**DataBase : Zomato\_db**

1. **Create table Restaurants**

CREATE TABLE Restaurants ( RestaurantID INT AUTO\_INCREMENT PRIMARY KEY, Name VARCHAR(100), City VARCHAR(50), CuisineType VARCHAR(50), Rating DECIMAL(2, 1), AverageCostForTwo DECIMAL(10, 2) );

Insert data into table

INSERT INTO Restaurants (Name, City, CuisineType, Rating, AverageCostForTwo) VALUES ('Tandoori Flames', 'Mumbai', 'Indian', 4.7, 800), ('Pasta Paradise', 'Mumbai', 'Italian', 4.6, 1200), ('Dragon Delight', 'Delhi', 'Chinese', 4.5, 1000), ('Indian Spices', 'Chennai', 'Indian', 4.3, 700), ('Pizza Hub', 'Mumbai', 'Italian', 4.4, 1500);

1. **Create table Costumers**

CREATE TABLE Customers (

CustomerID INT AUTO\_INCREMENT PRIMARY KEY,

FirstName VARCHAR(50) not null,

LastName VARCHAR(50) not null,

Phone VARCHAR(15) ,

City VARCHAR(50) not null,

JoinDate DATE

);

Insert data into Costumer table

INSERT INTO Customers (FirstName, LastName, Phone, City, JoinDate) VALUES ('Rohit', 'Sharma', '9876543210', 'Mumbai', '2023-01-10'), ('Priya', 'Singh', '9876543211', 'Delhi', '2023-02-20'), ('Amit', 'Kumar', '9876543212', 'Mumbai', '2023-03-15'), ('Sneha', 'Patil', '9876543213', 'Chennai', '2023-04-25');

1. **Create table Orders**

CREATE TABLE Orders ( OrderID INT AUTO\_INCREMENT PRIMARY KEY, CustomerID INT, RestaurantID INT, OrderDate DATE, OrderAmount DECIMAL(10, 2), OrderStatus VARCHAR(20), FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID), FOREIGN KEY (RestaurantID) REFERENCES Restaurants(RestaurantID) );

Insert data into orders table

INSERT INTO Orders (CustomerID, RestaurantID, OrderDate, OrderAmount, OrderStatus) VALUES (1, 1, '2023-05-01', 1200, 'Delivered'), (2, 3, '2023-05-03', 1500, 'Cancelled'), (3, 2, '2023-05-05', 1100, 'Delivered'), (1, 5, '2023-05-10', 900, 'Pending');

1. **Create Reviews table**

CREATE TABLE Reviews ( ReviewID INT AUTO\_INCREMENT PRIMARY KEY, CustomerID INT, RestaurantID INT, Rating DECIMAL(2, 1), Comment TEXT, ReviewDate DATE, FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID), FOREIGN KEY (RestaurantID) REFERENCES Restaurants(RestaurantID) );

Insert data into reviews table

INSERT INTO Reviews (CustomerID, RestaurantID, Rating, Comment, ReviewDate) VALUES (1, 1, 4.8, 'Amazing food and service!', '2023-05-02'), (2, 3, 4.5, 'Great taste!', '2023-05-04'), (3, 2, 4.6, 'Loved the pasta!', '2023-05-06');

1. **Create payments table**

CREATE TABLE Payments ( PaymentID INT AUTO\_INCREMENT PRIMARY KEY, OrderID INT, PaymentMethod VARCHAR(20), Amount DECIMAL(10, 2), PaymentDate DATE, FOREIGN KEY (OrderID) REFERENCES Orders(OrderID) );

**Queires**

1. **Retrieve the names and locations of restaurants with a rating of 4.5 or higher.**

🡺SELECT \* from restaurants WHERE restaurants.Rating >= 4.5

1. **Find the total number of orders placed by each customer.**

**🡺** SELECT customers.FirstName,customers.LastName,COUNT(orders.OrderID)as placed\_orders from customers

LEFT JOIN orders on customers.CustomerID = orders.CustomerID

GROUP by customers.FirstName

1. **List all restaurants offering "Italian" cuisine in "Mumbai".**

🡺SELECT \* FROM restaurants WHERE restaurants.CuisineType = "Italian" or restaurants.City = "Mumbai"

1. **Calculate the total revenue generated by each restaurant from completed orders.**

🡺SELECT restaurants.Name, SUM(orders.OrderAmount) as Total\_Revenue FROM restaurants

JOIN orders on restaurants.RestaurantID = orders.RestaurantID

WHERE orders.OrderStatus = "Delivered"

GROUP by restaurants.RestaurantID

1. **Retrieve the most recent order placed by each customer.**

🡺SELECT customers.FirstName,customers.LastName ,max(orders.OrderDate) As Reent\_order

FROM customers JOIN orders on customers.CustomerID = orders.CustomerID

GROUP by customers.CustomerID

1. **List customers who have not placed any orders yet.**

🡺SELECT customers.FirstName , customers.LastName from customers

LEFT JOIN orders on customers.CustomerID = orders.CustomerID

WHERE orders.CustomerID is null

1. **Identify the most reviewed restaurants.**

🡺SELECT restaurants.Name,reviews.Rating,COUNT(reviews.ReviewID) as Most\_Reviews FROM restaurants

JOIN reviews on restaurants.RestaurantID = reviews.RestaurantID

GROUP by restaurants.RestaurantID order by Most\_Reviews

1. **Find the most preferred payment method.**

🡺 SELECT payments.PaymentMethod , COUNT(payments.OrderID) FROM payments

JOIN orders on payments.OrderID = orders.OrderID

GROUP by payments.PaymentMethod

1. **List the top 5 restaurants by total revenue**

🡺 SELECT restaurants.Name , restaurants.City ,SUM(orders.OrderAmount) FROM restaurants

JOIN orders on restaurants.RestaurantID = orders.RestaurantID

JOIN payments on payments.OrderID = orders.OrderID

GROUP by restaurants.RestaurantID

1. **Show the details of all cancelled orders along with the customer's and restaurant's names.**

🡺 SELECT customers.FirstName,customers.LastName,restaurants.Name,restaurants.City ,orders.OrderStatus FROM

customers JOIN orders on customers.CustomerID = orders.CustomerID

JOIN restaurants on orders.RestaurantID = restaurants.RestaurantID

WHERE orders.OrderStatus = "Cancelled"