|  |  |
| --- | --- |
| **HACKATHON LEVEL 2** | |
| **TITLE** | **ONLINE FOOD ORDERING SYSEM** |
| **NAME** | **M.R. DEVADHARSHINI** |
| **REGISTER NUMBER** | **C2S31753** |
| **INSTITUTION** | **MARY MATHA COLLEGE OF ARTS AND SCIENCE** |
| **DEPARTMENT** | **BSC INFORMATION TECHNOLOGY** |
| **DATE OF SUBMISSION** | **04 - 04 -2025** |

**1. Problem Statement:**

Restaurants need an efficient system to manage online food orders, track inventory, and handle customer transactions. Manual management leads to inefficiencies, errors in inventory tracking, and customer service delays. The current food ordering platforms face challenges such as inconsistent inventory updates, complex order tracking, and lack of automated reporting.

**Key Challenges:**

1. **Order Management Issues:** Manual order processing leads to errors and delays in customer deliveries.
2. **Inventory Mismatch:** Restaurants often struggle with inaccurate stock tracking, leading to incorrect order fulfillment.
3. **Payment Handling:** Managing various payment methods securely can be challenging without a proper system.
4. **Customer Engagement:** Lack of personalized recommendations and order history tracking reduces customer retention.
5. **Scalability Issues:** Traditional systems may not efficiently handle a high volume of concurrent orders.

**Significance of the System:**

* **Automation:** The proposed system automates order placement, inventory updates, and payment processing.
* **Real-time Tracking:** Customers can track their orders, and restaurant owners can monitor inventory levels in real time.
* **Data-Driven Decisions:** Sales analytics help businesses identify top-selling items and optimize inventory accordingly.
* **Enhanced Security:** Secure payment processing and customer data protection ensure a safe transaction environment.
* **User-friendly Interface:** A simplified and efficient interface for both customers and restaurant management improves the overall user experience.

The **Online Food Ordering System Database** addresses these challenges by integrating menu management, order processing, payment handling, and inventory tracking seamlessly, ensuring a smooth and efficient ordering process for both customers and restaurant operators.

**2. Database Design & Implementation**

**2.1 Database Creation & Tables:**

**Create database:**

|  |
| --- |
| CREATE DATABASE OnlineFoodOrdering;  USE OnlineFoodOrdering; |

**Create Customers Table:**

|  |
| --- |
| CREATE TABLE Customers (  customer\_id INT AUTO\_INCREMENT PRIMARY KEY,  name VARCHAR(100) NOT NULL,  email VARCHAR(100) UNIQUE NOT NULL,  phone VARCHAR(15) UNIQUE NOT NULL,  address TEXT,  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  ); |

**Create Menu Table:**

|  |
| --- |
| CREATE TABLE Menu (  item\_id INT AUTO\_INCREMENT PRIMARY KEY,  name VARCHAR(100) NOT NULL,  category VARCHAR(50) NOT NULL,  price DECIMAL(10,2) NOT NULL,  available BOOLEAN DEFAULT TRUE  ); |

**Create Order Table:**

|  |
| --- |
| CREATE TABLE Orders (  order\_id INT AUTO\_INCREMENT PRIMARY KEY,  customer\_id INT,  order\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  status ENUM('Pending', 'Processing', 'Delivered', 'Cancelled'),  total\_amount DECIMAL(10,2),  FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id)  ); |

