**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. );

26. CREATE TABLE Menu (

27. item\_id INT PRIMARY KEY AUTO\_INCREMENT,

28. r\_id INT,

29. name VARCHAR(100),

30. description TEXT,

31. price DECIMAL(10,2),

32. category VARCHAR(50),

33. is\_available BOOLEAN DEFAULT TRUE,

34. FOREIGN KEY (r\_id) REFERENCES Restaurants(r\_id)

35. );

36. CREATE TABLE Orders (

37. order\_id INT PRIMARY KEY AUTO\_INCREMENT,

38. u\_id INT,

39. r\_id INT,

40. total\_amount DECIMAL(10,2),

41. order\_status VARCHAR(20),

42. placed\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

43. delivered\_at TIMESTAMP NULL,

44. FOREIGN KEY (u\_id) REFERENCES Users(u\_id),

45. FOREIGN KEY (r\_id) REFERENCES Restaurants(r\_id)

46. );

47. CREATE TABLE DeliveryAgents (

48. agent\_id INT PRIMARY KEY AUTO\_INCREMENT,

49. name VARCHAR(100),

50. phone VARCHAR(15),

51. current\_status VARCHAR(20),

52. assigned\_area VARCHAR(100)

53. );

54. CREATE TABLE Payments (

55. payment\_id INT PRIMARY KEY AUTO\_INCREMENT,

56. order\_id INT,

57. u\_id INT,

58. amount DECIMAL(10,2),

59. payment\_method VARCHAR(50),

60. payment\_status VARCHAR(20),

61. paid\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

62. FOREIGN KEY (order\_id) REFERENCES Orders(order\_id),

63. FOREIGN KEY (u\_id) REFERENCES Users(u\_id)

64. );

65. CREATE TABLE Reviews (

66. review\_id INT PRIMARY KEY AUTO\_INCREMENT,

67. u\_id INT,

68. r\_id INT,

69. rating INT CHECK (rating BETWEEN 1 AND 5),

70. comment TEXT,

71. review\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

72. FOREIGN KEY (u\_id) REFERENCES Users(u\_id),

73. FOREIGN KEY (r\_id) REFERENCES Restaurants(r\_id)

74. );

75. 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;