SQL Assignment Courier Management System

Student Name: Sugandan Elangovan

**Task1 :**

**Database Design Design a SQL schema for a Courier Management System with tables for Customers, Couriers, Orders, and Parcels. Define the relationships between these tables using appropriate foreign keys.**

CREATE TABLE Users (

UserID INT PRIMARY KEY,

Name VARCHAR(255),

Email VARCHAR(255) UNIQUE,

Password VARCHAR(255),

ContactNumber VARCHAR(20),

Address TEXT

);

CREATE TABLE CourierServices (

ServiceID INT PRIMARY KEY,

ServiceName VARCHAR(100),

Cost DECIMAL(8, 2)

);

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

Name VARCHAR(255),

Email VARCHAR(255) UNIQUE,

ContactNumber VARCHAR(20),

Role VARCHAR(50),

Salary DECIMAL(10, 2)

);

CREATE TABLE Locations (

LocationID INT PRIMARY KEY,

LocationName VARCHAR(100),

Address TEXT

);

CREATE TABLE Couriers (

CourierID INT PRIMARY KEY,

SenderName VARCHAR(255),

SenderAddress TEXT,

ReceiverName VARCHAR(255),

ReceiverAddress TEXT,

Weight DECIMAL(5, 2),

Status VARCHAR(50),

TrackingNumber VARCHAR(20) UNIQUE,

DeliveryDate DATE

LocationID INT,

EmployeeID INT,

ServiceID INT,

CONSTRAINT FK\_Couriers\_Location FOREIGN KEY (LocationID) REFERENCES Locations(LocationID),

CONSTRAINT FK\_Couriers\_Employee FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID),

CONSTRAINT FK\_Couriers\_Service FOREIGN KEY (ServiceID) REFERENCES CourierServices(ServiceID)

);

CREATE TABLE Payments (

PaymentID INT PRIMARY KEY,

CourierID INT,

LocationID INT,

Amount DECIMAL(10, 2),

PaymentDate DATE,

EmployeeID INT,

FOREIGN KEY (CourierID) REFERENCES Couriers(CourierID),

FOREIGN KEY (LocationID) REFERENCES Locations(LocationID),

FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID),

);

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

OrderDate DATE,

CONSTRAINT FK\_Orders\_Customers FOREIGN KEY (CustomerID) REFERENCES Users(UserID)

);

CREATE TABLE Parcels (

ParcelID INT PRIMARY KEY,

OrderID INT,

CourierID INT,

ServiceID INT,

Weight DECIMAL(5, 2),

Status VARCHAR(50),

TrackingNumber VARCHAR(20) UNIQUE,

DeliveryDate DATE,

EmployeeID INT,

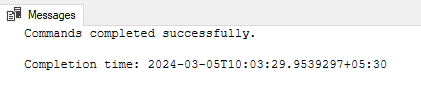
CONSTRAINT FK\_Parcels\_Orders FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

CONSTRAINT FK\_Parcels\_Couriers FOREIGN KEY (CourierID) REFERENCES Couriers(CourierID),

CONSTRAINT FK\_Parcels\_Services FOREIGN KEY (ServiceID) REFERENCES CourierServices(ServiceID),

CONSTRAINT FK\_Parcels\_Employee FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID)

);



INSERT INTO Users (UserID, Name, Email, Password, ContactNumber, Address)

VALUES(1, 'Rajesh Kumar', 'rajesh.kumar@email.com', 'password123', '9876543210', '12 Gandhi Nagar, Chennai'),

(2, 'Priya Sharma', 'priya.sharma@email.com', 'password456', '8765432109', '34 Kaveri Street, Bangalore'),

(3, 'Amit Patel', 'amit.patel@email.com', 'password789', '7654321098', '56 Krishna Lane, Hyderabad'),

(4, 'Ananya Singh', 'ananya.singh@email.com', 'passwordabc', '6543210987', '78 Vindhya Nagar, Coimbatore');

INSERT INTO CourierServices (ServiceID, ServiceName, Cost)

VALUES(1, 'Standard', 10.00),

(2, 'Express', 15.00);

INSERT INTO Employees (EmployeeID, Name, Email, ContactNumber, Role, Salary)

VALUES(1, 'Manager1', 'manager1@email.com', '1112223333', 'Manager', 50000.00),

(2, 'DeliveryPerson1', 'delivery1@email.com', '4445556666', 'Delivery Person', 30000.00),

(3, 'DeliveryPerson2', 'delivery2@email.com', '5556667777', 'Delivery Person', 30000.00);

INSERT INTO Employees (EmployeeID, Name, Email, ContactNumber, Role, Salary)

VALUES(4, 'John cena', 'john.cena@email.com', '1234567890', 'Manager', 60000.00),

(5, 'Authur Johnson', 'authur.johnson@email.com', '9876543210', 'Clerk', 45000.00);

INSERT INTO Locations (LocationID, LocationName, Address)

VALUES(1, 'Warehouse1', '789 Storage St, Chennai'),

(2, 'Warehouse2', '456 Logistics Ave, Bangalore'),

(3, 'Warehouse3', '123 Distribution Rd, Hyderabad');

INSERT INTO Couriers (CourierID, SenderName, SenderAddress, ReceiverName, ReceiverAddress, Weight, Status, TrackingNumber, DeliveryDate, LocationID,EmployeeID,ServiceID)

VALUES(1, 'Rajesh Kumar', '12 Gandhi Nagar, Chennai', 'Ananya Singh', '78 Vindhya Nagar, Coimbatore', 2.5, 'In Transit', 'TN123456', '2024-03-01',1,2,1),

(2, 'Priya Sharma', '34 Kaveri Street, Bangalore', 'Amit Patel', '56 Krishna Lane, Hyderabad', 1.8, 'Delivered', 'TN789012', '2024-03-02',1,2,2),

(3, 'Amit Patel', '56 Krishna Lane, Hyderabad', 'Priya Sharma', '34 Kaveri Street, Bangalore', 3.0, 'In Transit', 'TN345678', '2024-03-03',2,3,2),

(4, 'Ananya Singh', '78 Vindhya Nagar, Coimbatore', 'Rajesh Kumar', '12 Gandhi Nagar, Chennai', 4.0, 'In Transit', 'TN345679', '2024-03-05',2,3,2);

