COURIER MANAGEMENT SYSTEM

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. Requirements: • Define the Database Schema • Create SQL tables for entities such as User, Courier, Employee, Location,Payment • Define relationships between these tables (one-to-many, many-to-many, etc.). • Populate Sample Data • Insert sample data into the tables to simulate real-world scenarios.

User Table:

```
User
(UserID INT PRIMARY KEY,
Name VARCHAR(255),
Email VARCHAR(255) UNIQUE,
Password VARCHAR(255),
ContactNumber VARCHAR(20),
Address TEXT
);
Courier
(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);
CourierServices
(ServiceID INT PRIMARY KEY,
ServiceName VARCHAR(100),
Cost DECIMAL(8, 2));
```

Employee Table:

(EmployeeID INT PRIMARY KEY,

Name VARCHAR(255),

Email VARCHAR(255) UNIQUE,

ContactNumber VARCHAR(20),

Role VARCHAR(50),

Salary DECIMAL(10, 2));

Location Table:

(LocationID INT PRIMARY KEY,

LocationName VARCHAR(100),

Address TEXT);

Payment Table:

(PaymentID INT PRIMARY KEY,

CourierID INT,

LocationId INT,

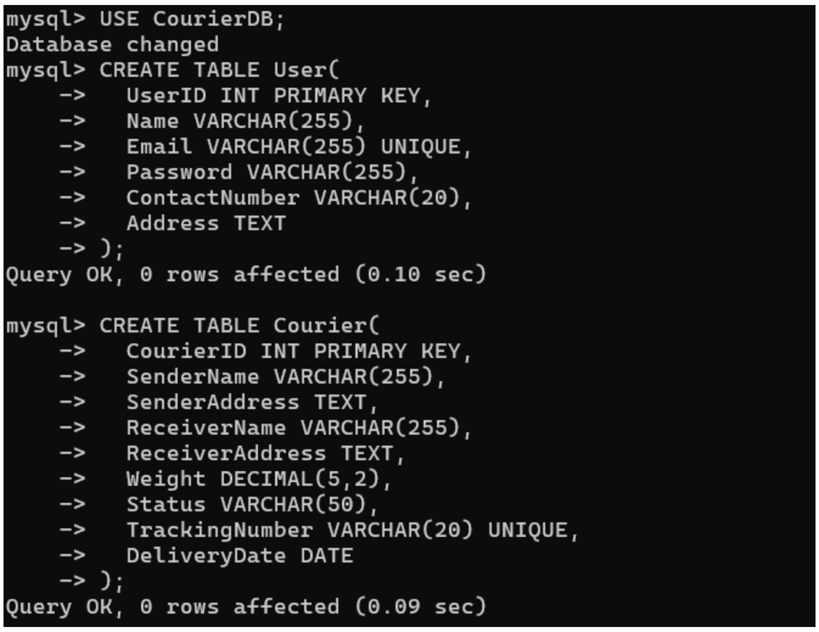
Amount DECIMAL(10, 2),

PaymentDate DATE,

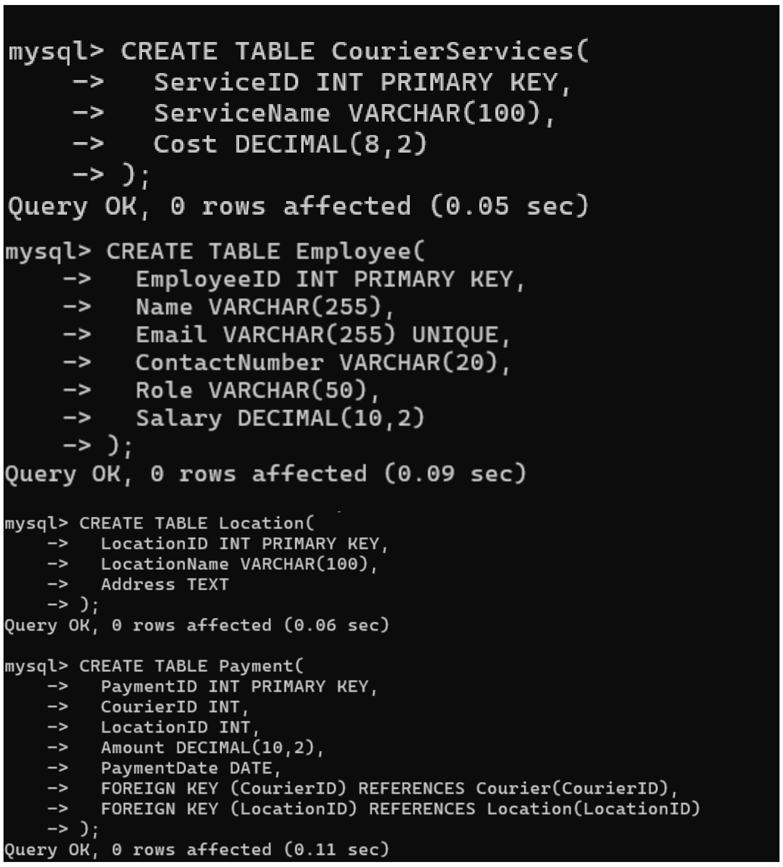
FOREIGN KEY (CourierID) REFERENCES Couriers(CourierID),

FOREIGN KEY (LocationID) REFERENCES Location(LocationID));

CREATING TABLES

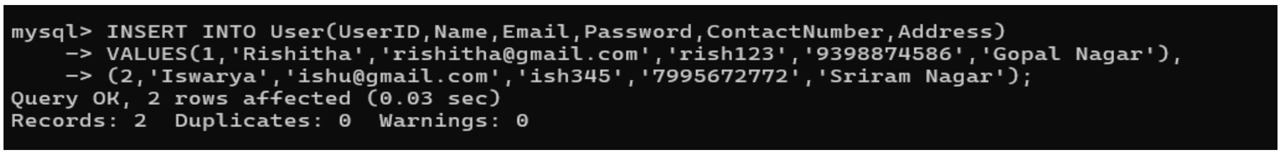


```
mysql> USE CourierDB;
Database changed
mysql> CREATE TABLE User(
    ->   UserID INT PRIMARY KEY,
    ->   Name VARCHAR(255),
    ->   Email VARCHAR(255) UNIQUE,
    ->   Password VARCHAR(255),
    ->   ContactNumber VARCHAR(20),
    ->   Address TEXT
    -> );
Query OK, 0 rows affected (0.10 sec)

mysql> CREATE TABLE Courier(
    ->   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
    -> );
Query OK, 0 rows affected (0.09 sec)
```

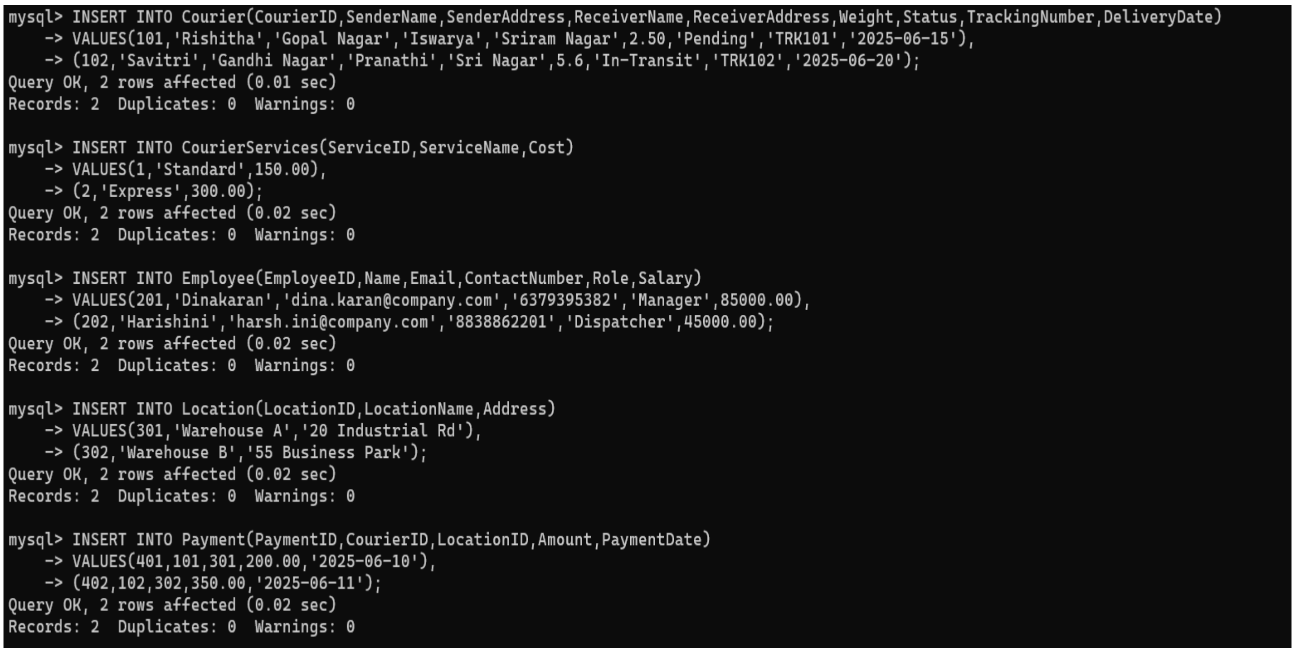


```
mysql> CREATE TABLE CourierServices(
    ->   ServiceID INT PRIMARY KEY,
    ->   ServiceName VARCHAR(100),
    ->   Cost DECIMAL(8,2)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE Employee(
    ->   EmployeeID INT PRIMARY KEY,
    ->   Name VARCHAR(255),
    ->   Email VARCHAR(255) UNIQUE,
    ->   ContactNumber VARCHAR(20),
    ->   Role VARCHAR(50),
    ->   Salary DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.09 sec)

mysql> CREATE TABLE Location(
    ->   LocationID INT PRIMARY KEY,
    ->   LocationName VARCHAR(100),
    ->   Address TEXT
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> CREATE TABLE Payment(
    ->   PaymentID INT PRIMARY KEY,
    ->   CourierID INT,
    ->   LocationID INT,
    ->   Amount DECIMAL(10,2),
    ->   PaymentDate DATE,
    ->   FOREIGN KEY (CourierID) REFERENCES Courier(CourierID),
    ->   FOREIGN KEY (LocationID) REFERENCES Location(LocationID)
    -> );
Query OK, 0 rows affected (0.11 sec)
```