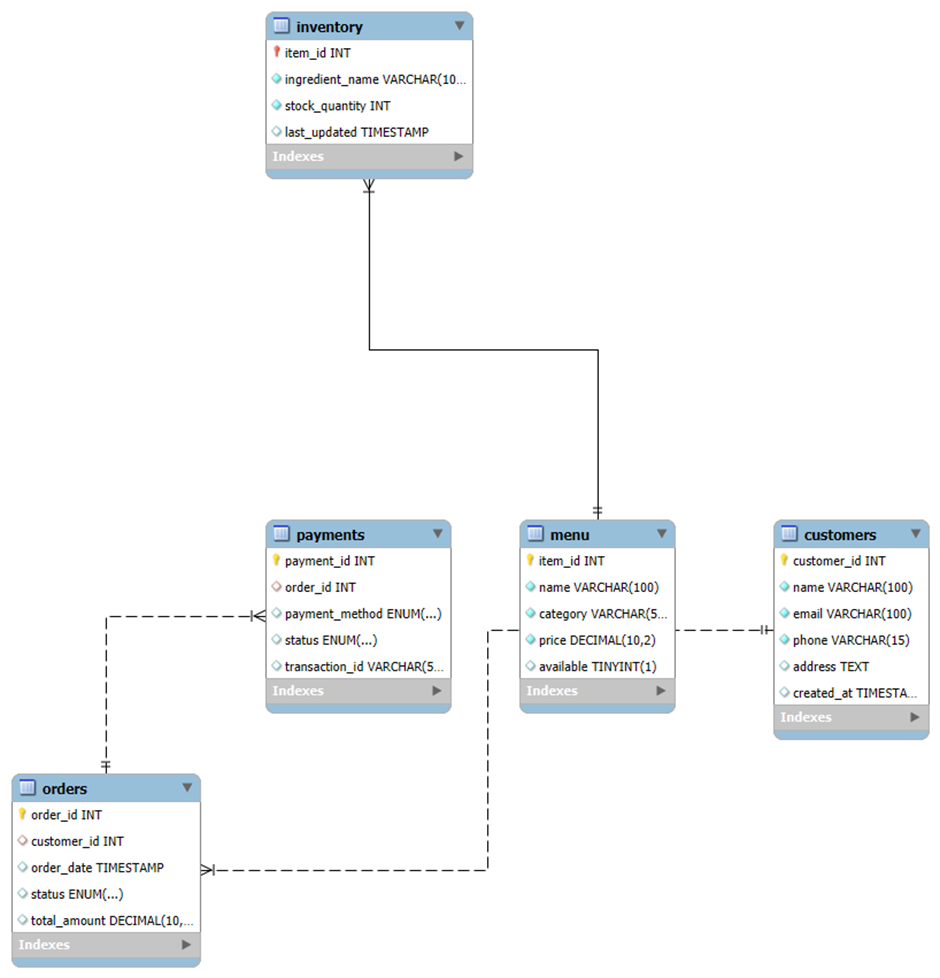
**Create Payments Table:**

|  |
| --- |
| CREATE TABLE Payments (  payment\_id INT AUTO\_INCREMENT PRIMARY KEY,  order\_id INT,  payment\_method ENUM('Credit Card', 'Debit Card', 'UPI', 'Cash'),  status ENUM('Pending', 'Completed', 'Failed'),  transaction\_id VARCHAR(50) UNIQUE,  FOREIGN KEY (order\_id) REFERENCES Orders(order\_id)  ); |

**Create Inventory** **Table:**

|  |
| --- |
| CREATE TABLE Inventory (  item\_id INT PRIMARY KEY,  ingredient\_name VARCHAR(100) NOT NULL,  stock\_quantity INT NOT NULL,  last\_updated TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  FOREIGN KEY (item\_id) REFERENCES Menu(item\_id)  ); |

**2.2 ER Diagram (Reverse Engineered)**



**3. Queries for Data Management**

**3.1 Insert Sample Data**

**Insert Customers Table:**

|  |
| --- |
| INSERT INTO Customers (name, email, phone, address) VALUES  ('Lingeshwaran M', 'lingeshwaran@gmail.com', '9876543210', '12, Anna Nagar, Chennai'),  ('Priya Ramesh', 'priya.ramesh@gmail.com', '8765432109', '45, MG Road, Bangalore'),  ('Arun Kumar', 'arun.kumar@gmail.com', '9988776655', '78, Connaught Place, Delhi'),  ('Sneha Iyer', 'sneha.iyer@gmail.com', '9123456789', '23, T Nagar, Chennai'),  ('Vikas Singh', 'vikas.singh@gmail.com', '9345678901', '55, Park Street, Kolkata'),  ('Ananya Joshi', 'ananya.joshi@gmail.com', '9567890123', '67, FC Road, Pune'); |

**Insert Menu Table:**

|  |
| --- |
| INSERT INTO Menu (name, category, price, available) VALUES  ('Paneer Butter Masala', 'North Indian', 249.99, TRUE),  ('Dosa', 'South Indian', 99.99, TRUE),  ('Chicken Biryani', 'Mughlai', 299.99, TRUE),  ('Pav Bhaji', 'Street Food', 129.99, TRUE),  ('Masala Dosa', 'South Indian', 119.99, TRUE),  ('Chole Bhature', 'North Indian', 149.99, TRUE); |

**Insert Orders Table:**

|  |
| --- |
| INSERT INTO Orders (customer\_id, order\_date, status, total\_amount) VALUES  (1, NOW(), 'Pending', 249.99),  (2, NOW(), 'Processing', 399.99),  (3, NOW(), 'Delivered', 199.99),  (4, NOW(), 'Cancelled', 129.99),  (5, NOW(), 'Pending', 299.99),  (6, NOW(), 'Delivered', 149.99); |

