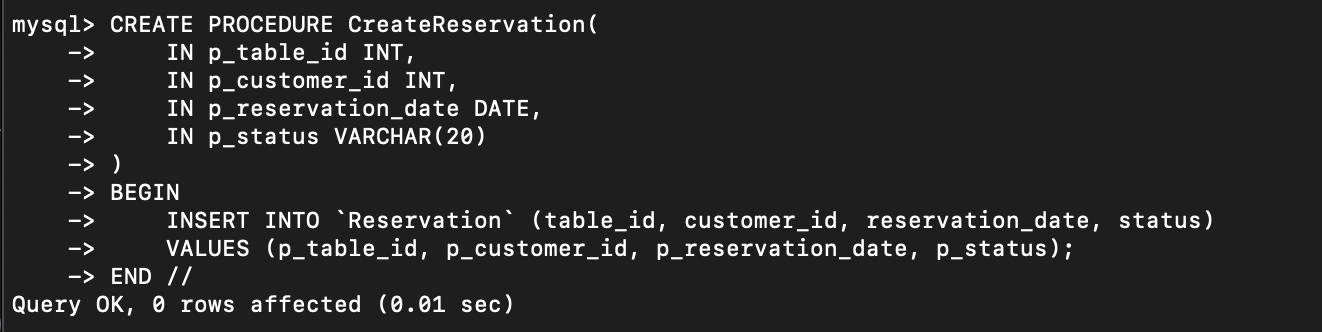
## 3. Procedure to Add Menu Item

This procedure allows you to add a new menu item.