INSERT INTO Payments (PaymentID, CourierID, LocationID, Amount, PaymentDate,EmployeeID)

VALUES(1, 1, 1, 10000.00, '2024-03-03',2),

(2, 2, 2, 1500.00, '2024-03-04',2),

(3, 3, 3, 1200.50, '2024-03-05',3);

---

INSERT INTO Orders (OrderID, CustomerID, OrderDate)

VALUES(1, 1, '2024-02-01'),

(2, 2, '2024-02-02'),

(3, 3, '2024-02-03');

INSERT INTO Parcels (ParcelID, OrderID, CourierID, ServiceID, Weight, Status, TrackingNumber, DeliveryDate,EmployeeID)

VALUES(1, 1, 1, 1, 2.5, 'In Transit', 'TN123456', '2024-03-01',2),

(2, 2, 2, 2, 1.8, 'Delivered', 'TN789012', '2024-03-02',2),

(3, 3, 3, 1, 3.0, 'In Transit', 'TN345678', '2024-03-03',3);

INSERT INTO Users (UserID, Name, Email, Password, ContactNumber, Address)

VALUES

(5, 'Sara Khan', 'sara.khan@email.com', 'passwordxyz', '5432109876', '90 Yamuna Road, Delhi'),

(6, 'Michael Johnson', 'michael.johnson@email.com', 'password123', '6547893210', '23 Park Street, Mumbai'),

(7, 'Emma Wilson', 'emma.wilson@email.com', 'password456', '7896543210', '45 Lake Avenue, Kolkata'),

(8, 'David Lee', 'david.lee@email.com', 'password789', '9876543210', '67 Forest Lane, Pune'),

(9, 'Sophia Garcia', 'sophia.garcia@email.com', 'passwordabc', '8765432109', '89 Ocean View, Goa');

INSERT INTO Orders (OrderID, CustomerID, OrderDate)

VALUES

(4, 4, '2024-02-04'),

(5, 5, '2024-02-05'),

(6, 6, '2024-02-06'),

(7, 7, '2024-02-07'),

(8, 8, '2024-02-08'),

(9, 1, '2024-02-08');

INSERT INTO Couriers (CourierID, SenderName, SenderAddress, ReceiverName, ReceiverAddress, Weight, Status, TrackingNumber, DeliveryDate, LocationID, EmployeeID, ServiceID)

VALUES

(5, 'Sara Khan', '90 Yamuna Road, Delhi', 'David Lee', '67 Forest Lane, Pune', 2.0, 'In Transit', 'TN234567', '2024-03-04', 1, 2, 2),

(6, 'Michael Johnson', '23 Park Street, Mumbai', 'Emma Wilson', '45 Lake Avenue, Kolkata', 1.5, 'Delivered', 'TN345688', '2024-03-05', 2, 3, 1),

(7, 'Emma Wilson', '45 Lake Avenue, Kolkata', 'Michael Johnson', '23 Park Street, Mumbai', 3.5, 'In Transit', 'TN456789', '2024-03-06', 3, 2, 2),

(8, 'David Lee', '67 Forest Lane, Pune', 'Sara Khan', '90 Yamuna Road, Delhi', 4.5, 'In Transit', 'TN567890', '2024-03-07', 3, 2, 1),

(9, 'Rajesh Kumar', '12 Gandhi Nagar, Chennai', 'Sara Khan', '90 Yamuna Road, Delhi', 2.8, 'Delivered', 'TN678901', '2024-03-08', 2, 3, 2);

INSERT INTO Payments (PaymentID, CourierID, LocationID, Amount, PaymentDate, EmployeeID)

VALUES

(4, 4, 1, 1500.00, '2024-03-06', 2),

(5, 5, 2, 1000.00, '2024-03-07', 3),

(6, 6, 3, 25000.00, '2024-03-08', 2),

(7, 7, 1, 2000.00, '2024-03-09', 3),

(8, 8, 2, 1200.50, '2024-03-10', 2),

(9, 9, 2, 1200.50, '2024-03-10', 2);

INSERT INTO Parcels (ParcelID, OrderID, CourierID, ServiceID, Weight, Status, TrackingNumber, DeliveryDate, EmployeeID)

VALUES

(4, 4, 4, 1, 4.0, 'In Transit', 'TN456789', '2024-03-04', 2),

(5, 5, 5, 2, 2.5, 'Delivered', 'TN567890', '2024-03-05', 3),

(6, 6, 6, 1, 3.2, 'In Transit', 'TN678901', '2024-03-06', 2),

(7, 7, 7, 2, 2.0, 'Delivered', 'TN789013', '2024-03-07', 2),

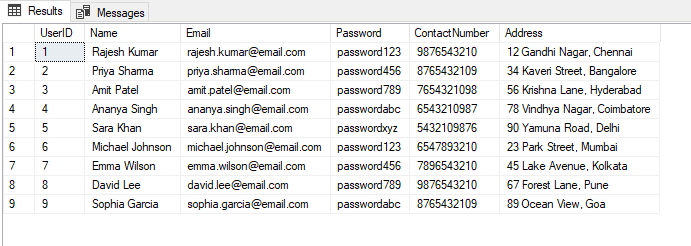
(8, 8, 8, 1, 3.5, 'In Transit', 'TN89012', '2024-03-08', 2),

(9, 9, 9, 1, 3.5, 'In Transit', 'TN89014', '2024-03-08', 2);

**Task 2: Select,Where**

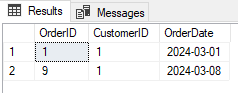
1. **List all customers**

SELECT \* FROM Users;



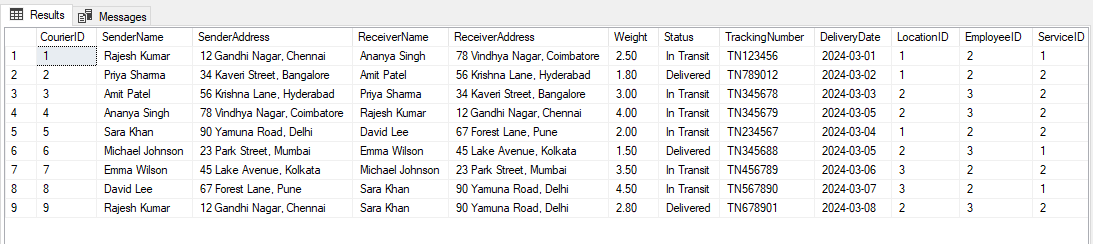
1. **List all orders for a specific customer:**