**Insert Payments Table:**

|  |
| --- |
| INSERT INTO Payments (order\_id, payment\_method, status, transaction\_id) VALUES  (1, 'UPI', 'Completed', 'TXN123456'),  (2, 'Credit Card', 'Pending', 'TXN123457'),  (3, 'Debit Card', 'Completed', 'TXN123458'),  (4, 'Cash', 'Completed', 'TXN123459'),  (5, 'UPI', 'Failed', 'TXN123460'),  (6, 'Credit Card', 'Completed', 'TXN123461'); |

**Insert Inventory Table:**

|  |
| --- |
| INSERT INTO Inventory (item\_id, ingredient\_name, stock\_quantity) VALUES  (1, 'Paneer & Spices', 50),  (2, 'Rice & Masala', 100),  (3, 'Chicken & Rice', 75),  (4, 'Vegetables & Bread', 120),  (5, 'Lentils & Rice', 80),  (6, 'Chickpeas & Flour', 90); |

**3.2 Retrieval Queries**

**Fetch all pending orders with customer details:**

|  |
| --- |
| SELECT  Orders.order\_id,  Customers.name AS customer\_name,  Customers.phone,  Orders.order\_date,  Orders.total\_amount  FROM  Orders  JOIN  Customers ON Orders.customer\_id = Customers.customer\_id  WHERE  Orders.status = 'Pending'; |

**List all available menu items by category:**

|  |
| --- |
| SELECT  category,  name,  price  FROM  Menu  WHERE  available = TRUE  ORDER BY  category; |

**Retrieve payment status for all orders:**

|  |
| --- |
| SELECT  Payments.payment\_id,  Orders.order\_id,  Customers.name AS customer\_name,  Payments.payment\_method,  Payments.status  FROM  Payments  JOIN  Orders ON Payments.order\_id = Orders.order\_id  JOIN  Customers ON Orders.customer\_id = Customers.customer\_id; |

**Find customers who placed more than one order:**

|  |
| --- |
| SELECT  Customers.name,  Customers.email,  COUNT(Orders.order\_id) AS total\_orders  FROM  Customers  JOIN  Orders ON Customers.customer\_id = Orders.customer\_id  GROUP BY  Customers.customer\_id  HAVING  COUNT(Orders.order\_id) > 1; |

**Get all delivered orders with their transaction IDs:**

|  |
| --- |
| SELECT  Orders.order\_id,  Customers.name AS customer\_name,  Orders.total\_amount,  Payments.transaction\_id  FROM  Orders  JOIN  Payments ON Orders.order\_id = Payments.order\_id  JOIN  Customers ON Orders.customer\_id = Customers.customer\_id  WHERE  Orders.status = 'Delivered'  AND  Payments.status = 'Completed'; |