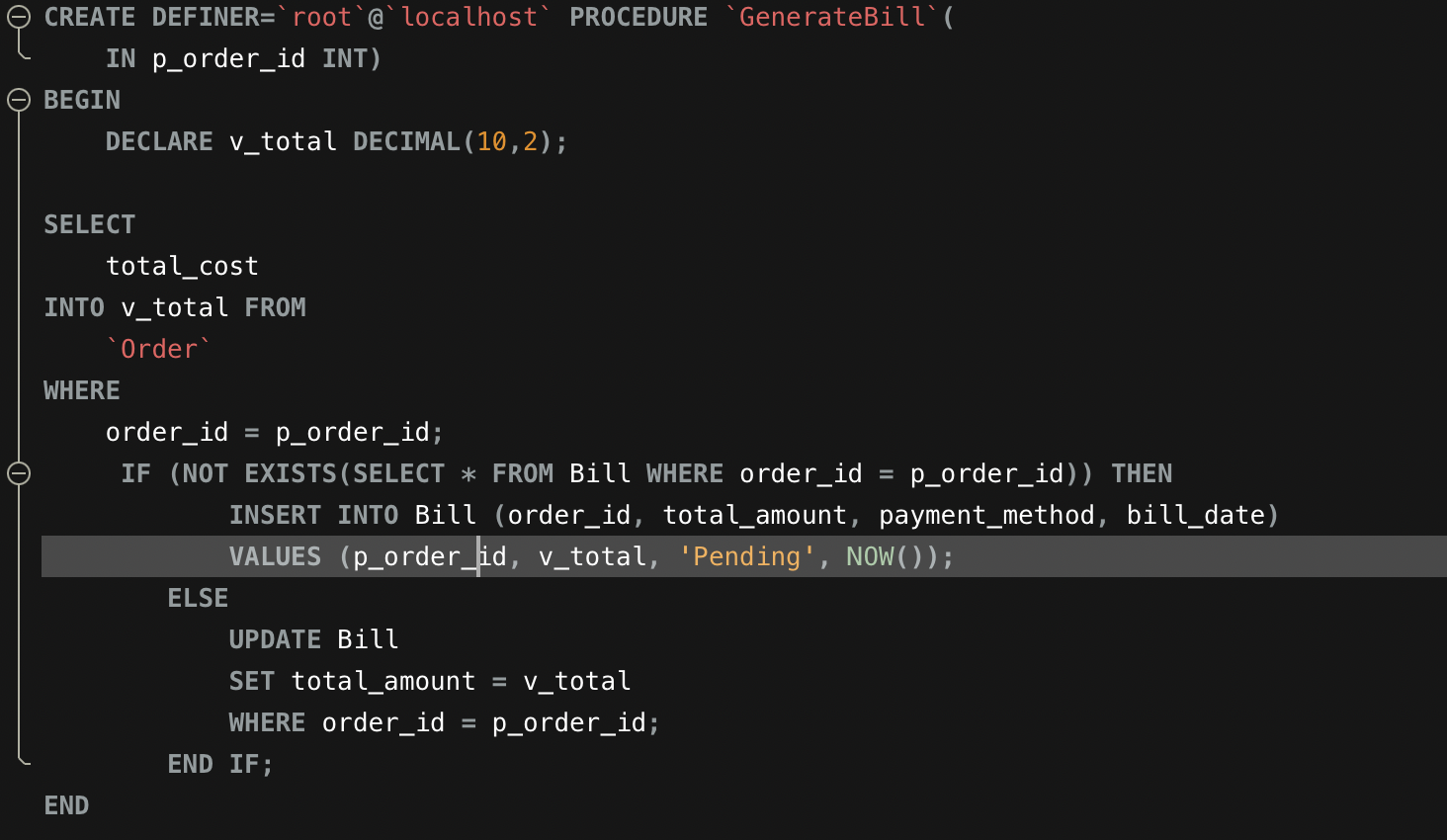
4. Procedure to Create a Reservation

This procedure allows you to create a reservation for a customer.



5. Procedure to Generate Bill

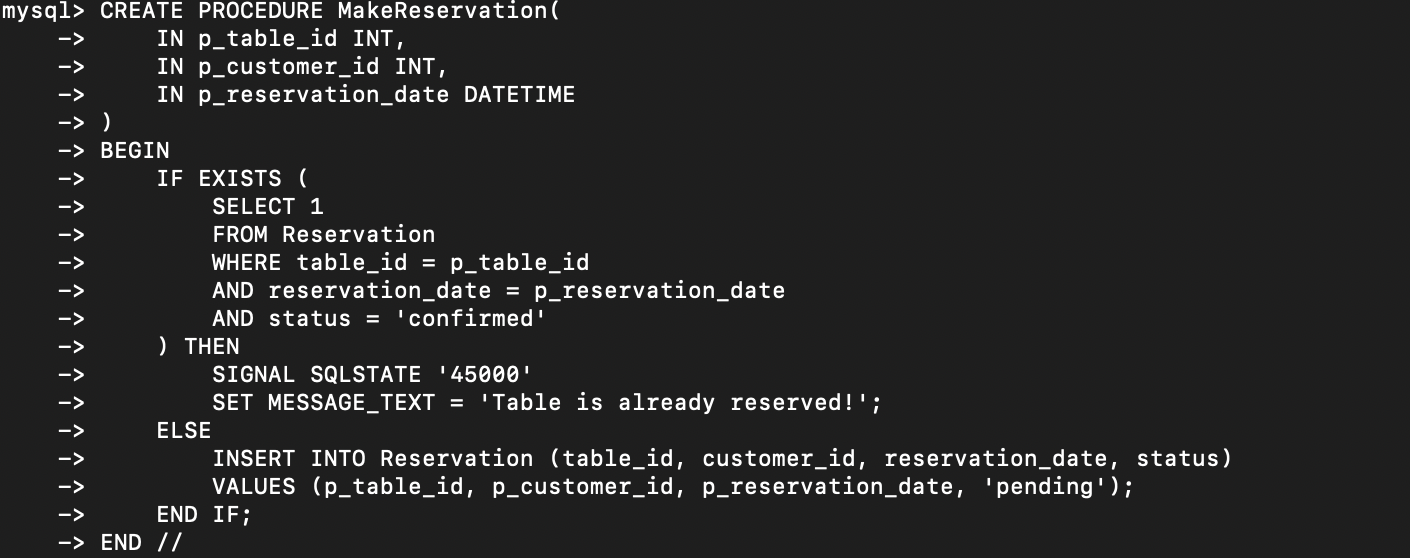
This procedure generates a bill based on an order, calculating the total amount due and inserting it into the Bill table