SELECT \* FROM Orders WHERE CustomerID = 1;



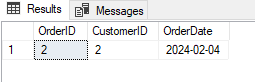
1. **List all couriers:**

SELECT \* FROM Couriers;



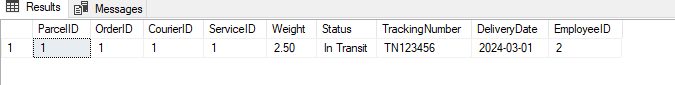
1. **List all packages for a specific order:**

SELECT \* FROM Orders WHERE OrderID = 2;



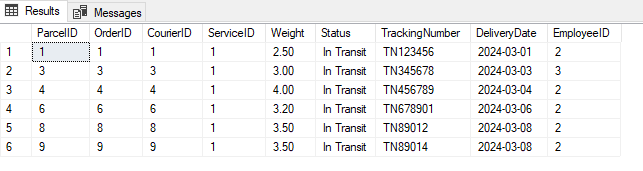
1. **List all deliveries for a specific courier:**

SELECT \* FROM Parcels WHERE CourierID = 1;



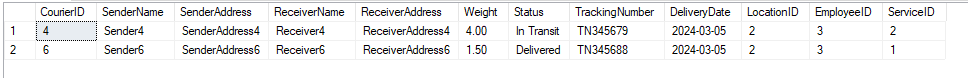
1. **List all undelivered packages:**

SELECT \* FROM Parcels WHERE Status = 'In Transit'



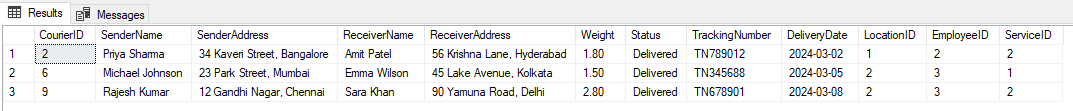
1. **List all packages that are scheduled for delivery today:**

SELECT \* FROM Couriers WHERE DeliveryDate = CAST(GETDATE() AS DATE);



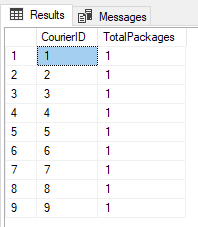
1. **List all packages with a specific status:**

SELECT \* FROM Parcels WHERE Status = 'Delivered';



1. **Calculate the total number of packages for each courier**.

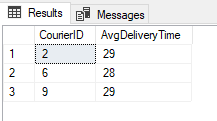
SELECT CourierID, COUNT(\*) AS TotalPackages FROM Parcels GROUP BY CourierID;

.

1. **Find the average delivery time for each courier**

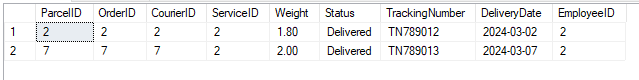
SELECT CourierID,AVG(DATEDIFF(DAY, o.OrderDate, c.DeliveryDate)) AS AvgDeliveryTime

FROM Couriers c INNER JOIN Orders o ON c.CourierID=o.OrderID AND c.Status = 'Delivered' GROUP BY c.CourierID;



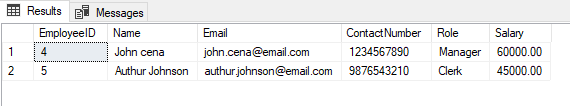
1. **List all packages with a specific weight range:**

SELECT \* FROM Parcels WHERE Weight BETWEEN 1.0 AND 2.0;



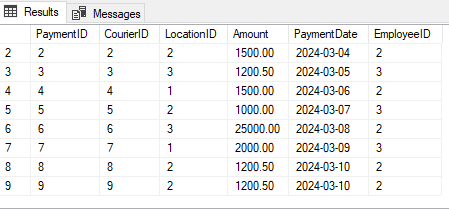
1. **Retrieve employees whose names contain 'Joh**n'

SELECT \* FROM Employees WHERE Name LIKE '%John%';



1. **Retrieve all courier records with payments greater than $50**

SELECT \* FROM Payments WHERE Amount > 50.00;

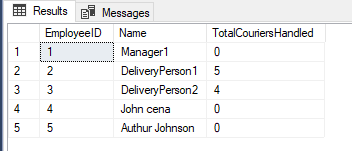


**Task 3: GroupBy, Aggregate Functions, Having, Order By, where**

1. **Find the total number of couriers handled by each employee.**

SELECT e.EmployeeID, e.Name, COUNT(p.CourierID) AS TotalCouriersHandled FROM Employees e

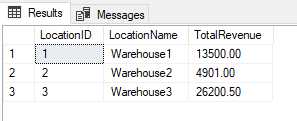
LEFT JOIN Couriers p ON e.EmployeeID = p.EmployeeID GROUP BY e.EmployeeID, e.Name;



1. **Calculate the total revenue generated by each location**

SELECT l.LocationID, l.LocationName, SUM(py.Amount) AS TotalRevenue FROM Locations l

LEFT JOIN Payments py ON l.LocationID = py.LocationID GROUP BY l.LocationID, l.LocationName;

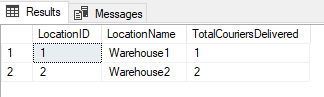


1. **Find the total number of couriers delivered to each location.**

SELECT l.LocationID, l.LocationName, COUNT(p.CourierID) AS TotalCouriersDelivered

FROM Locations l LEFT JOIN Couriers p ON l.LocationID = p.LocationID

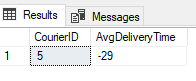
WHERE p.Status = 'Delivered' GROUP BY l.LocationID, l.LocationName;