**4. Implementation & Results**

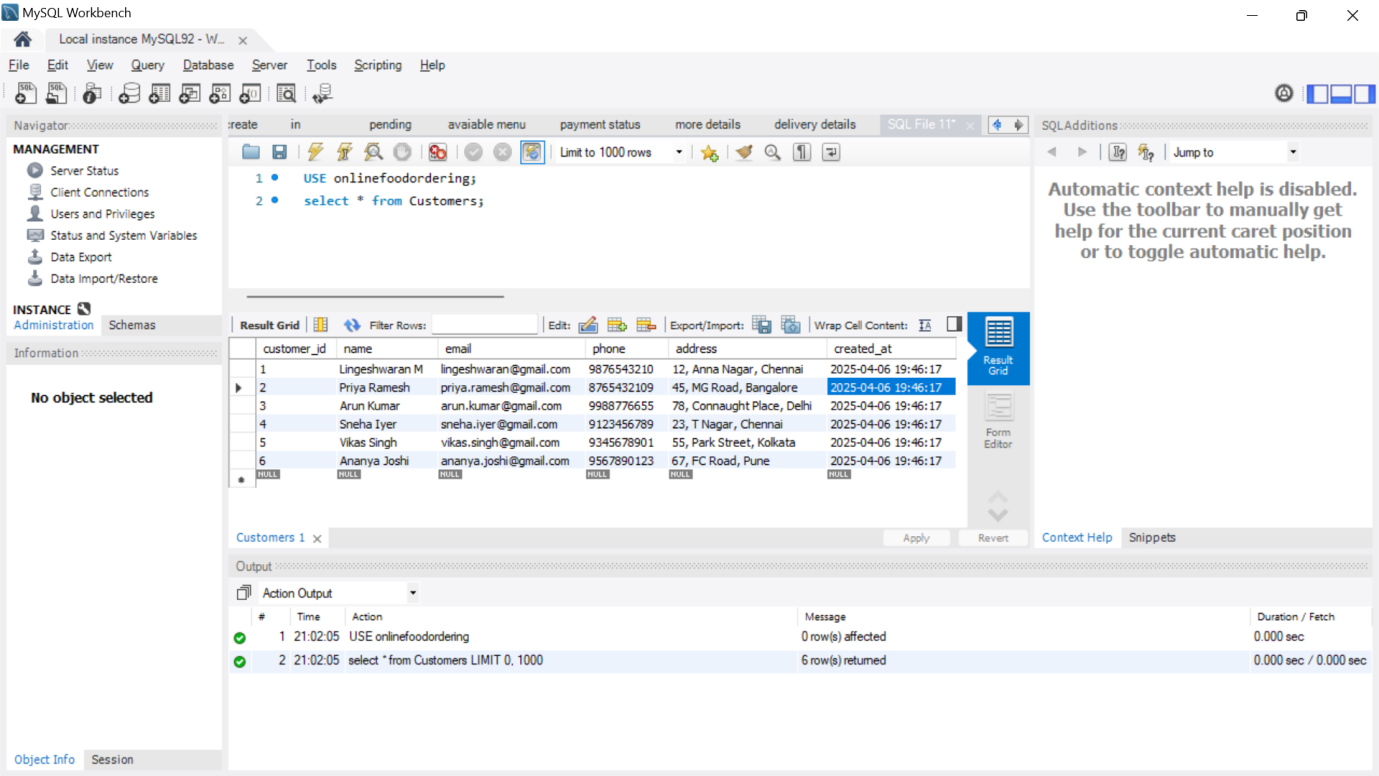
**4.1 Execution Environment:**

* **Database Server:** MySQL Server 8.0
* **Database Client:** MySQL Workbench 8.0
* **Operating System:** Windows 11
* **Processor:** 12th Gen Intel(R) Core(TM) i5-1235U 1.30 GHz
* **RAM:** 16 GB
* **Software Requirements:**
* MySQL Workbench for database design, query execution, and management