INSERTING VALUES:

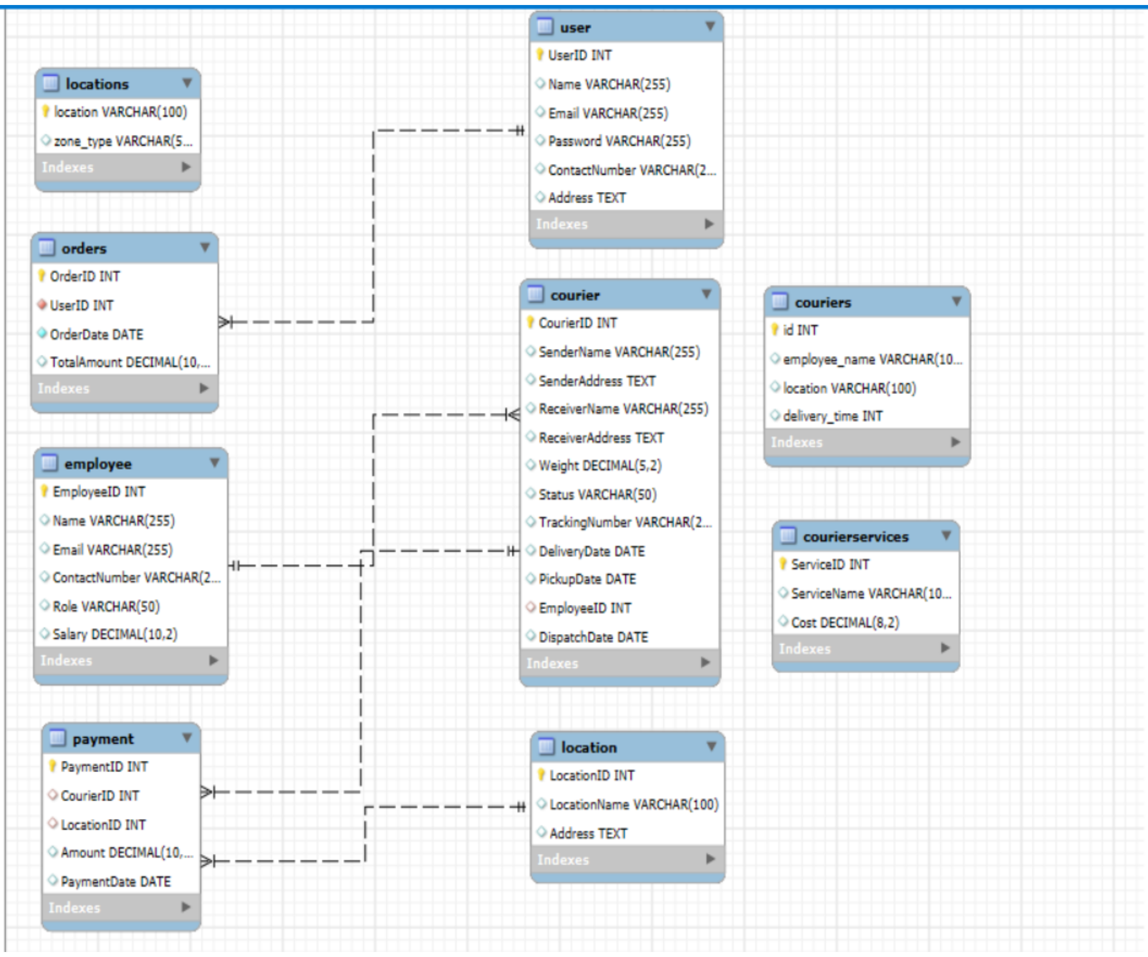


```
mysql> INSERT INTO User(UserID,Name,Email,Password,ContactNumber,Address)
    -> VALUES(1,'Rishitha','rishitha@gmail.com','rish123','9398874586','Gopal Nagar'),
    -> (2,'Iswarya','ishu@gmail.com','ish345','7995672772','Sriram Nagar');
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0
```