1. **Find the courier with the highest average delivery time:**

SELECT TOP 1 CourierID, AVG(DATEDIFF(DAY, p.DeliveryDate, o.OrderDate)) AS AvgDeliveryTime FROM Parcels p, Orders o

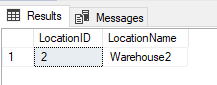
WHERE Status = 'Delivered' AND p.OrderID = o.OrderID GROUP BY CourierID ORDER BY AvgDeliveryTime DESC;



1. **Find Locations with Total Payments Less Than a Certain Amount**

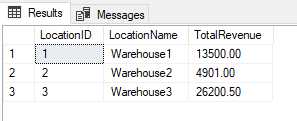
SELECT LocationID, LocationName FROM Locations WHERE LocationID

IN (SELECT LocationID FROM Payments GROUP BY LocationID HAVING SUM(Amount) < 5000);



1. **Calculate Total Payments per Location**

SELECT LocationID, SUM(Amount) AS TotalPayments FROM Payments GROUP BY LocationID;

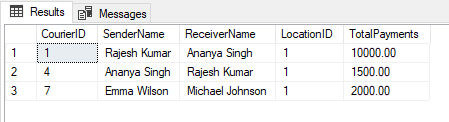


1. **Retrieve couriers who have received payments totaling more than $1000 in a specific location (LocationID = X):**

SELECT p.CourierID, c.SenderName, c.ReceiverName, p.LocationID, SUM(p.Amount) AS TotalPayments

FROM Payments p JOIN Couriers c ON p.CourierID = c.CourierID WHERE p.LocationID = 1

GROUP BY p.CourierID, c.SenderName, c.ReceiverName, p.LocationID HAVING SUM(p.Amount) > 1000;



1. **Retrieve couriers who have received payments totaling more than $1000 after a certain date (PaymentDate > 'YYYY-MM-DD'):**

SELECT p.CourierID, c.SenderName, c.ReceiverName, SUM(p.Amount) AS TotalPayments

FROM Payments p JOIN Couriers c ON p.CourierID = c.CourierID WHERE p.PaymentDate > '2024-03-01'

GROUP BY p.CourierID, c.SenderName, c.ReceiverName HAVING SUM(p.Amount) > 1000;

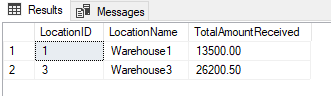


**22. Retrieve locations where the total amount received is more than $5000 before a certain date (PaymentDate > 'YYYY-MM-DD')**

SELECT l.LocationID, l.LocationName, SUM(p.Amount) AS TotalAmountReceived FROM Locations l

JOIN Payments p ON l.LocationID = p.LocationID WHERE p.PaymentDate > '2024-03-01'

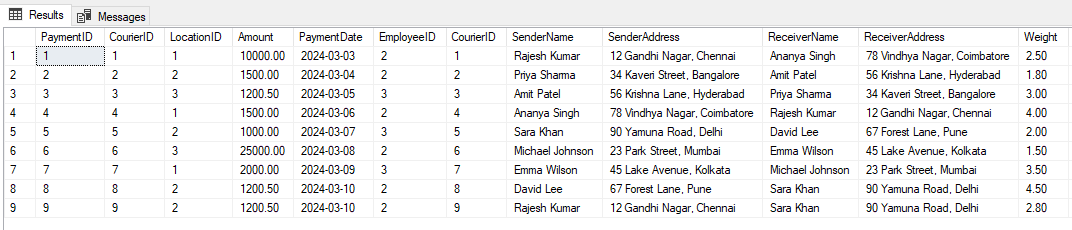
GROUP BY l.LocationID, l.LocationName HAVING SUM(p.Amount) > 5000;

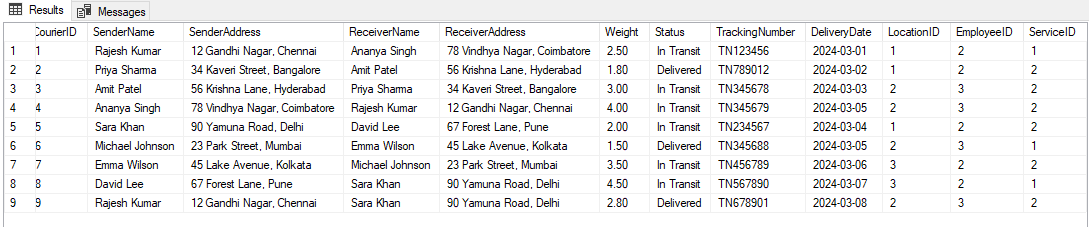


**Task 4: Inner Join,Full Outer Join, Cross Join, Left Outer Join,Right Outer Join**

**23. Retrieve Payments with Courier Information**

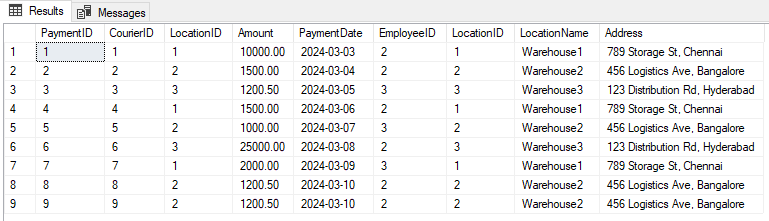
SELECT p.\*, c.\* FROM Payments p INNER JOIN Couriers c ON p.CourierID = c.CourierID;





**24. Retrieve Payments with Location Information**

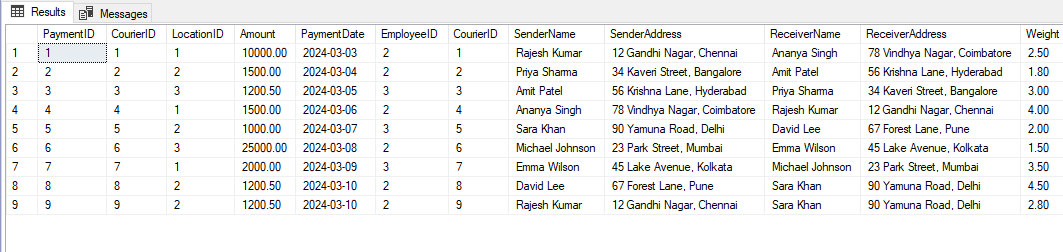
SELECT p.\*, l.\* FROM Payments p INNER JOIN Locations l ON p.LocationID = l.LocationID;

