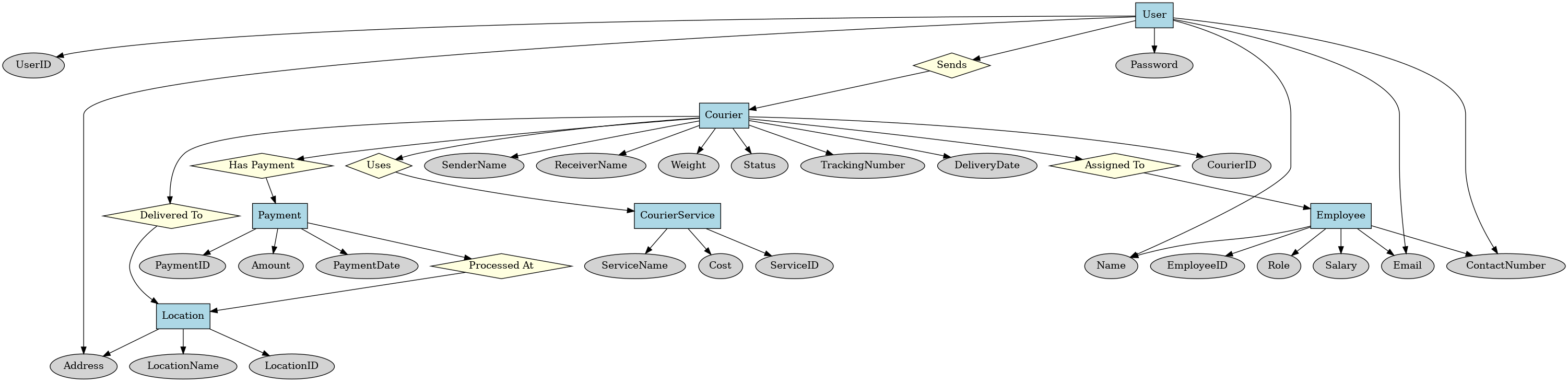
**COURIER MANAGEMENT SYSTEM**

***Submitted by – S Lakshmi Srujana***

**Task 1 – Database Design**

** *ER Diagram***

**Create tables :-**

***User table:***

CREATE TABLE User (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(255) NOT NULL,

Email VARCHAR(255) UNIQUE NOT NULL,

Password VARCHAR(255) NOT NULL,

ContactNumber VARCHAR(20) NOT NULL,

Address TEXT NOT NULL

);

***Courier table:***

CREATE TABLE Courier (

CourierID INT PRIMARY KEY AUTO\_INCREMENT,

SenderName VARCHAR(255) NOT NULL,

SenderAddress TEXT NOT NULL,

ReceiverName VARCHAR(255) NOT NULL,

ReceiverAddress TEXT NOT NULL,

Weight DECIMAL(5,2) NOT NULL,

Status VARCHAR(50) NOT NULL CHECK (Status IN ('Processing', 'In Transit', 'Delivered', 'Cancelled')),

TrackingNumber VARCHAR(20) UNIQUE NOT NULL,

DeliveryDate DATE

);

***Courier Services Table:***

CREATE TABLE CourierServices (

ServiceID INT PRIMARY KEY AUTO\_INCREMENT,

ServiceName VARCHAR(100) NOT NULL,

Cost DECIMAL(8,2) NOT NULL

);

***Employee Table:***

CREATE TABLE Employee (

EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(255) NOT NULL,

Email VARCHAR(255) UNIQUE NOT NULL,

ContactNumber VARCHAR(20) NOT NULL,

Role VARCHAR(50) NOT NULL CHECK (Role IN ('Driver', 'Manager', 'Dispatcher')),

Salary DECIMAL(10,2) NOT NULL

);

***Location table:***

CREATE TABLE Location (

LocationID INT PRIMARY KEY AUTO\_INCREMENT,

LocationName VARCHAR(100) NOT NULL,

Address TEXT NOT NULL

);

***Payment table:***

CREATE TABLE Payment (

PaymentID INT PRIMARY KEY AUTO\_INCREMENT,

CourierID INT NOT NULL,

LocationID INT NOT NULL,

Amount DECIMAL(10,2) NOT NULL,

PaymentDate DATE NOT NULL,

FOREIGN KEY (CourierID) REFERENCES Courier(CourierID) ON DELETE CASCADE,

FOREIGN KEY (LocationID) REFERENCES Location(LocationID) ON DELETE CASCADE );

**Insert Sample Data:-**

***User table:***

INSERT INTO User (Name, Email, Password, ContactNumber, Address)

VALUES

('Alice Johnson', 'alice@email.com', 'hashedpassword1', '123-456-7890', '123 Main St, NY'),

('Bob Smith', 'bob@email.com', 'hashedpassword2', '987-654-3210', '456 Elm St, CA');

