Practical No. 1

Objective: To design and implement a relational database schema for an online food delivery system. The objective is to simulate a real-world scenario involving users, restaurants, menu items, orders, delivery agents, payments, and reviews.

SQL Implementation Begins Here:

```
1. create database swiggy;
2. show databases;
3. use swiggy;


4. CREATE TABLE Users (
5.     u_id INT PRIMARY KEY AUTO_INCREMENT,
6.     name VARCHAR(100),
7.     email VARCHAR(100) UNIQUE,
8.     phone VARCHAR(15) UNIQUE,
9.    password TEXT,
10.    address TEXT,
11.    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
12. );


13. CREATE TABLE Restaurants (
14.     r_id INT PRIMARY KEY AUTO_INCREMENT,
15.     name VARCHAR(100),
16.     email VARCHAR(100),
17.     phone VARCHAR(15),
18.     address TEXT,
19.     cuisine_type VARCHAR(100),
20.     rating DECIMAL(2,1),
21.     opening_time TIME,
22.     closing_time TIME,
23.     is_open BOOLEAN DEFAULT TRUE,
24.     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
25. );
```

```sql
CREATE TABLE Menu (
    item_id INT PRIMARY KEY AUTO_INCREMENT,
    r_id INT,
    name VARCHAR(100),
    description TEXT,
    price DECIMAL(10,2),
    category VARCHAR(50),
    is_available BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (r_id) REFERENCES Restaurants(r_id)
);

CREATE TABLE Orders (
    order_id INT PRIMARY KEY AUTO_INCREMENT,
    u_id INT,
    r_id INT,
    total_amount DECIMAL(10,2),
    order_status VARCHAR(20),
    placed_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    delivered_at TIMESTAMP NULL,
    FOREIGN KEY (u_id) REFERENCES Users(u_id),
    FOREIGN KEY (r_id) REFERENCES Restaurants(r_id)
);


CREATE TABLE DeliveryAgents (
    agent_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    phone VARCHAR(15),
    current_status VARCHAR(20),
    assigned_area VARCHAR(100)
);


CREATE TABLE Payments (
    payment_id INT PRIMARY KEY AUTO_INCREMENT,
    order_id INT,
    u_id INT,
    amount DECIMAL(10,2),
    payment_method VARCHAR(50),
    payment_status VARCHAR(20),
    paid_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    FOREIGN KEY (order_id) REFERENCES Orders(order_id),
    FOREIGN KEY (u_id) REFERENCES Users(u_id)
);


CREATE TABLE Reviews (
    review_id INT PRIMARY KEY AUTO_INCREMENT,
    u_id INT,
    r_id INT,
    rating INT CHECK (rating BETWEEN 1 AND 5),
    comment TEXT,
    review_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (u_id) REFERENCES Users(u_id),
    FOREIGN KEY (r_id) REFERENCES Restaurants(r_id)
);


show tables;
```

Practical No. 2

Objective: To understand how to simulate real-world scenarios using a relational database schema for an online food delivery system (e.g., Swiggy) and perform data insertion and retrieval using SQL queries. This includes working with multiple related entities like Users, Restaurants, MenuItems, Orders, OrderItems, and Payments to retrieve meaningful insights from the data.