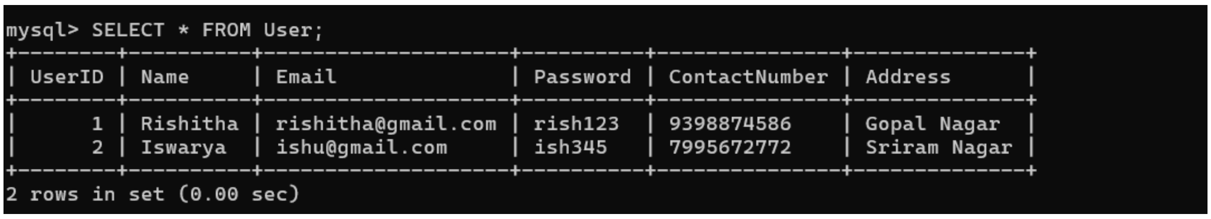
```
mysql> INSERT INTO Courier(CourierID,SenderName,SenderAddress,ReceiverName,ReceiverAddress,Weight,Status,TrackingNumber,DeliveryDate)
    -> VALUES(101,'Rishitha','Gopal Nagar','Iswarya','Sriram Nagar',2.50,'Pending','TRK101','2025-06-15'),
    -> (102,'Savitri','Gandhi Nagar','Pranathi','Sri Nagar',5.6,'In-Transit','TRK102','2025-06-20');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO CourierServices(ServiceID,ServiceName,Cost)
    -> VALUES(1,'Standard',150.00),
    -> (2,'Express',300.00);
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Employee(EmployeeID,Name,Email,ContactNumber,Role,Salary)
    -> VALUES(201,'Dinakaran','dina.karan@company.com','6379395382','Manager',85000.00),
    -> (202,'Harishini','harsh.ini@company.com','8838862201','Dispatcher',45000.00);
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Location(LocationID,LocationName,Address)
    -> VALUES(301,'Warehouse A','20 Industrial Rd'),
    -> (302,'Warehouse B','55 Business Park');
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Payment(PaymentID,CourierID,LocationID,Amount,PaymentDate)
    -> VALUES(401,101,301,200.00,'2025-06-10'),
    -> (402,102,302,350.00,'2025-06-11');
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

SQL SCHEMA :


locations
location VARCHAR(100)
zone_type VARCHAR(5...
Indexes
orders
OrderID INT
UserID INT
OrderDate DATE
TotalAmount DECIMAL(10,...
Indexes
employee
EmployeeID INT
Name VARCHAR(255)
Email VARCHAR(255)
ContactNumber VARCHAR(2...
Role VARCHAR(50)
Salary DECIMAL(10,2)
Indexes
payment
PaymentID INT
CourierID INT
LocationID INT
Amount DECIMAL(10,...
PaymentDate DATE
Indexes
user
UserID INT
Name VARCHAR(255)
Email VARCHAR(255)
Password VARCHAR(255)
ContactNumber VARCHAR(2...
Address TEXT
Indexes
courier
CourierID INT
SenderName VARCHAR(255)
SenderAddress TEXT
ReceiverName VARCHAR(255)
ReceiverAddress TEXT
Weight DECIMAL(5,2)
Status VARCHAR(50)
TrackingNumber VARCHAR(2...
DeliveryDate DATE
PickupDate DATE
EmployeeID INT
DispatchDate DATE
Indexes
location
LocationID INT
LocationName VARCHAR(100)
Address TEXT
Indexes
couriers
id INT
employee_name VARCHAR(10...
location VARCHAR(100)
delivery_time INT
Indexes
courierservices
ServiceID INT
ServiceName VARCHAR(10...
Cost DECIMAL(8,2)
Indexes

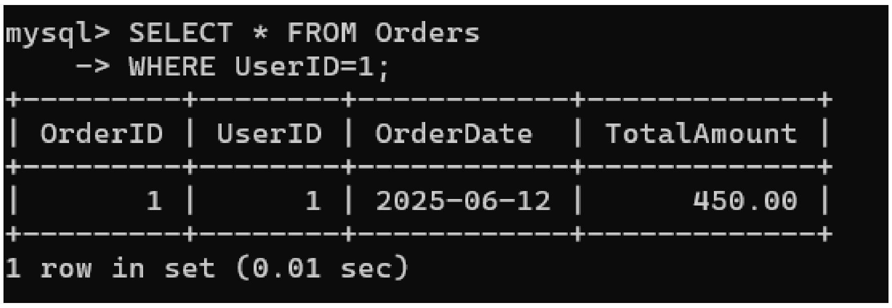
Task 2: Select,Where

1.List all customers:



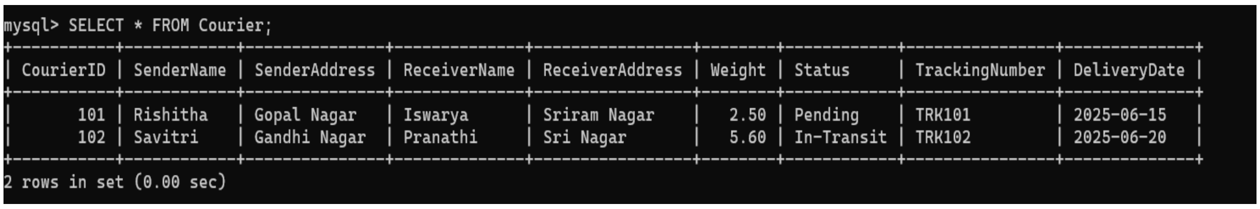
```
mysql> SELECT * FROM User;
+--------+----------+--------------------+----------+---------------+--------------+
| UserID | Name     | Email              | Password | ContactNumber | Address      |
+--------+----------+--------------------+----------+---------------+--------------+
|      1 | Rishitha | rishitha@gmail.com | rish123  | 9398874586    | Gopal Nagar  |
|      2 | Iswarya  | ishu@gmail.com     | ish345   | 7995672772    | Sriram Nagar |
+--------+----------+--------------------+----------+---------------+--------------+
2 rows in set (0.00 sec)
```

2.List all orders for a specific customer:



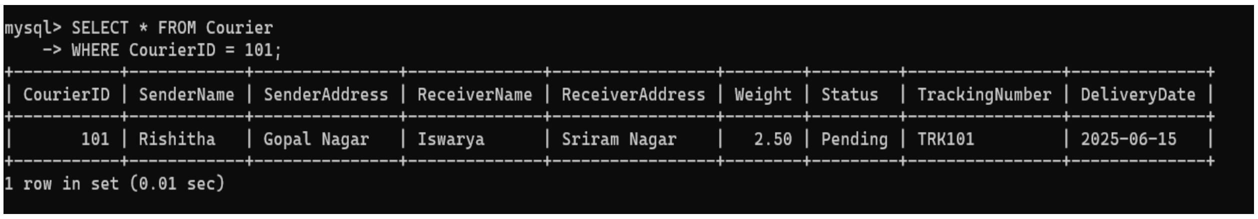
```
mysql> SELECT * FROM Orders
    -> WHERE UserID=1;
+---------+--------+------------+-------------+
| OrderID | UserID | OrderDate  | TotalAmount |
+---------+--------+------------+-------------+
|       1 |      1 | 2025-06-12 |      450.00 |
+---------+--------+------------+-------------+
1 row in set (0.01 sec)
```

3.List all couriers:



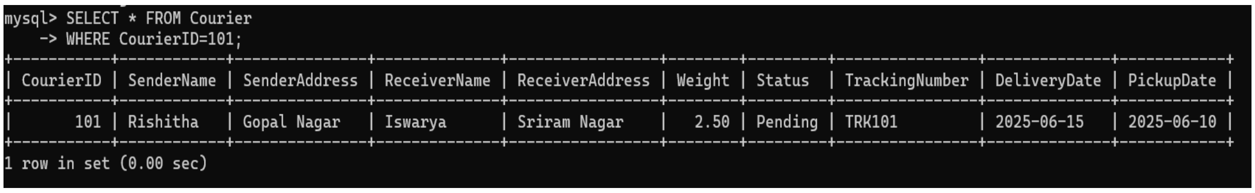
```
mysql> SELECT * FROM Courier;
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status     | TrackingNumber | DeliveryDate |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
|       101 | Rishitha   | Gopal Nagar   | Iswarya      | Sriram Nagar    |   2.50 | Pending    | TRK101         | 2025-06-15   |
|       102 | Savitri    | Gandhi Nagar  | Pranathi     | Sri Nagar       |   5.60 | In-Transit | TRK102         | 2025-06-20   |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
2 rows in set (0.00 sec)
```

4.List all packages for a specific order:



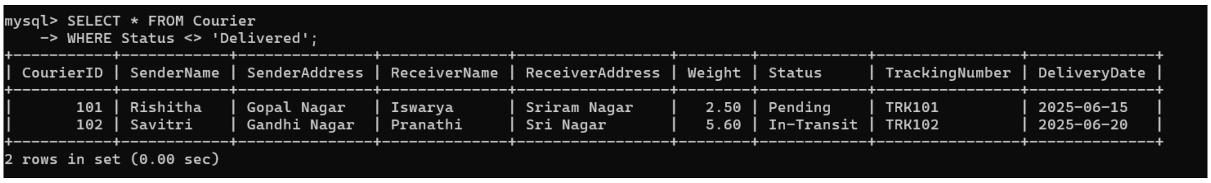
```
mysql> SELECT * FROM Courier
    -> WHERE CourierID = 101;
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status  | TrackingNumber | DeliveryDate |
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+
|       101 | Rishitha   | Gopal Nagar   | Iswarya      | Sriram Nagar    |   2.50 | Pending | TRK101         | 2025-06-15   |
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+
1 row in set (0.01 sec)
```

5.List all deliveries for a specific courier:



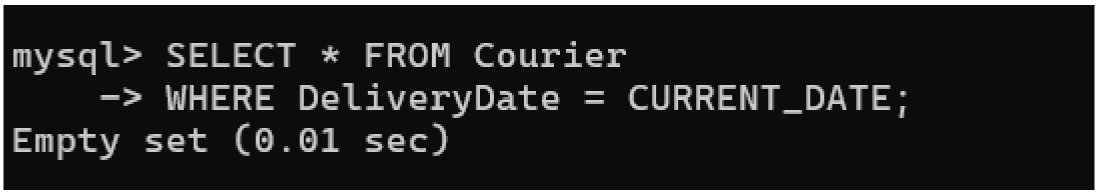
```
mysql> SELECT * FROM Courier
    -> WHERE CourierID=101;
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+------------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status  | TrackingNumber | DeliveryDate | PickupDate |
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+------------+
|       101 | Rishitha   | Gopal Nagar   | Iswarya      | Sriram Nagar    |   2.50 | Pending | TRK101         | 2025-06-15   | 2025-06-10 |
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+------------+
1 row in set (0.00 sec)
```

6.List all undelivered packages:



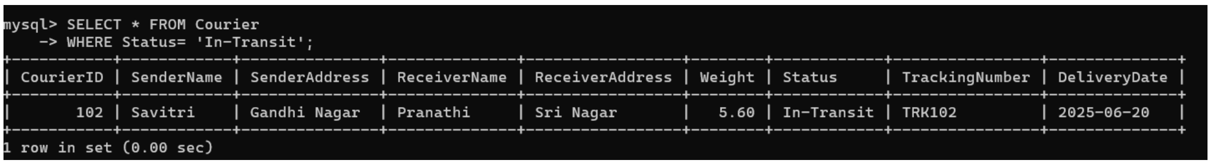
```
mysql> SELECT * FROM Courier
    -> WHERE Status <> 'Delivered';
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status     | TrackingNumber | DeliveryDate |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
|       101 | Rishitha   | Gopal Nagar   | Iswarya      | Sriram Nagar    |   2.50 | Pending    | TRK101         | 2025-06-15   |
|       102 | Savitri    | Gandhi Nagar  | Pranathi     | Sri Nagar       |   5.60 | In-Transit | TRK102         | 2025-06-20   |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
2 rows in set (0.00 sec)
```

7.List all packages that are scheduled for delivery today:



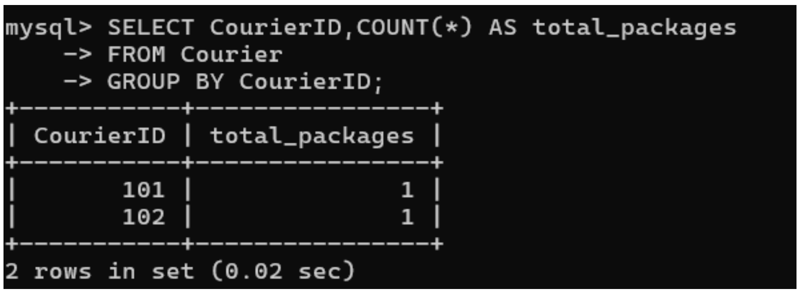
```
mysql> SELECT * FROM Courier
    -> WHERE DeliveryDate = CURRENT_DATE;
Empty set (0.01 sec)
```

8.List all packages with a specific status:



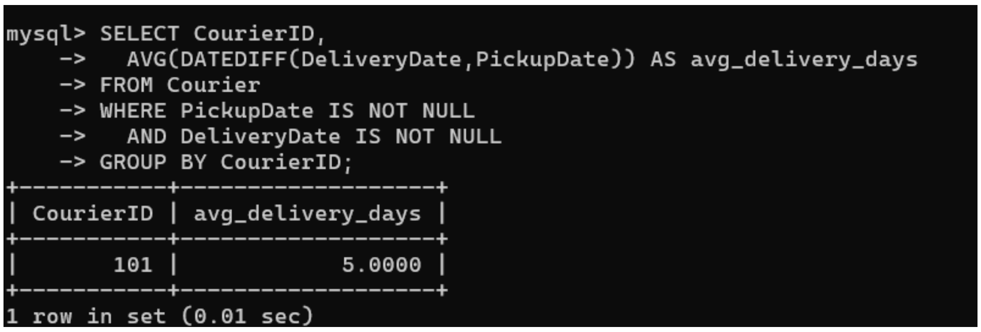
```
mysql> SELECT * FROM Courier
    -> WHERE Status= 'In-Transit';
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status     | TrackingNumber | DeliveryDate |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
|       102 | Savitri    | Gandhi Nagar  | Pranathi     | Sri Nagar       |   5.60 | In-Transit | TRK102         | 2025-06-20   |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+
1 row in set (0.00 sec)
```

9.Calculate the total number of packages for each courier.



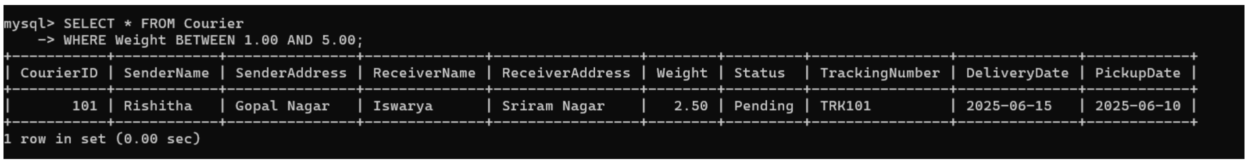
```
mysql> SELECT CourierID,COUNT(*) AS total_packages
    -> FROM Courier
    -> GROUP BY CourierID;
+-----------+----------------+
| CourierID | total_packages |
+-----------+----------------+
|       101 |              1 |
|       102 |              1 |
+-----------+----------------+
2 rows in set (0.02 sec)
```

10.Find the average delivery time for each courier



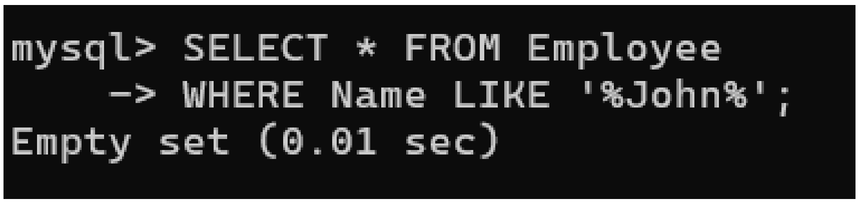
```
mysql> SELECT CourierID,
    ->   AVG(DATEDIFF(DeliveryDate,PickupDate)) AS avg_delivery_days
    -> FROM Courier
    -> WHERE PickupDate IS NOT NULL
    ->   AND DeliveryDate IS NOT NULL
    -> GROUP BY CourierID;
+-----------+-------------------+
| CourierID | avg_delivery_days |
+-----------+-------------------+
|       101 |            5.0000 |
+-----------+-------------------+
1 row in set (0.01 sec)
```

11.List all packages with a specific weight range:



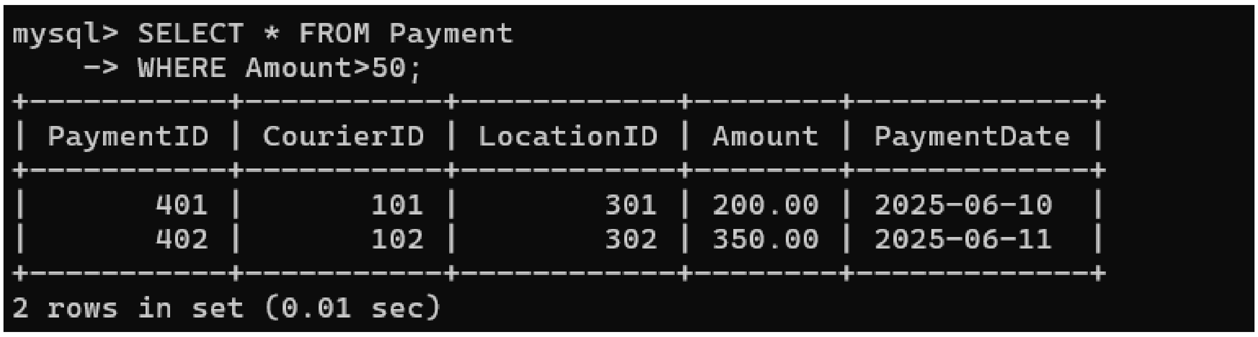
```
mysql> SELECT * FROM Courier
    -> WHERE Weight BETWEEN 1.00 AND 5.00;
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+------------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status  | TrackingNumber | DeliveryDate | PickupDate |
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+------------+
|       101 | Rishitha   | Gopal Nagar   | Iswarya      | Sriram Nagar    |   2.50 | Pending | TRK101         | 2025-06-15   | 2025-06-10 |
+-----------+------------+---------------+--------------+-----------------+--------+---------+----------------+--------------+------------+
1 row in set (0.00 sec)
```

12.Retrieve employees whose names contain 'John'