***Courier table:***

INSERT INTO Courier (SenderName, SenderAddress, ReceiverName, ReceiverAddress, Weight, Status, TrackingNumber, DeliveryDate) VALUES

('Alice Johnson', '123 Main St, NY', 'Bob Smith', '456 Elm St, CA', 2.5, 'Processing', 'TRK123456', '2025-04-01'),

('Bob Smith', '456 Elm St, CA', 'Alice Johnson', '123 Main St, NY', 1.8, 'In Transit', 'TRK654321', '2025-04-02');

***Courier Services Table:***

INSERT INTO CourierServices (ServiceName, Cost)

VALUES

('Express Delivery', 20.00),

('Standard Delivery', 10.00),

('Overnight Shipping', 30.00);

***Employee Table:***

INSERT INTO Employee (Name, Email, ContactNumber, Role, Salary)

VALUES

('John Doe', 'john@email.com', '123-123-1234', 'Driver', 50000),

('Jane Smith', 'jane@email.com', '456-456-4567', 'Manager', 70000);

***Location table:***

INSERT INTO Location (LocationName, Address)

VALUES

('New York Hub', 'NYC, NY'),

('California Hub', 'Los Angeles, CA');

***Payment table:***

INSERT INTO Payment (CourierID, LocationID, Amount, PaymentDate)

VALUES

(1, 1, 15.00, '2025-03-20'),

(2, 2, 25.00, '2025-03-21');

**Task 2 – SELECT, WHERE**

(1). SELECT \* FROM User;

(2). SELECT \* FROM Courier WHERE SenderName = 'Alice Johnson';

(3). SELECT \* FROM Courier;

(4). SELECT \* FROM Courier WHERE TrackingNumber = 'TRK123456';

(5). SELECT \* FROM Courier WHERE Status = 'Delivered' AND CourierID = 1;

(6). SELECT \* FROM Courier WHERE Status != 'Delivered';

(7). SELECT \* FROM Courier WHERE DeliveryDate = CURDATE();

(8). SELECT \* FROM Courier WHERE Status = 'Processing';

(9). SELECT CourierID, COUNT(\*) AS TotalPackages FROM Courier

GROUP BY CourierID;

(10). SELECT CourierID, AVG(DATEDIFF(DeliveryDate, CURDATE()))

AS AvgDeliveryTime FROM Courier WHERE Status = 'Delivered'

GROUP BY CourierID;

(11). SELECT \* FROM Courier WHERE Weight BETWEEN 1.0 AND 5.0;

(12). SELECT \* FROM Employee WHERE Name LIKE '%John%';

(13). SELECT c.\* FROM Courier c JOIN Payment p ON c.CourierID = p.CourierID

WHERE p.Amount > 50;

**Task 3: SQL Queries (GROUP BY, Aggregate Functions, HAVING, ORDER BY, WHERE)**

(14). // Not possible to determine with the given information provided //

(15). SELECT LocationID, SUM(Amount) AS TotalRevenue FROM Payment

GROUP BY LocationID;

(16). SELECT LocationID, COUNT(CourierID) AS TotalDelivered FROM Courier

WHERE Status = 'Delivered' GROUP BY LocationID;

(17). SELECT CourierID, AVG(DATEDIFF(DeliveryDate, CURDATE()))

AS AvgDeliveryTime FROM Courier WHERE Status = 'Delivered'

GROUP BY CourierID ORDER BY AvgDeliveryTime DESC LIMIT 1;

(18). SELECT LocationID, SUM(Amount) AS TotalPayments FROM Payment

GROUP BY LocationID HAVING SUM(Amount) < 500;

(19). SELECT LocationID, SUM(Amount) AS TotalPayments FROM Payment

GROUP BY LocationID ORDER BY TotalPayments DESC;

(20). SELECT CourierID, SUM(Amount) AS TotalPayments FROM Payment

WHERE LocationID = X GROUP BY CourierID HAVING SUM(Amount) > 1000;

(21). SELECT CourierID, SUM(Amount) AS TotalPayments FROM Payment

WHERE PaymentDate > 'YYYY-MM-DD' GROUP BY CourierID

HAVING SUM(Amount) > 1000;

(22). SELECT LocationID, SUM(Amount) AS TotalPayments FROM Payment

WHERE PaymentDate < 'YYYY-MM-DD' GROUP BY LocationID

HAVING SUM(Amount) > 5000;

**Task 4: SQL Queries (JOINs - Inner Join, Outer Join, Cross Join, etc.)**

(23). SELECT p.\*, c.TrackingNumber, c.Status FROM Payment p INNER JOIN Courier c

ON p.CourierID = c.CourierID;

(24). SELECT p.\*, l.LocationName FROM Payment p INNER JOIN Location l

ON p.LocationID = l.LocationID;