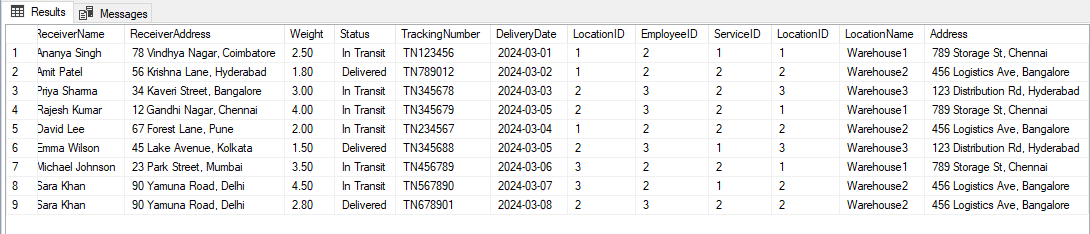


**25. Retrieve Payments with Courier and Location Information**

SELECT p.\*, c.\*, l.\* FROM Payments p INNER JOIN Couriers c ON p.CourierID = c.CourierID

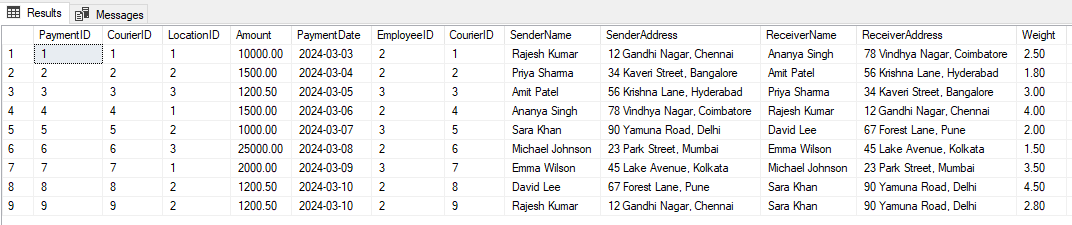
INNER JOIN Locations l ON p.LocationID = l.LocationID;

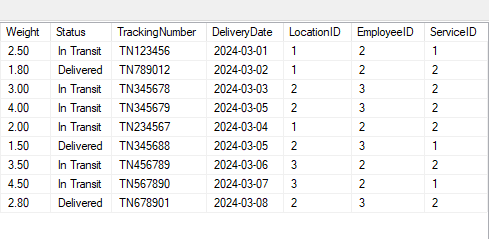




**26. List all payments with courier details**

SELECT p.\*, c.\* FROM Payments p INNER JOIN Couriers c ON p.CourierID = c.CourierID;

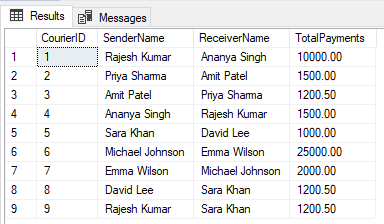




**27. Total payments received for each courier**

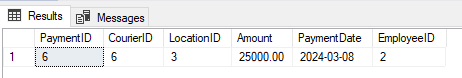
SELECT c.CourierID, c.SenderName, c.ReceiverName, SUM(p.Amount) AS TotalPayments

FROM Couriers c LEFT JOIN Payments p ON c.CourierID = p.CourierID GROUP BY c.CourierID, c.SenderName, c.ReceiverName;



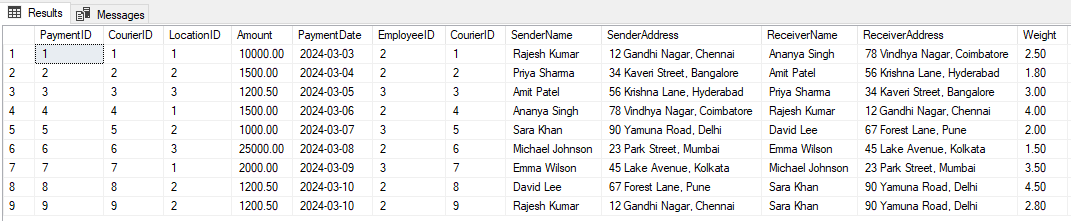
**28. List payments made on a specific date**

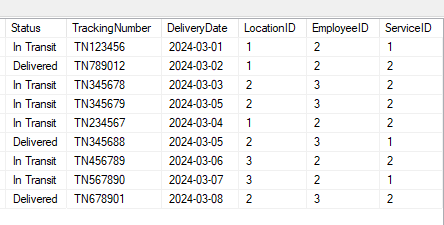
SELECT \* FROM Payments WHERE PaymentDate = '2024-03-08';



1. **Get Courier Information for Each Payment**

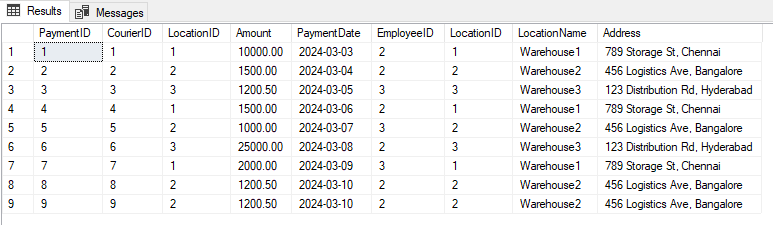
SELECT p.\*, c.\* FROM Payments p LEFT JOIN Couriers c ON p.CourierID = c.CourierID;





**30. Get Payment Details with Location**

SELECT p.\*, l.\* FROM Payments p LEFT JOIN Locations l ON p.LocationID = l.LocationID;



**31. Calculating Total Payments for Each Courier**

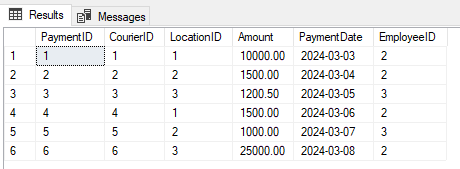
SELECT c.CourierID, c.SenderName, c.ReceiverName, SUM(p.Amount) AS TotalPayments FROM Couriers c