```
mysql> SELECT * FROM Employee
    -> WHERE Name LIKE '%John%';
Empty set (0.01 sec)
```

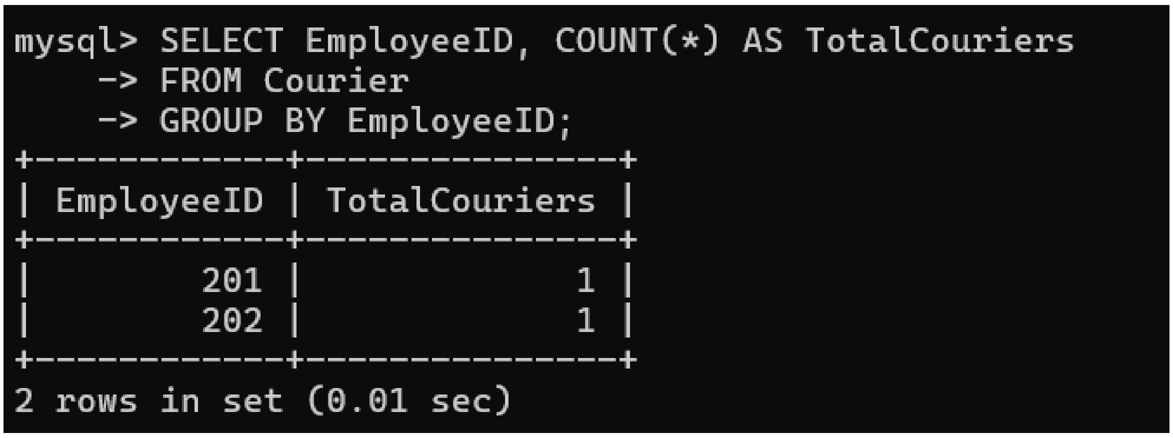
13.Retrieve all courier records with payments greater than $50.