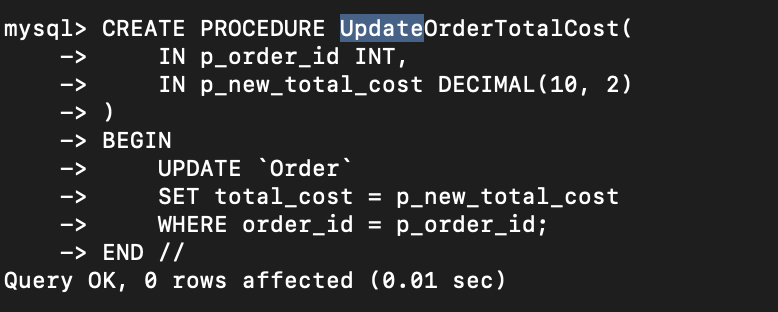
#### 

#### 6. Procedure to Make a Reservation

This procedure inserts a reservation and checks if the table is available at the requested time.



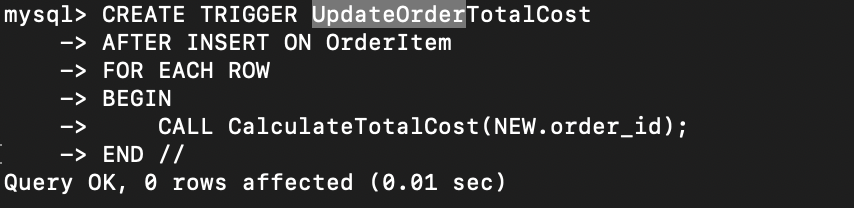
7. Procedure to update order Total Cost



TRIGGERS

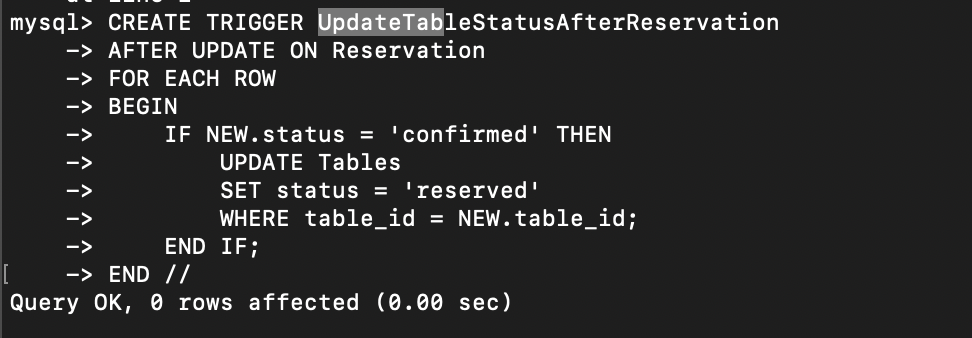
1. A Trigger to Update Total Cost

This trigger ensures that whenever a new item is added to an order, the total cost is updated in the Order table.



#### Trigger to Update Table Status When Reservation is Confirmed

When a reservation is confirmed, the status of the table should be updated to "reserved."



## 

## 

## E. Observations

A Database Management System significantly improves restaurant management by organizing data into structured tables, ensuring integrity and consistency. It enables comprehensive reporting and analytics, allowing restaurants to make informed decisions based on sales trends and customer preferences. By streamlining order processing and integrating with inventory management, a DBMS minimizes errors and optimizes stock levels, ultimately improving efficiency. Additionally, it enhances the customer experience through features like online ordering and reservation management. With its scalability and flexibility, a DBMS supports the growth of the restaurant while adapting to changing needs, making it an essential tool for modern restaurant operations.There are still bottlenecks in the tables and reservation system of the Restaurant Management System, as some tasks require manual intervention. For instance, table availability may still need to be manually confirmed at certain times, and managing complex reservations during peak hours could slow down operations. Automating these tasks and fully integrating the system with real-time updates on table status would further streamline the process and enhance overall efficiency.