LEFT JOIN Payments p ON c.CourierID = p.CourierID GROUP BY c.CourierID, c.SenderName, c.ReceiverName;



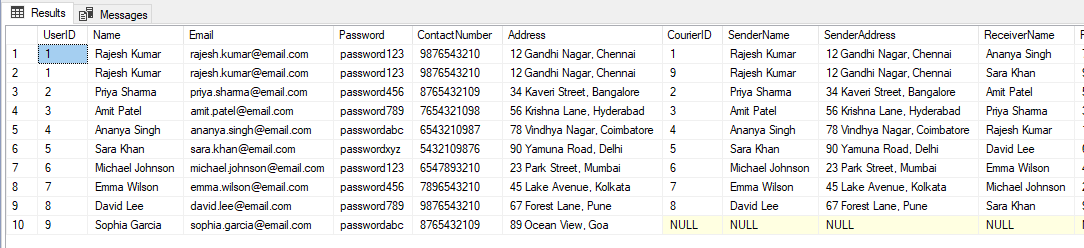
**32. List Payments Within a Date Range**

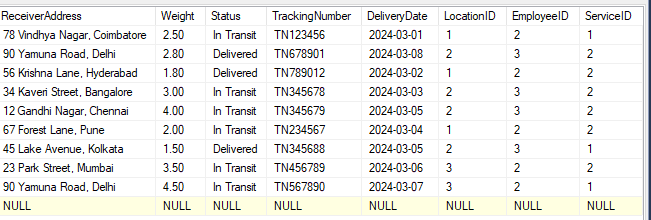
SELECT \* FROM Payments WHERE PaymentDate BETWEEN '2024-02-01' AND '2024-03-08';



**33. Retrieve a list of all users and their corresponding courier records, including cases where there are no matches on either side**

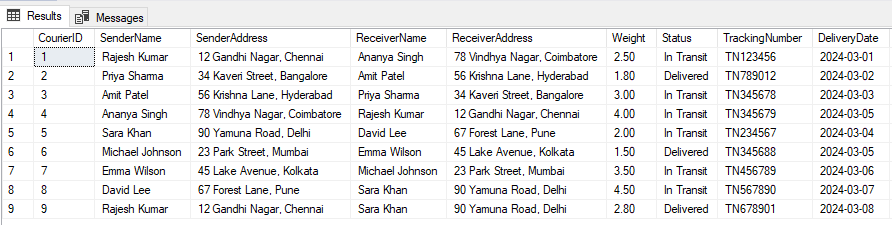
SELECT u.\*, c.\* FROM Users u FULL OUTER JOIN Couriers c ON u.name = c.SenderName;





**34. Retrieve a list of all couriers and their corresponding services, including cases where there are no matches on either side**

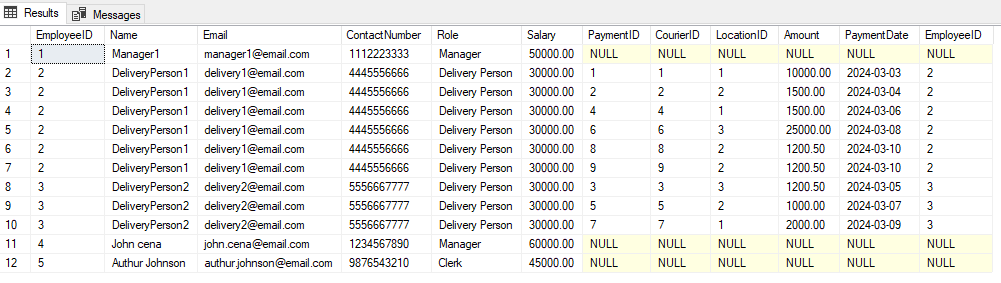
SELECT c.\*, cs.\* FROM Couriers c FULL OUTER JOIN CourierServices cs ON c.ServiceID = cs.ServiceID;





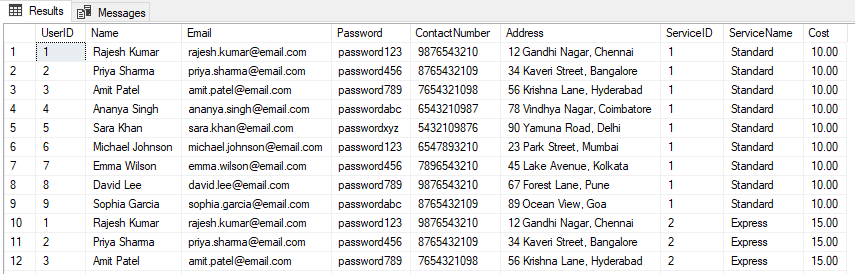
**35. Retrieve a list of all employees and their corresponding payments, including cases where there are no matches on either side**

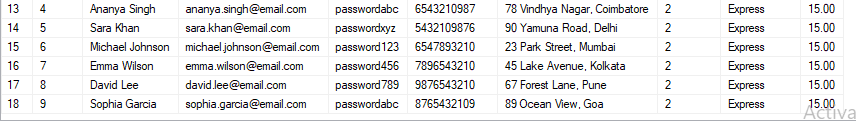
SELECT e.\*, p.\*FROM Employees e FULL OUTER JOIN Payments p ON e.EmployeeID = p.EmployeeID;



**36. List all users and all courier services, showing all possible combinations.**

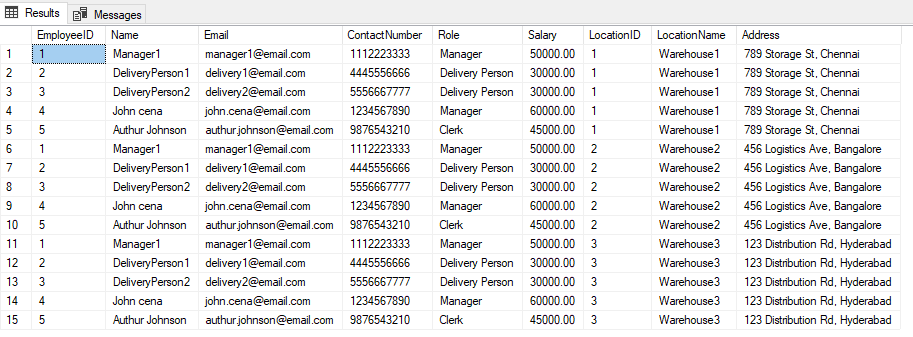
SELECT u.\*, cs.\*FROM Users u CROSS JOIN CourierServices cs;