```
mysql> SELECT * FROM Payment
    -> WHERE Amount>50;
+-----------+-----------+------------+--------+-------------+
| PaymentID | CourierID | LocationID | Amount | PaymentDate |
+-----------+-----------+------------+--------+-------------+
|       401 |       101 |        301 | 200.00 | 2025-06-10  |
|       402 |       102 |        302 | 350.00 | 2025-06-11  |
+-----------+-----------+------------+--------+-------------+
2 rows in set (0.01 sec)
```

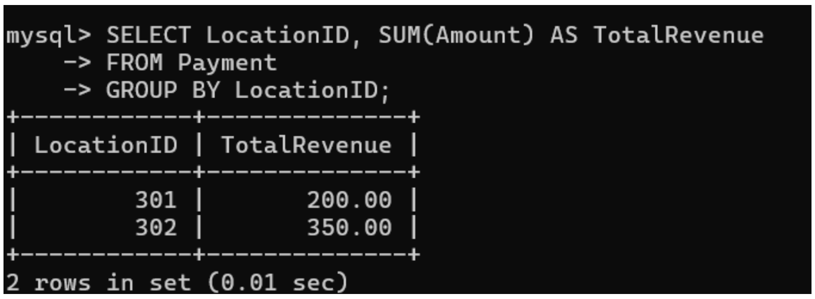
Task 3: GroupBy, Aggregate Functions, Having, Order By, where

14.Find the total number of couriers handled by each employee.



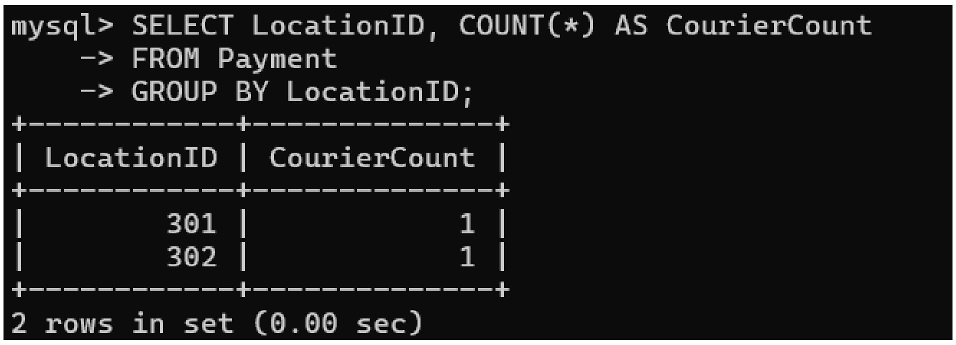
```
mysql> SELECT EmployeeID, COUNT(*) AS TotalCouriers
    -> FROM Courier
    -> GROUP BY EmployeeID;
+------------+---------------+
| EmployeeID | TotalCouriers |
+------------+---------------+
|        201 |             1 |
|        202 |             1 |
+------------+---------------+
2 rows in set (0.01 sec)
```

15.Calculate the total revenue generated by each location



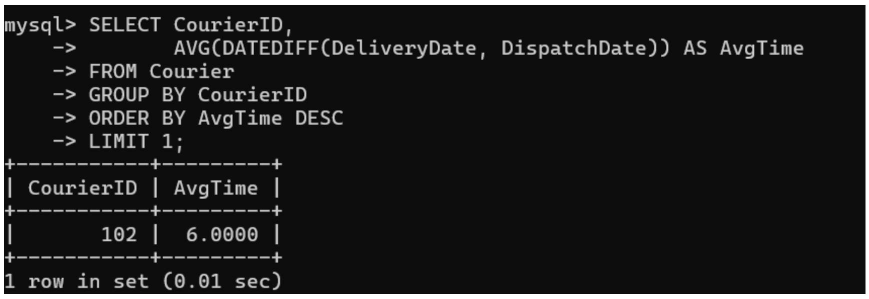
```
mysql> SELECT LocationID, SUM(Amount) AS TotalRevenue
    -> FROM Payment
    -> GROUP BY LocationID;
+------------+--------------+
| LocationID | TotalRevenue |
+------------+--------------+
|        301 |       200.00 |
|        302 |       350.00 |
+------------+--------------+
2 rows in set (0.01 sec)
```

16.Find the total number of couriers delivered to each location



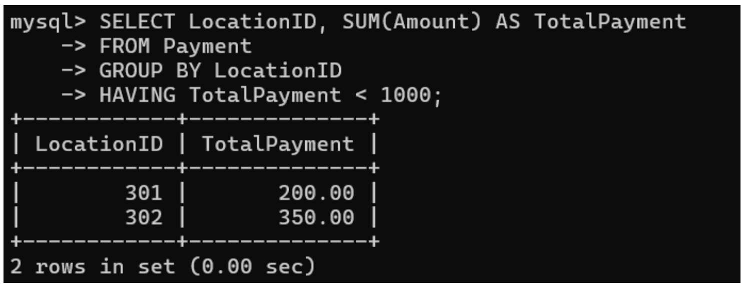
```
mysql> SELECT LocationID, COUNT(*) AS CourierCount
    -> FROM Payment
    -> GROUP BY LocationID;
+------------+--------------+
| LocationID | CourierCount |
+------------+--------------+
|        301 |            1 |
|        302 |            1 |
+------------+--------------+
2 rows in set (0.00 sec)
```

17.Find the courier with the highest average delivery time:



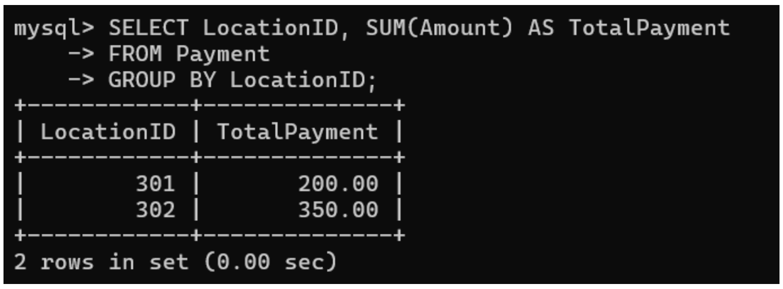
```
mysql> SELECT CourierID,
    ->        AVG(DATEDIFF(DeliveryDate, DispatchDate)) AS AvgTime
    -> FROM Courier
    -> GROUP BY CourierID
    -> ORDER BY AvgTime DESC
    -> LIMIT 1;
+-----------+---------+
| CourierID | AvgTime |
+-----------+---------+
|       102 |  6.0000 |
+-----------+---------+
1 row in set (0.01 sec)
```

18.Find Locations with Total Payments Less Than a Certain Amount



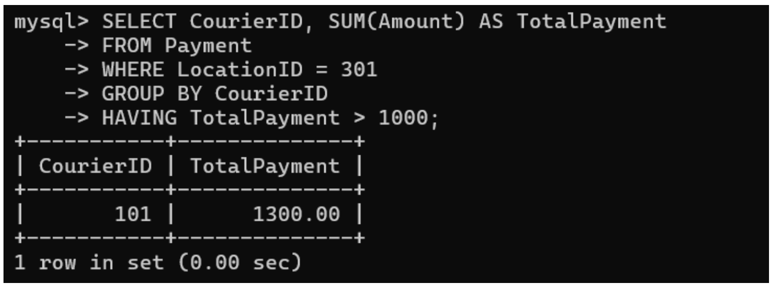
```
mysql> SELECT LocationID, SUM(Amount) AS TotalPayment
    -> FROM Payment
    -> GROUP BY LocationID
    -> HAVING TotalPayment < 1000;
+------------+--------------+
| LocationID | TotalPayment |
+------------+--------------+
|        301 |       200.00 |
|        302 |       350.00 |
+------------+--------------+
2 rows in set (0.00 sec)
```

19.Calculate Total Payments per Location



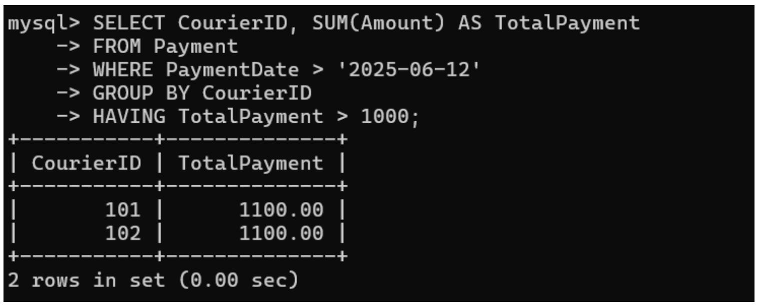
```
mysql> SELECT LocationID, SUM(Amount) AS TotalPayment
    -> FROM Payment
    -> GROUP BY LocationID;
+------------+--------------+
| LocationID | TotalPayment |
+------------+--------------+
|        301 |       200.00 |
|        302 |       350.00 |
+------------+--------------+
2 rows in set (0.00 sec)
```

20.Retrive couriers who have received payments totaling more than $1000 in a specific location



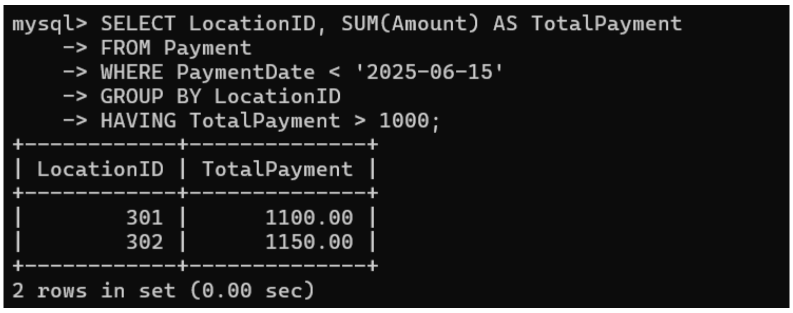
```
mysql> SELECT CourierID, SUM(Amount) AS TotalPayment
    -> FROM Payment
    -> WHERE LocationID = 301
    -> GROUP BY CourierID
    -> HAVING TotalPayment > 1000;
+-----------+--------------+
| CourierID | TotalPayment |
+-----------+--------------+
|       101 |      1300.00 |
+-----------+--------------+
1 row in set (0.00 sec)
```

21. Retrive couriers who have received payments totaling more than $1000 after a certain date



```
mysql> SELECT CourierID, SUM(Amount) AS TotalPayment
    -> FROM Payment
    -> WHERE PaymentDate > '2025-06-12'
    -> GROUP BY CourierID
    -> HAVING TotalPayment > 1000;
+-----------+--------------+
| CourierID | TotalPayment |
+-----------+--------------+
|       101 |      1100.00 |
|       102 |      1100.00 |
+-----------+--------------+
2 rows in set (0.00 sec)
```

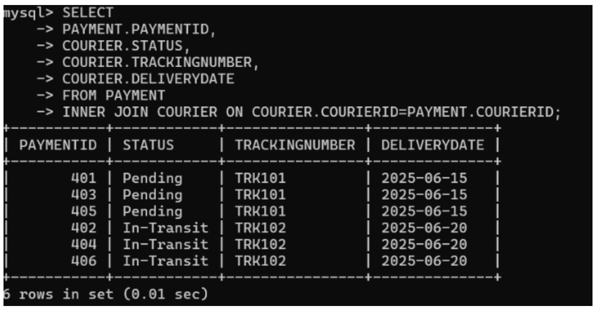
22.Retrive locations where the total amount received is more than $5000 before a certain date



```
mysql> SELECT LocationID, SUM(Amount) AS TotalPayment
    -> FROM Payment
    -> WHERE PaymentDate < '2025-06-15'
    -> GROUP BY LocationID
    -> HAVING TotalPayment > 1000;
+------------+--------------+
| LocationID | TotalPayment |
+------------+--------------+
|        301 |      1100.00 |
|        302 |      1150.00 |
+------------+--------------+
2 rows in set (0.00 sec)
```

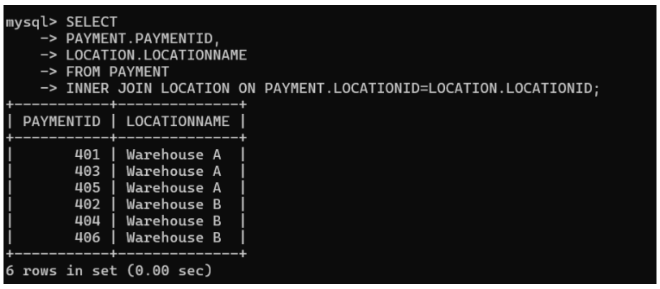
Task 4: Inner Join,Full Outer Join, Cross Join, Left Outer Join,Right Outer Join

23. Retrieve Payments with Courier Information



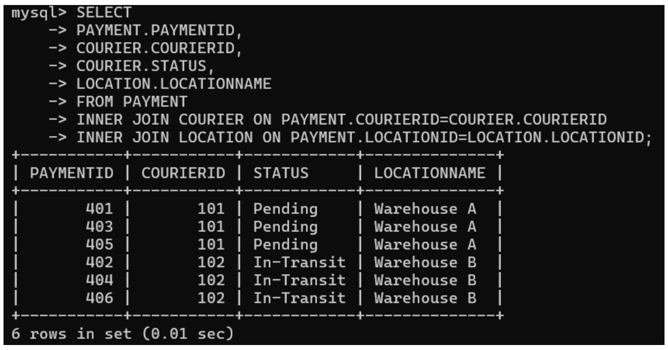
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> COURIER.STATUS,
    -> COURIER.TRACKINGNUMBER,
    -> COURIER.DELIVERYDATE
    -> FROM PAYMENT
    -> INNER JOIN COURIER ON COURIER.COURIERID=PAYMENT.COURIERID;
+-----------+------------+----------------+--------------+
| PAYMENTID | STATUS     | TRACKINGNUMBER | DELIVERYDATE |
+-----------+------------+----------------+--------------+
|       401 | Pending    | TRK101         | 2025-06-15   |
|       403 | Pending    | TRK101         | 2025-06-15   |
|       405 | Pending    | TRK101         | 2025-06-15   |
|       402 | In-Transit | TRK102         | 2025-06-20   |
|       404 | In-Transit | TRK102         | 2025-06-20   |
|       406 | In-Transit | TRK102         | 2025-06-20   |
+-----------+------------+----------------+--------------+
6 rows in set (0.01 sec)
```