SQL Implementation Begins Here:

```
1. show databases;
2. use swiggy;
3. show tables;


4. INSERT INTO Users (name, email, phone, password,
address) VALUES ('Ravi Kumar', 'ravi.kumar@gmail.com',
'9876543210', 'pass@123', '123 MG Road, Bengaluru'),
('Priya Sharma', 'priya.sharma@yahoo.com', '9123456789',
'priya@456', 'Sector 22, Noida'),
('Amit Verma', 'amit.verma@rediffmail.com', '9988776655',
'amit789', 'Kothrud, Pune');


5. INSERT INTO Restaurants (name, email, phone, address,
cuisine_type, rating, opening_time, closing_time) VALUES
('Tandoori Nights', 'tandoori@gmail.com', '9012345678',
'Lajpat Nagar, Delhi', 'North Indian', 4.2, '11:00:00',
'23:00:00'),
('Dosa Express', 'dosaexpress@blr.com', '9112233445',
'Indiranagar, Bengaluru', 'South Indian', 4.5, '08:00:00',
'22:00:00'),
('Biryani House', 'biryanihouse@hyd.in', '9123456677',
'Banjara Hills, Hyderabad', 'Hyderabadi', 4.3, '10:00:00',
'23:30:00');
```

```
6. INSERT INTO Menu (r_id, name, description, price,
category) VALUES
(1, 'Butter Chicken', 'Creamy chicken in rich tomato
gravy', 320.00, 'Main Course'),
(1, 'Paneer Tikka', 'Grilled paneer with spices', 250.00,
'Starter'),
(2, 'Masala Dosa', 'Crispy dosa filled with potato
masala', 100.00, 'Breakfast'),
(2, 'Filter Coffee', 'Authentic South Indian coffee',
40.00, 'Beverage'),
(3, 'Chicken Biryani', 'Spicy rice with chicken', 220.00,
'Main Course'),
(3, 'Double Ka Meetha', 'Traditional Hyderabadi dessert',
90.00, 'Dessert');


7. INSERT INTO Orders (u_id, r_id, total_amount,
order_status, delivered_at) VALUES
(1, 1, 570.00, 'Delivered', '2025-04-10 13:45:00'),
(2, 2, 140.00, 'Delivered', '2025-04-11 09:15:00'),
(3, 3, 310.00, 'In Progress', NULL);


8. INSERT INTO DeliveryAgents (name, phone,
current_status, assigned_area) VALUES
('Rakesh Yadav', '9876512345', 'Available', 'South
Delhi'),
('Sunita Rao', '9988774411', 'On Delivery',
'Koramangala'),
('Imran Khan', '9111223344', 'Available', 'Charminar,
Hyderabad');


9. INSERT INTO Payments (order_id, u_id, amount,
payment_method, payment_status) VALUES
(1, 1, 570.00, 'UPI', 'Paid'),
(2, 2, 140.00, 'Cash on Delivery', 'Paid'),
(3, 3, 310.00, 'Credit Card', 'Pending');


10. INSERT INTO Reviews (u_id, r_id, rating, comment)
VALUES (1, 1, 5, 'Amazing butter chicken! Must try.'),
```

```
(2, 2, 4, 'Dosa was crispy and fresh. Coffee was nice.'),
(3, 3, 3, 'Biryani was average, but dessert was
great.');


11. SELECT * FROM Users;
12. SELECT * FROM Restaurants;
13. SELECT * FROM Menu;
14. SELECT * FROM Orders;
15. SELECT * FROM DeliveryAgents;
16. SELECT * FROM Payments;
17. SELECT * FROM Reviews;
```

(a) Orders placed by a specific user

```
1. SELECT o.order_id, u.name AS user_name, r.name AS
restaurant_name, m.name AS menu_item, m.price,
o.order_status
2. FROM Orders o
3. JOIN Users u ON o.u_id = u.u_id
4. JOIN Restaurants r ON o.r_id = r.r_id
5. JOIN Menu m ON m.r_id = r.r_id
6. WHERE u.u_id = 1;
```

(b) Total revenue generated by each restaurant

```
SELECT r.name AS restaurant_name, SUM(o.total_amount) AS
total_revenue FROM Orders o JOIN Restaurants r ON o.r_id =
r.r_id GROUP BY r.name;
```

(c) Find the top 3 highest-priced menu items across all restaurants.

```
SELECT name AS menu_item, price, category, r_id
FROM Menu ORDER BY price DESC LIMIT 3;
```

(d) All pending payments with customer names and methods

```
SELECT u.name AS customer_name, p.payment_method, p.amount
FROM Payments p JOIN Users u ON p.u_id = u.u_id
WHERE p.payment_status = 'Pending';
```

(e) All menu items from a specific restaurant

```
SELECT m.name, m.description, m.price, m.category
FROM Menu m WHERE m.r_id = 1;
```