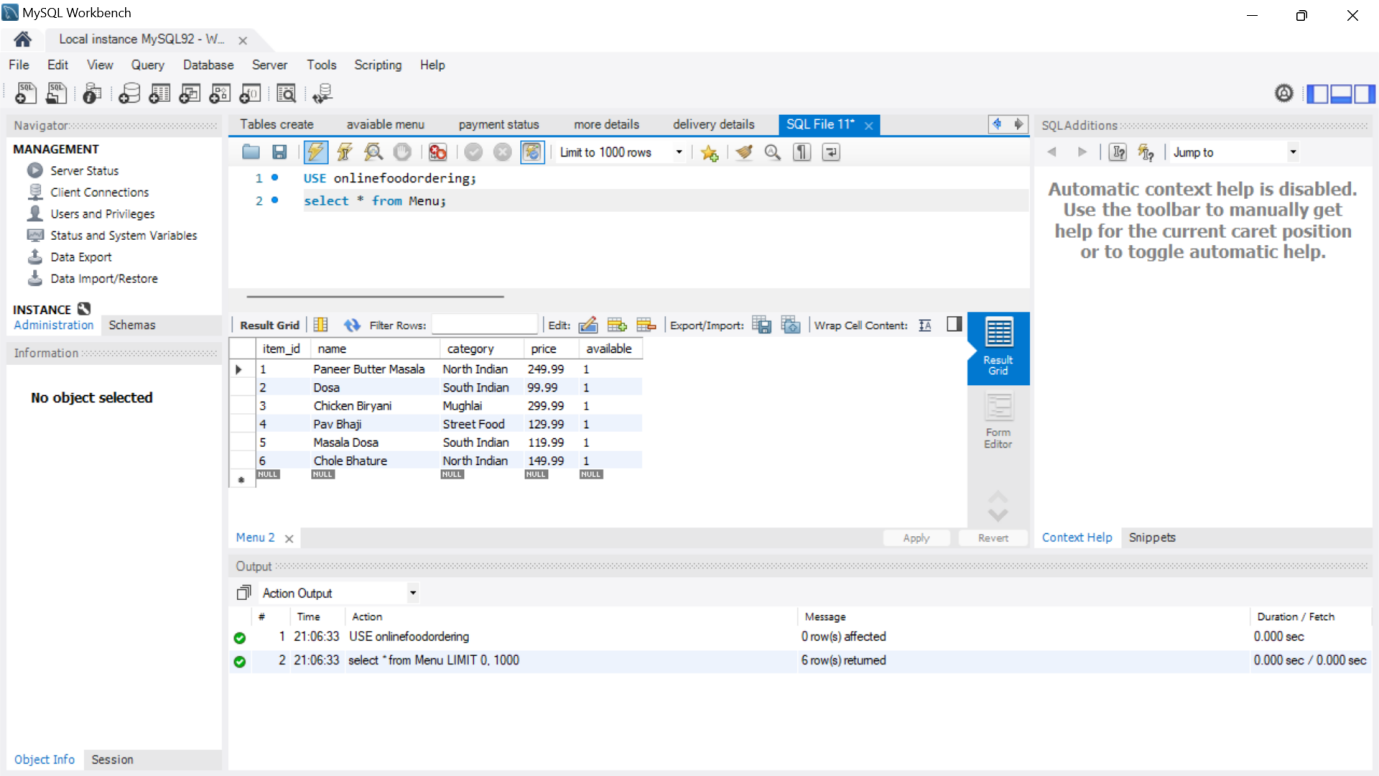
**4.2 Execution Results:**

**4.2.1 Data Insertion Results:**

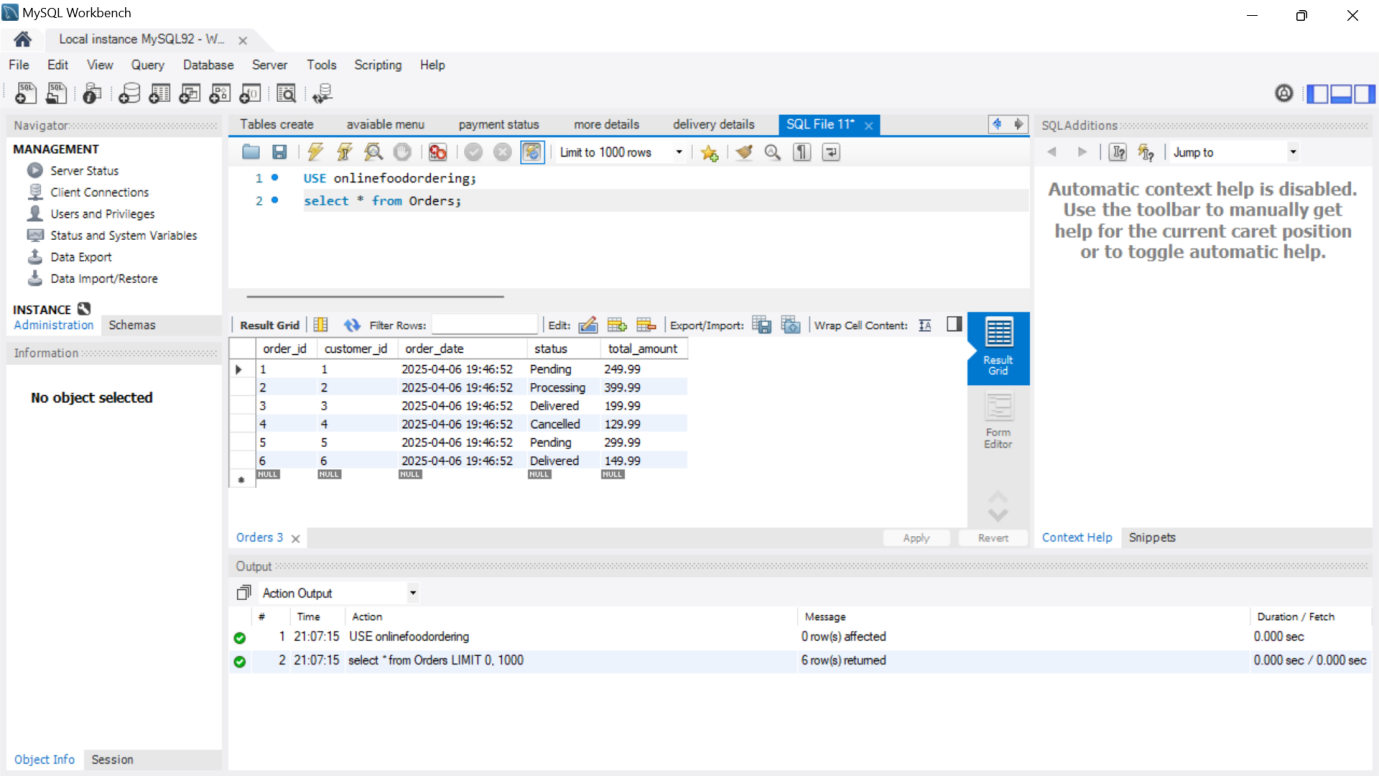
**Customer Table**

****