24. Retrieve Payments with Location Information



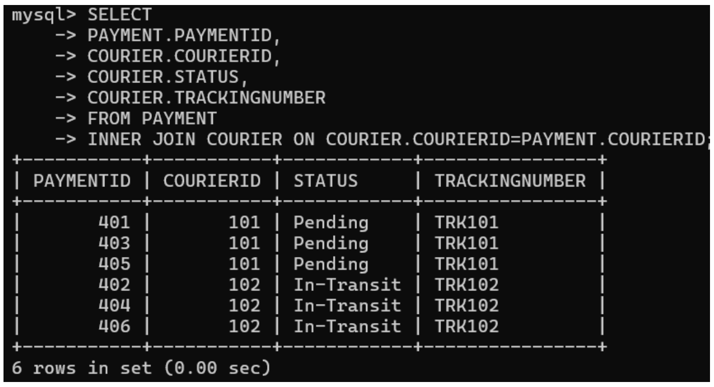
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> LOCATION.LOCATIONNAME
    -> FROM PAYMENT
    -> INNER JOIN LOCATION ON PAYMENT.LOCATIONID=LOCATION.LOCATIONID;
+-----------+--------------+
| PAYMENTID | LOCATIONNAME |
+-----------+--------------+
|       401 | Warehouse A  |
|       403 | Warehouse A  |
|       405 | Warehouse A  |
|       402 | Warehouse B  |
|       404 | Warehouse B  |
|       406 | Warehouse B  |
+-----------+--------------+
6 rows in set (0.00 sec)
```

25. Retrieve Payments with Courier and Location Information



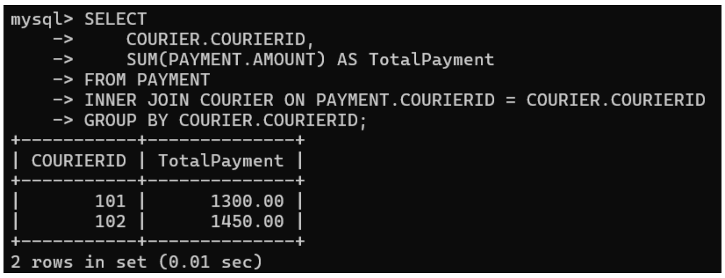
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> COURIER.COURIERID,
    -> COURIER.STATUS,
    -> LOCATION.LOCATIONNAME
    -> FROM PAYMENT
    -> INNER JOIN COURIER ON PAYMENT.COURIERID=COURIER.COURIERID
    -> INNER JOIN LOCATION ON PAYMENT.LOCATIONID=LOCATION.LOCATIONID;
+-----------+-----------+------------+--------------+
| PAYMENTID | COURIERID | STATUS     | LOCATIONNAME |
+-----------+-----------+------------+--------------+
|       401 |       101 | Pending    | Warehouse A  |
|       403 |       101 | Pending    | Warehouse A  |
|       405 |       101 | Pending    | Warehouse A  |
|       402 |       102 | In-Transit | Warehouse B  |
|       404 |       102 | In-Transit | Warehouse B  |
|       406 |       102 | In-Transit | Warehouse B  |
+-----------+-----------+------------+--------------+
6 rows in set (0.01 sec)
```

26. List all payments with courier deta



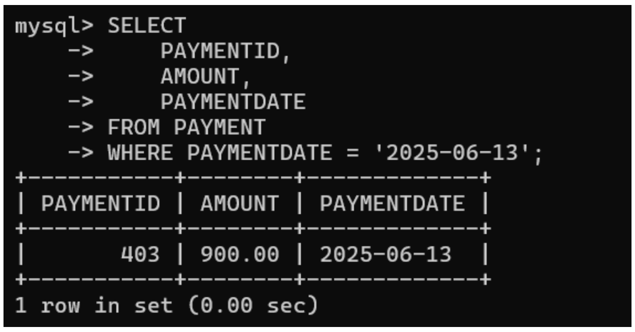
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> COURIER.COURIERID,
    -> COURIER.STATUS,
    -> COURIER.TRACKINGNUMBER
    -> FROM PAYMENT
    -> INNER JOIN COURIER ON COURIER.COURIERID=PAYMENT.COURIERID
+-----------+-----------+------------+----------------+
| PAYMENTID | COURIERID | STATUS     | TRACKINGNUMBER |
+-----------+-----------+------------+----------------+
|       401 |       101 | Pending    | TRK101         |
|       403 |       101 | Pending    | TRK101         |
|       405 |       101 | Pending    | TRK101         |
|       402 |       102 | In-Transit | TRK102         |
|       404 |       102 | In-Transit | TRK102         |
|       406 |       102 | In-Transit | TRK102         |
+-----------+-----------+------------+----------------+
6 rows in set (0.00 sec)
```

27. Total payments received for each courier



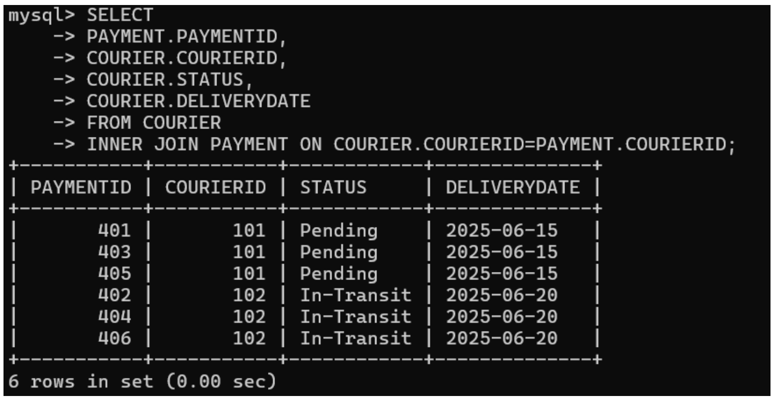
```
mysql> SELECT
    ->     COURIER.COURIERID,
    ->     SUM(PAYMENT.AMOUNT) AS TotalPayment
    -> FROM PAYMENT
    -> INNER JOIN COURIER ON PAYMENT.COURIERID = COURIER.COURIERID
    -> GROUP BY COURIER.COURIERID;
+-----------+--------------+
| COURIERID | TotalPayment |
+-----------+--------------+
|       101 |      1300.00 |
|       102 |      1450.00 |
+-----------+--------------+
2 rows in set (0.01 sec)
```

28. List payments made on a specific date



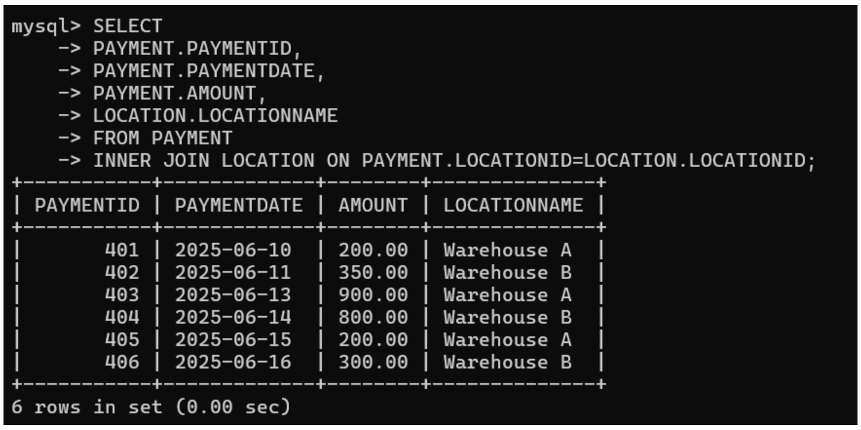
```
mysql> SELECT
    ->     PAYMENTID,
    ->     AMOUNT,
    ->     PAYMENTDATE
    -> FROM PAYMENT
    -> WHERE PAYMENTDATE = '2025-06-13';
+-----------+--------+-------------+
| PAYMENTID | AMOUNT | PAYMENTDATE |
+-----------+--------+-------------+
|       403 | 900.00 | 2025-06-13  |
+-----------+--------+-------------+
1 row in set (0.00 sec)
```

29. Get Courier Information for Each Payment



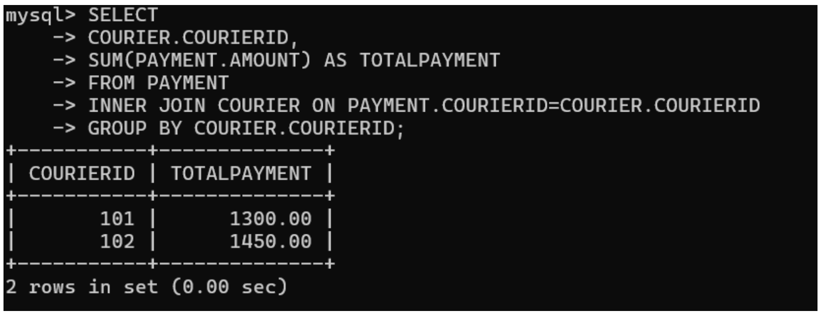
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> COURIER.COURIERID,
    -> COURIER.STATUS,
    -> COURIER.DELIVERYDATE
    -> FROM COURIER
    -> INNER JOIN PAYMENT ON COURIER.COURIERID=PAYMENT.COURIERID;
+-----------+-----------+------------+--------------+
| PAYMENTID | COURIERID | STATUS     | DELIVERYDATE |
+-----------+-----------+------------+--------------+
|       401 |       101 | Pending    | 2025-06-15   |
|       403 |       101 | Pending    | 2025-06-15   |
|       405 |       101 | Pending    | 2025-06-15   |
|       402 |       102 | In-Transit | 2025-06-20   |
|       404 |       102 | In-Transit | 2025-06-20   |
|       406 |       102 | In-Transit | 2025-06-20   |
+-----------+-----------+------------+--------------+
6 rows in set (0.00 sec)
```