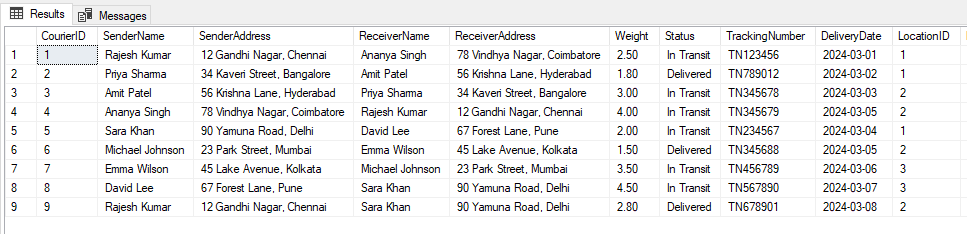
**37. List all employees and all locations, showing all possible combinations:**

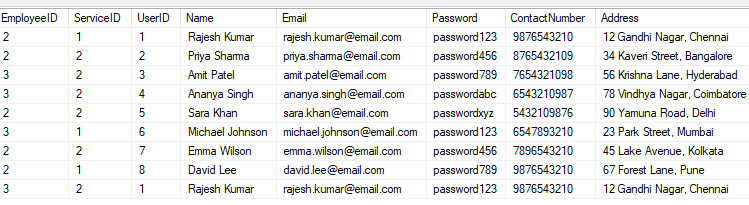
SELECT e.\*, l.\* FROM Employees e CROSS JOIN Locations l;



**38. Retrieve a list of couriers and their corresponding sender information (if available)**

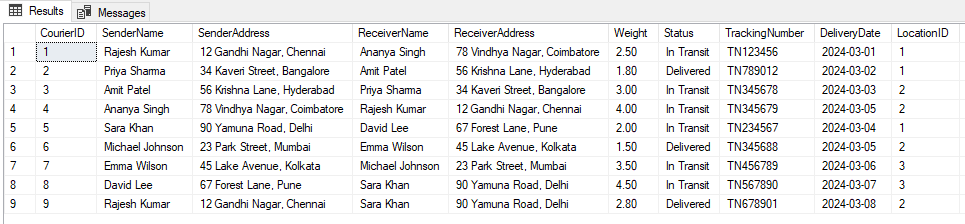
SELECT c.\*, u.\*FROM Couriers c LEFT JOIN Users u ON c.SenderName = u.Name;

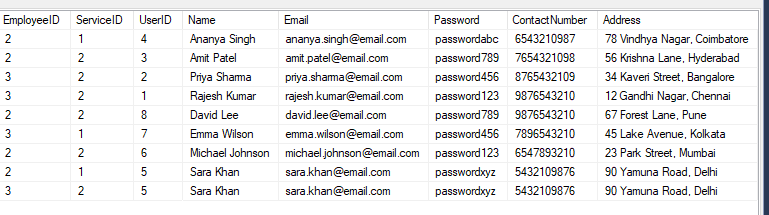




**39. Retrieve a list of couriers and their corresponding receiver information (if available):**

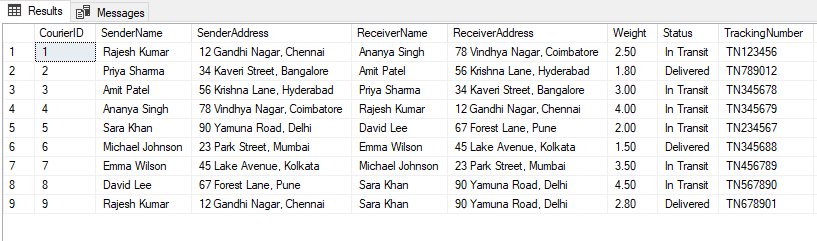
SELECT c.\*, u.\*FROM Couriers c LEFT JOIN Users u ON c.ReceiverName = u.Name;

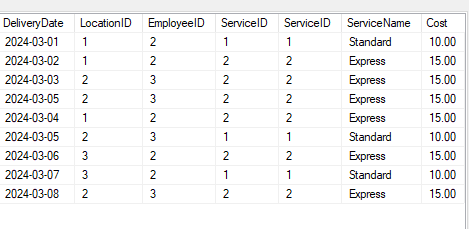




1. **Retrieve a list of couriers along with the courier service details (if available):**

SELECT c.\*, cs.\*FROM Couriers c LEFT JOIN CourierServices cs ON c.ServiceID = cs.ServiceID;

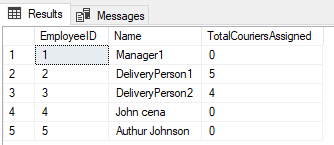




**41. Retrieve a list of employees and the number of couriers assigned to each employee:**

SELECT e.EmployeeID, e.Name, COUNT(c.CourierID) AS TotalCouriersAssigned FROM Employees e

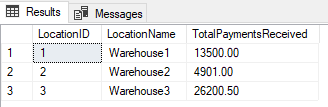
LEFT JOIN Couriers c ON e.EmployeeID = c.EmployeeID GROUP BY e.EmployeeID, e.Name;



**42. Retrieve a list of locations and the total payment amount received at each location:**

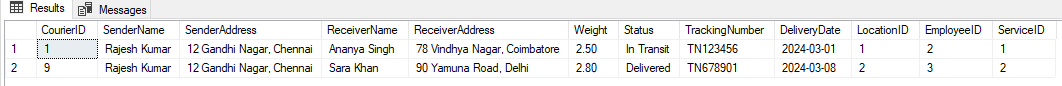
SELECT l.LocationID, l.LocationName, SUM(p.Amount) AS TotalPaymentsReceived

FROM Locations l LEFT JOIN Payments p ON l.LocationID = p.LocationID GROUP BY l.LocationID, l.LocationName;