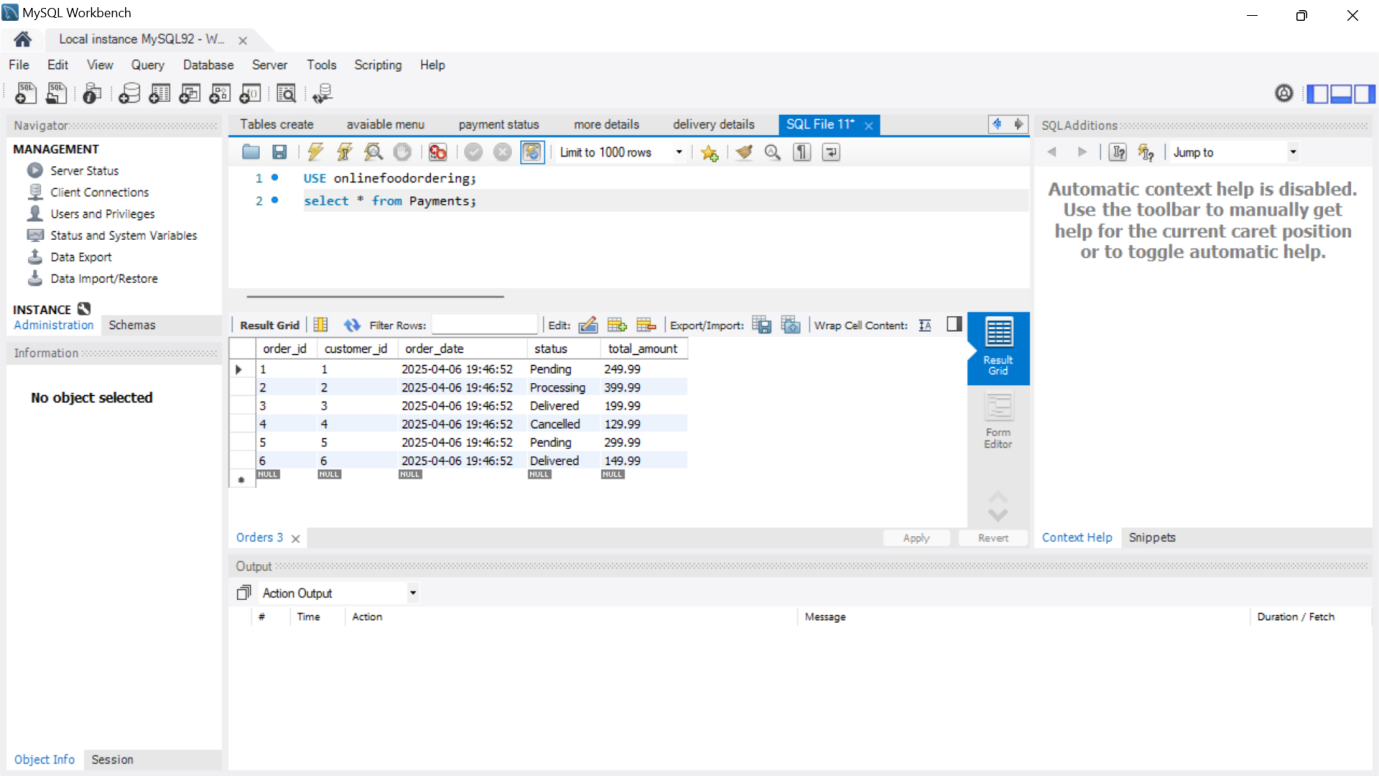
**Menu Table**

****

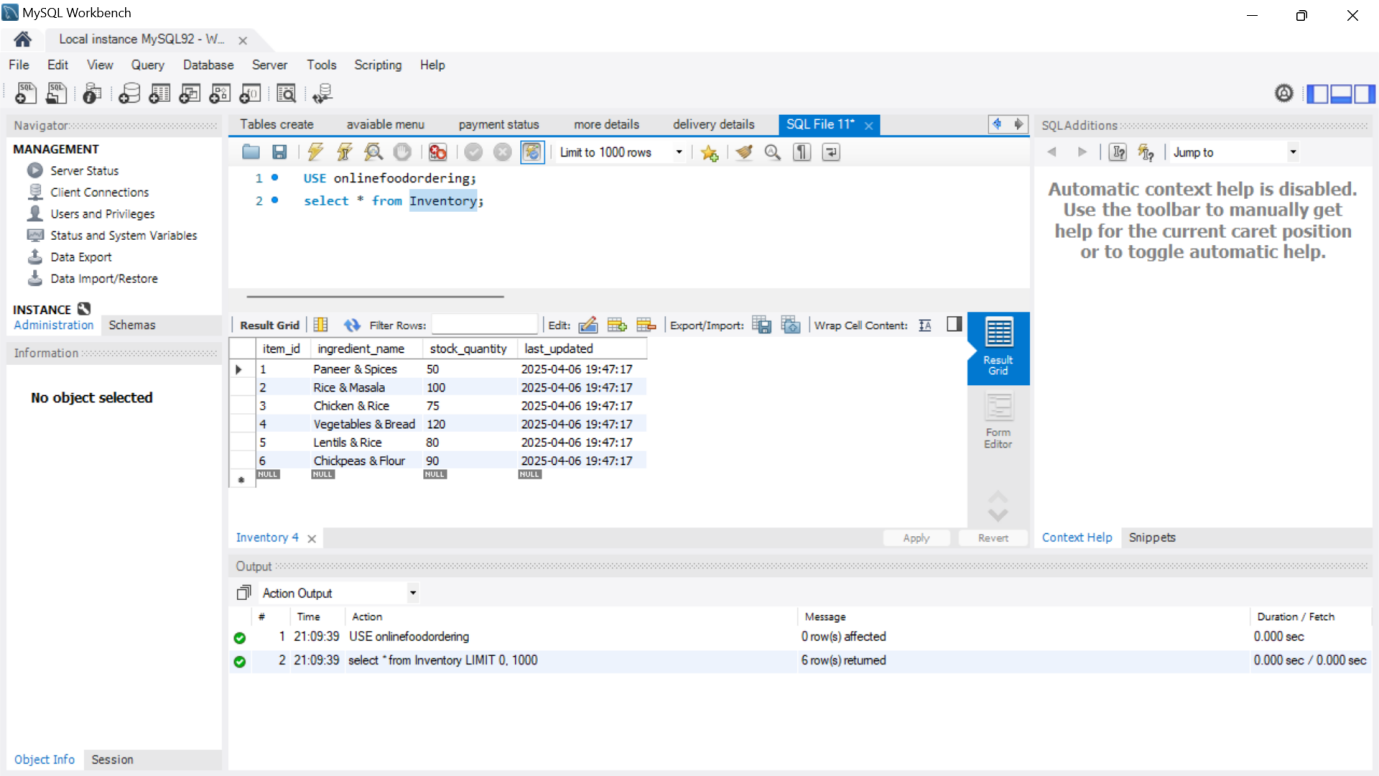
**Order Table**

****

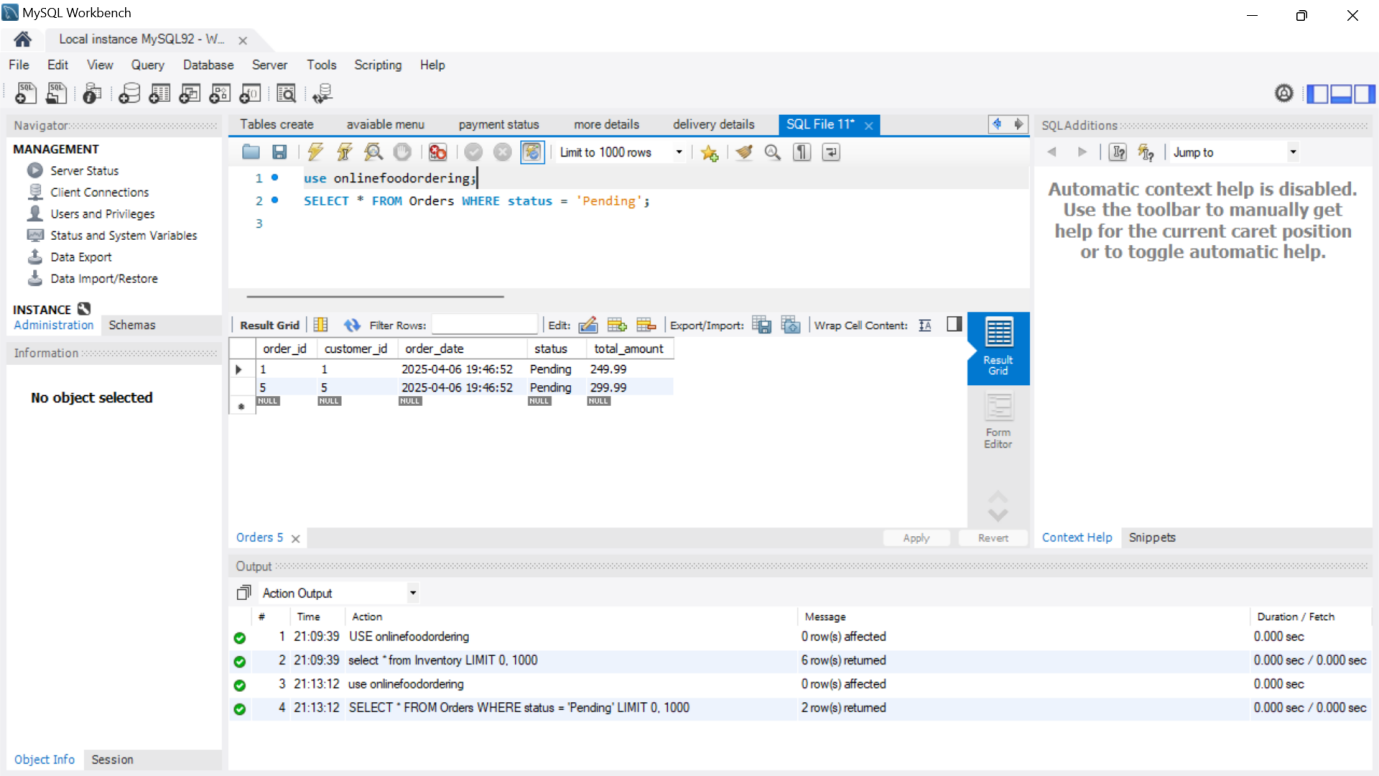