30. Get Payment Details with Location



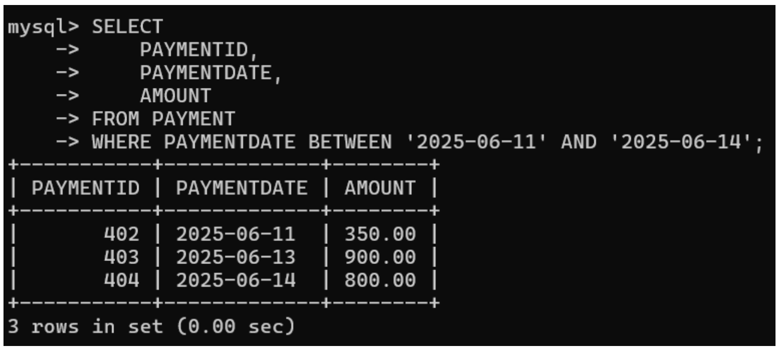
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> PAYMENT.PAYMENTDATE,
    -> PAYMENT.AMOUNT,
    -> LOCATION.LOCATIONNAME
    -> FROM PAYMENT
    -> INNER JOIN LOCATION ON PAYMENT.LOCATIONID=LOCATION.LOCATIONID;
+-----------+-------------+--------+--------------+
| PAYMENTID | PAYMENTDATE | AMOUNT | LOCATIONNAME |
+-----------+-------------+--------+--------------+
|       401 | 2025-06-10  | 200.00 | Warehouse A  |
|       402 | 2025-06-11  | 350.00 | Warehouse B  |
|       403 | 2025-06-13  | 900.00 | Warehouse A  |
|       404 | 2025-06-14  | 800.00 | Warehouse B  |
|       405 | 2025-06-15  | 200.00 | Warehouse A  |
|       406 | 2025-06-16  | 300.00 | Warehouse B  |
+-----------+-------------+--------+--------------+
6 rows in set (0.00 sec)
```

31. Calculating Total Payments for Each Courier



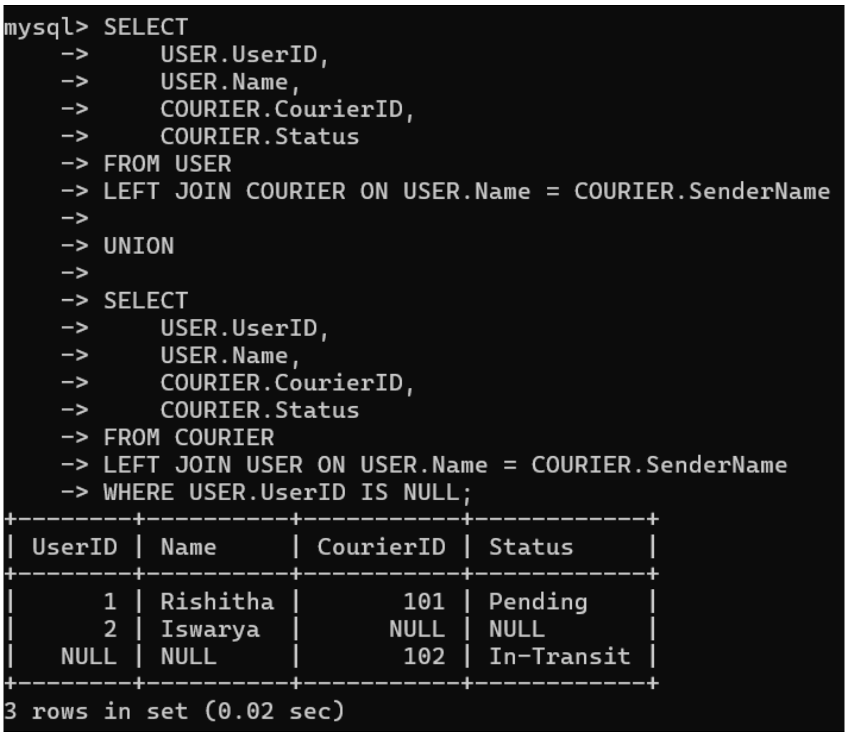
```
mysql> SELECT
    -> COURIER.COURIERID,
    -> SUM(PAYMENT.AMOUNT) AS TOTALPAYMENT
    -> FROM PAYMENT
    -> INNER JOIN COURIER ON PAYMENT.COURIERID=COURIER.COURIERID
    -> GROUP BY COURIER.COURIERID;
+-----------+--------------+
| COURIERID | TOTALPAYMENT |
+-----------+--------------+
|       101 |      1300.00 |
|       102 |      1450.00 |
+-----------+--------------+
2 rows in set (0.00 sec)
```

32. List Payments Within a Date Range



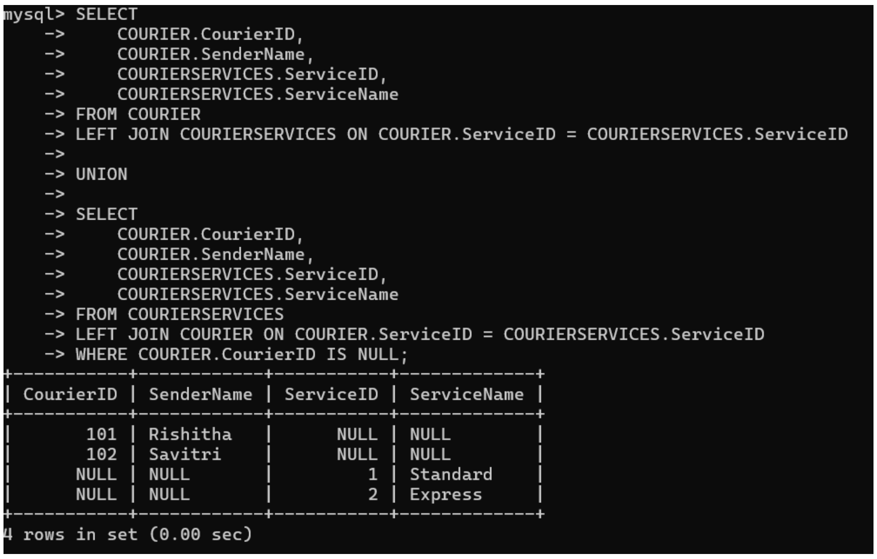
```
mysql> SELECT
    ->     PAYMENTID,
    ->     PAYMENTDATE,
    ->     AMOUNT
    -> FROM PAYMENT
    -> WHERE PAYMENTDATE BETWEEN '2025-06-11' AND '2025-06-14';
+-----------+-------------+--------+
| PAYMENTID | PAYMENTDATE | AMOUNT |
+-----------+-------------+--------+
|       402 | 2025-06-11  | 350.00 |
|       403 | 2025-06-13  | 900.00 |
|       404 | 2025-06-14  | 800.00 |
+-----------+-------------+--------+
3 rows in set (0.00 sec)
```

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



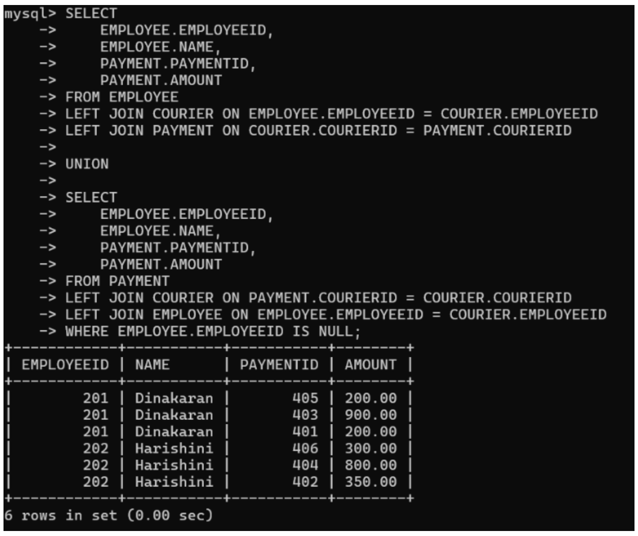
```
mysql> SELECT
    ->     USER.UserID,
    ->     USER.Name,
    ->     COURIER.CourierID,
    ->     COURIER.Status
    -> FROM USER
    -> LEFT JOIN COURIER ON USER.Name = COURIER.SenderName
    ->
    -> UNION
    ->
    -> SELECT
    ->     USER.UserID,
    ->     USER.Name,
    ->     COURIER.CourierID,
    ->     COURIER.Status
    -> FROM COURIER
    -> LEFT JOIN USER ON USER.Name = COURIER.SenderName
    -> WHERE USER.UserID IS NULL;
+--------+----------+-----------+------------+
| UserID | Name     | CourierID | Status     |
+--------+----------+-----------+------------+
|      1 | Rishitha |       101 | Pending    |
|      2 | Iswarya  |      NULL | NULL       |
|   NULL | NULL     |       102 | In-Transit |
+--------+----------+-----------+------------+
3 rows in set (0.02 sec)
```

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



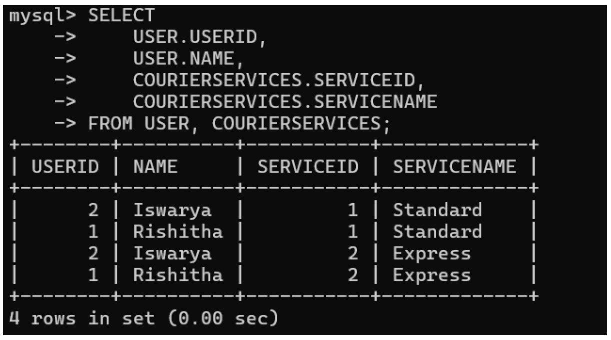
```
mysql> SELECT
    ->     COURIER.CourierID,
    ->     COURIER.SenderName,
    ->     COURIERSERVICES.ServiceID,
    ->     COURIERSERVICES.ServiceName
    -> FROM COURIER
    -> LEFT JOIN COURIERSERVICES ON COURIER.ServiceID = COURIERSERVICES.ServiceID
    ->
    -> UNION
    ->
    -> SELECT
    ->     COURIER.CourierID,
    ->     COURIER.SenderName,
    ->     COURIERSERVICES.ServiceID,
    ->     COURIERSERVICES.ServiceName
    -> FROM COURIERSERVICES
    -> LEFT JOIN COURIER ON COURIER.ServiceID = COURIERSERVICES.ServiceID
    -> WHERE COURIER.CourierID IS NULL;
+-----------+------------+-----------+-------------+
| CourierID | SenderName | ServiceID | ServiceName |
+-----------+------------+-----------+-------------+
|       101 | Rishitha   |      NULL | NULL        |
|       102 | Savitri    |      NULL | NULL        |
|      NULL | NULL       |         1 | Standard    |
|      NULL | NULL       |         2 | Express     |
+-----------+------------+-----------+-------------+
4 rows in set (0.00 sec)