(25). SELECT p.\*, c.TrackingNumber, c.Status, l.LocationName FROM Payment p

INNER JOIN Courier c ON p.CourierID = c.CourierID

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

(26). SELECT p.PaymentID, p.Amount, p.PaymentDate, c.TrackingNumber, c.Status,

c.SenderName, c.ReceiverName FROM Payment p INNER JOIN Courier c

ON p.CourierID = c.CourierID;

(27). SELECT c.CourierID, c.TrackingNumber, SUM(p.Amount) AS TotalPayments

FROM Payment p INNER JOIN Courier c ON p.CourierID = c.CourierID

GROUP BY c.CourierID, c.TrackingNumber;

(28). SELECT \* FROM Payment WHERE PaymentDate = '2025-03-21';

(29). SELECT p.\*, c.\* FROM Payment p LEFT JOIN Courier c

ON p.CourierID = c.CourierID;

(30). SELECT p.\*, l.LocationName FROM Payment p LEFT JOIN Location l

ON p.LocationID = l.LocationID;

(31). SELECT c.CourierID, c.TrackingNumber, SUM(p.Amount) AS TotalAmount

FROM Payment p INNER JOIN Courier c ON p.CourierID = c.CourierID

GROUP BY c.CourierID, c.TrackingNumber;

(32). SELECT \* FROM Payment WHERE PaymentDate BETWEEN '2025-03-01'

AND '2025-03-31';

(33). SELECT u.\*, c.CourierID, c.TrackingNumber FROM User u LEFT JOIN Courier c

ON u.UserID = c.CourierID;

(34). SELECT c.\*, s.ServiceName, s.Cost FROM Courier c LEFT JOIN CourierServices s

ON c.CourierID = s.ServiceID;

(35). SELECT e.\*, p.Amount, p.PaymentDate FROM Employee e LEFT JOIN Payment p

ON e.EmployeeID = p.PaymentID;

(36). SELECT \* FROM User CROSS JOIN CourierServices;

(37). SELECT \* FROM Employee CROSS JOIN Location;

(38). SELECT c.\*, u.Name AS SenderName, u.ContactNumber FROM Courier c

LEFT JOIN User u ON c.SenderName = u.Name;

(39). SELECT c.\*, u.Name AS ReceiverName, u.ContactNumber FROM Courier c

LEFT JOIN User u ON c.ReceiverName = u.Name;

(40). SELECT c.\*, s.ServiceName, s.Cost FROM Courier c LEFT JOIN CourierServices s

ON c.CourierID = s.ServiceID;

(41). // Not possible to determine with the given information provided //

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

FROM Location l LEFT JOIN Payment p ON l.LocationID = p.LocationID

GROUP BY l.LocationID, l.LocationName;

(43). SELECT \* FROM Courier WHERE SenderName = 'Alice Johnson';

(44). SELECT Role, GROUP\_CONCAT(Name) AS Employees FROM Employee

GROUP BY Role;

(45). SELECT p.\*, c.SenderAddress

FROM Payment p

JOIN Courier c ON p.CourierID = c.CourierID

WHERE c.SenderAddress = '123 Main St, NY';

(46). SELECT \* FROM Courier

WHERE SenderAddress = '123 Main St, NY';

(47). // Not possible to determine with the given information provided //

(48). SELECT c.CourierID, c.TrackingNumber, s.ServiceName, s.Cost AS ServiceCost,

SUM(p.Amount) AS TotalPaid FROM Courier c

JOIN Payment p ON c.CourierID = p.CourierID

JOIN CourierServices s ON c.CourierID = s.ServiceID

GROUP BY c.CourierID, c.TrackingNumber, s.ServiceName, s.Cost

HAVING SUM(p.Amount) > s.Cost;

**Task 5: SQL Queries (Nested Queries, EXISTS, ANY, ALL, etc.)**

(49). SELECT \* FROM Courier WHERE Weight > (SELECT AVG(Weight)

FROM Courier);

(50). SELECT Name, Salary FROM Employee

WHERE Salary > (SELECT AVG(Salary) FROM Employee);

(51). SELECT SUM(Cost) AS TotalCost FROM CourierServices

WHERE Cost < (SELECT MAX(Cost) FROM CourierServices);

(52). SELECT \* FROM Courier WHERE CourierID

IN (SELECT DISTINCT CourierID FROM Payment);

(53). SELECT LocationID, SUM(Amount) AS TotalPayments

FROM Payment

GROUP BY LocationID

HAVING SUM(Amount) = (SELECT MAX(TotalPayments)

FROM (SELECT SUM(Amount) AS TotalPayments

FROM Payment GROUP BY LocationID) AS PaymentSummary);

(54). SELECT \* FROM Courier

WHERE Weight > ALL (SELECT Weight FROM Courier

WHERE SenderName = 'Alice Johnson'

);