**Payment Table**

****

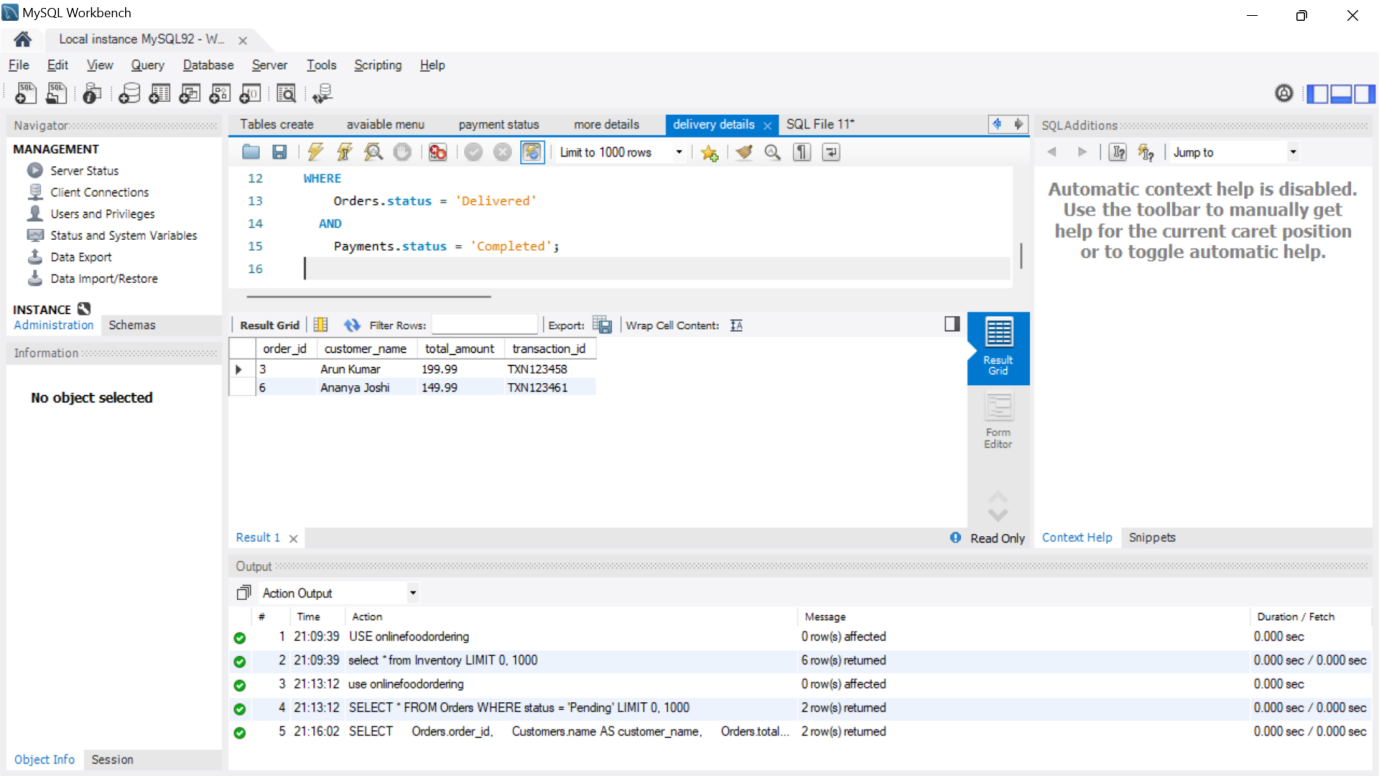
**Inventory Table**

****

**4.2.2 Retrieval Query Execution**

****

**Delivery Details**

****

**5. GitHub Repository**

**5.1 Repository Link**

**<https://github.com/Devadharshini1/Hackathon-2> (Ctrl + Click)**

**5.2 Uploaded Files in Repository**

* Repository
* Scripts
* Query results
* ER diagrams
* Documentation