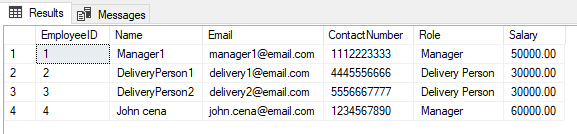
**43. Retrieve all couriers sent by the same sender (based on SenderName).**

SELECT c1.\*FROM Couriers c1 JOIN Couriers c2 ON c1.SenderName = c2.SenderName WHERE c1.CourierID <> c2.CourierID;;



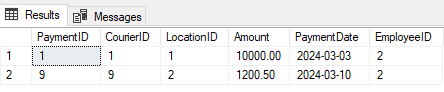
**44. List all employees who share the same role.**

SELECT e1.\*FROM Employees e1 JOIN Employees e2 ON e1.Role = e2.Role WHERE e1.EmployeeID <> e2.EmployeeID;



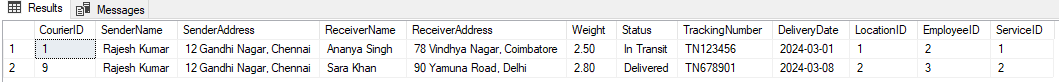
1. **Retrieve all payments made for couriers sent from the same location.**

SELECT Payments.\*FROM Payments JOIN Couriers ON Payments.CourierID = Couriers.CourierID WHERE SenderAddress LIKE '12 Gandhi Nagar, Chennai';



**46. Retrieve all couriers sent from the same location (based on SenderAddress).**

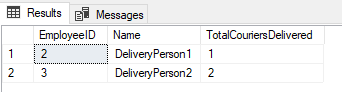
SELECT \*FROM Couriers WHERE SenderAddress LIKE '12 Gandhi Nagar, Chennai';



**47. List employees and the number of couriers they have delivered:**

SELECT e.EmployeeID, e.Name, COUNT(c.CourierID) AS TotalCouriersDelivered FROM Employees e

LEFT JOIN Couriers c ON e.EmployeeID = c.EmployeeID WHERE c.Status = 'Delivered' GROUP BY e.EmployeeID, e.Name;

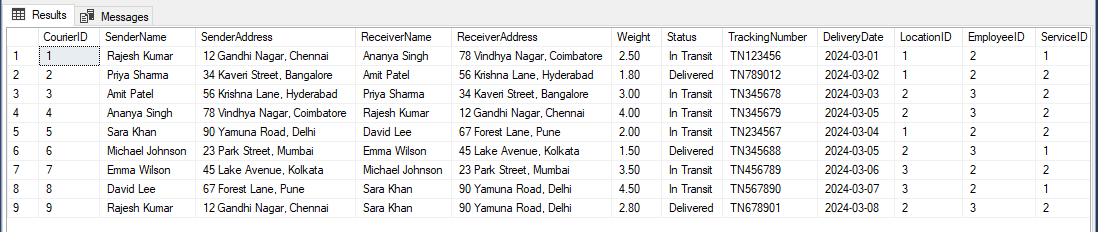


1. **Find couriers that were paid an amount greater than the cost of their respective**

**courier services**

SELECT c.\*FROM Couriers c JOIN Payments p ON c.CourierID = p.CourierID

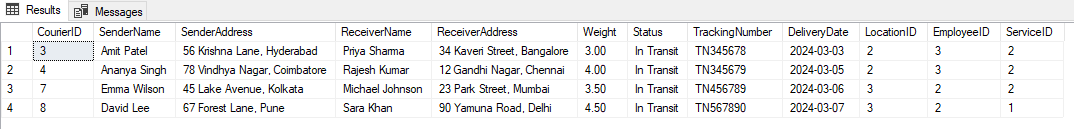
JOIN CourierServices cs ON c.ServiceID = cs.ServiceID WHERE p.Amount > cs.Cost;



**Scope: Inner Queries, Non Equi Joins, Equi joins,Exist,Any,All**

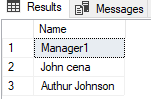
**49. Find couriers that have a weight greater than the average weight of all couriers**

SELECT \* FROM Couriers WHERE Weight > (SELECT AVG(Weight) FROM Couriers);



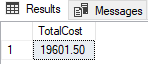
**50. Find the names of all employees who have a salary greater than the average salary:**

SELECT Name FROM Employees WHERE Salary > (SELECT AVG(Salary) FROM Employees);



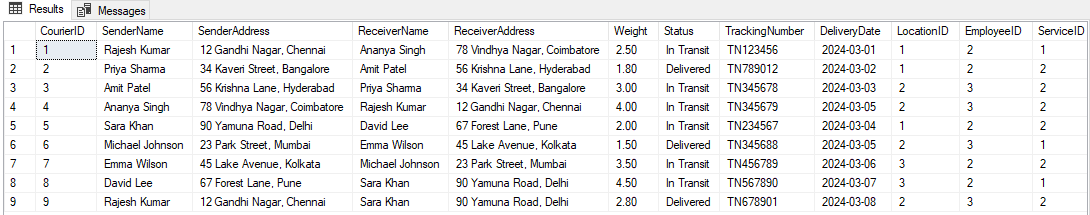
**51. Find the total cost of all courier services where the cost is less than the maximum cost**

SELECT SUM(Amount) AS TotalCost FROM Payments WHERE Amount < (SELECT MAX(Amount) FROM Payments);



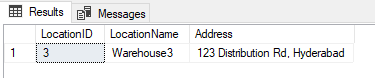
**52. Find all couriers that have been paid for**

SELECT c.\*FROM Couriers c WHERE EXISTS (SELECT 1 FROM Payments p WHERE p.CourierID = c.CourierID);



**53. Find the locations where the maximum payment amount was made**

SELECT l.\*FROM Locations l WHERE LocationID = (SELECT TOP 1 LocationID FROM Payments ORDER BY Amount DESC);



1. **Find all couriers whose weight is greater than the weight of all couriers sent by a specific sender (e.g., 'SenderName'):**

SELECT c.\*FROM Couriers c WHERE Weight > ALL (SELECT Weight FROM Couriers WHERE SenderName = 'Rajesh Kumar');