```

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



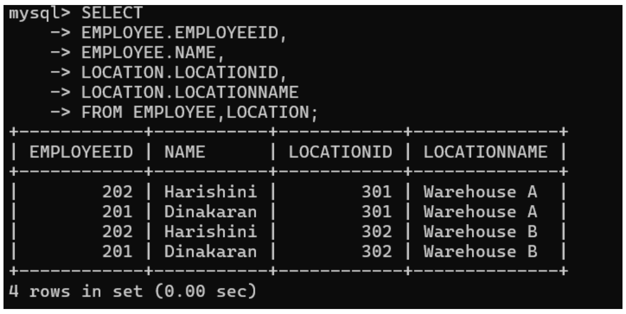
```
mysql> SELECT
    ->     EMPLOYEE.EMPLOYEEID,
    ->     EMPLOYEE.NAME,
    ->     PAYMENT.PAYMENTID,
    ->     PAYMENT.AMOUNT
    -> FROM EMPLOYEE
    -> LEFT JOIN COURIER ON EMPLOYEE.EMPLOYEEID = COURIER.EMPLOYEEID
    -> LEFT JOIN PAYMENT ON COURIER.COURIERID = PAYMENT.COURIERID
    ->
    -> UNION
    ->
    -> SELECT
    ->     EMPLOYEE.EMPLOYEEID,
    ->     EMPLOYEE.NAME,
    ->     PAYMENT.PAYMENTID,
    ->     PAYMENT.AMOUNT
    -> FROM PAYMENT
    -> LEFT JOIN COURIER ON PAYMENT.COURIERID = COURIER.COURIERID
    -> LEFT JOIN EMPLOYEE ON EMPLOYEE.EMPLOYEEID = COURIER.EMPLOYEEID
    -> WHERE EMPLOYEE.EMPLOYEEID IS NULL;
+------------+-----------+-----------+--------+
| EMPLOYEEID | NAME      | PAYMENTID | AMOUNT |
+------------+-----------+-----------+--------+
|        201 | Dinakaran |       405 | 200.00 |
|        201 | Dinakaran |       403 | 900.00 |
|        201 | Dinakaran |       401 | 200.00 |
|        202 | Harishini |       406 | 300.00 |
|        202 | Harishini |       404 | 800.00 |
|        202 | Harishini |       402 | 350.00 |
+------------+-----------+-----------+--------+
6 rows in set (0.00 sec)
```

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



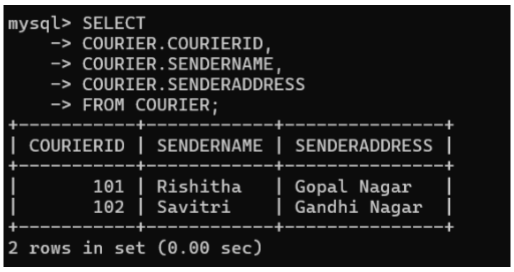
```
mysql> SELECT
    ->     USER.USERID,
    ->     USER.NAME,
    ->     COURIERSERVICES.SERVICEID,
    ->     COURIERSERVICES.SERVICENAME
    -> FROM USER, COURIERSERVICES;
+--------+----------+-----------+-------------+
| USERID | NAME     | SERVICEID | SERVICENAME |
+--------+----------+-----------+-------------+
|      2 | Iswarya  |         1 | Standard    |
|      1 | Rishitha |         1 | Standard    |
|      2 | Iswarya  |         2 | Express     |
|      1 | Rishitha |         2 | Express     |
+--------+----------+-----------+-------------+
4 rows in set (0.00 sec)
```

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



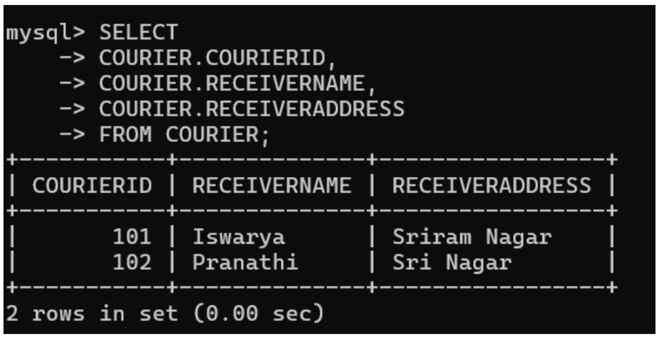
```
mysql> SELECT
    -> EMPLOYEE.EMPLOYEEID,
    -> EMPLOYEE.NAME,
    -> LOCATION.LOCATIONID,
    -> LOCATION.LOCATIONNAME
    -> FROM EMPLOYEE,LOCATION;
+------------+-----------+------------+--------------+
| EMPLOYEEID | NAME      | LOCATIONID | LOCATIONNAME |
+------------+-----------+------------+--------------+
|        202 | Harishini |        301 | Warehouse A  |
|        201 | Dinakaran |        301 | Warehouse A  |
|        202 | Harishini |        302 | Warehouse B  |
|        201 | Dinakaran |        302 | Warehouse B  |
+------------+-----------+------------+--------------+
4 rows in set (0.00 sec)
```

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



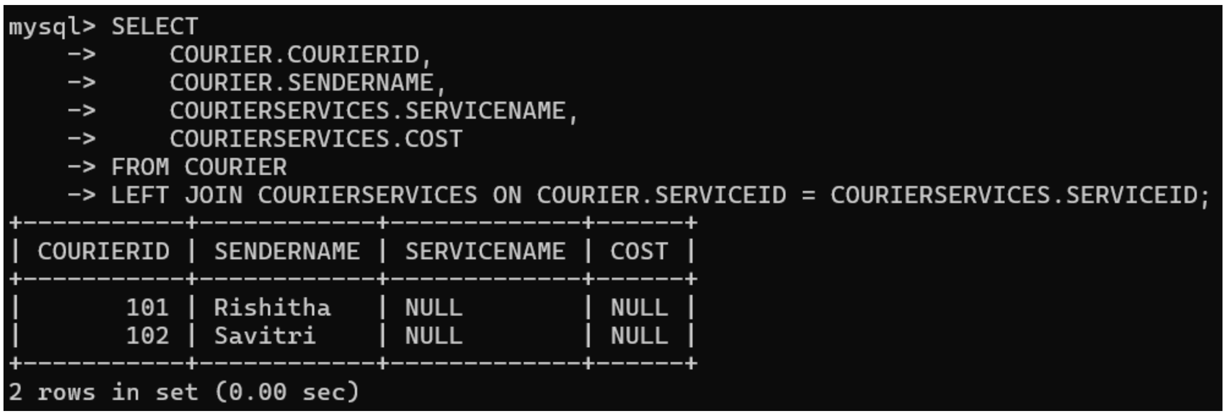
```
mysql> SELECT
    -> COURIER.COURIERID,
    -> COURIER.SENDERNAME,
    -> COURIER.SENDERADDRESS
    -> FROM COURIER;
+-----------+------------+---------------+
| COURIERID | SENDERNAME | SENDERADDRESS |
+-----------+------------+---------------+
|       101 | Rishitha   | Gopal Nagar   |
|       102 | Savitri    | Gandhi Nagar  |
+-----------+------------+---------------+
2 rows in set (0.00 sec)
```

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



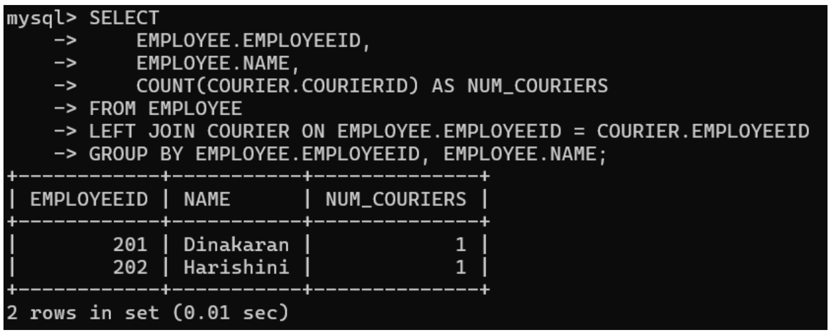
```
mysql> SELECT
    -> COURIER.COURIERID,
    -> COURIER.RECEIVERNAME,
    -> COURIER.RECEIVERADDRESS
    -> FROM COURIER;
+-----------+--------------+-----------------+
| COURIERID | RECEIVERNAME | RECEIVERADDRESS |
+-----------+--------------+-----------------+
|       101 | Iswarya      | Sriram Nagar    |
|       102 | Pranathi     | Sri Nagar       |
+-----------+--------------+-----------------+
2 rows in set (0.00 sec)
```

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



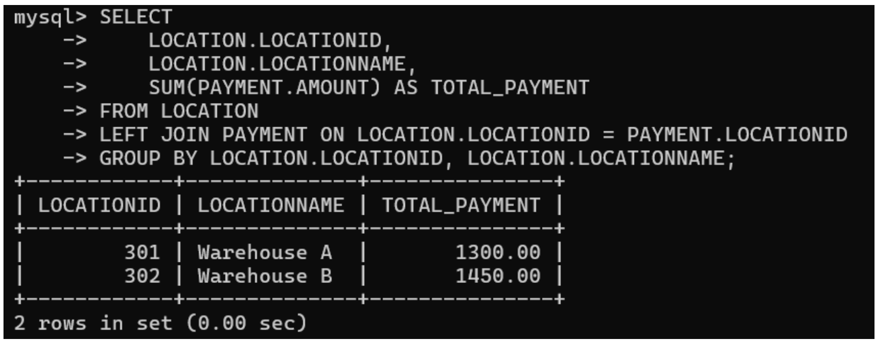
```
mysql> SELECT
    ->     COURIER.COURIERID,
    ->     COURIER.SENDERNAME,
    ->     COURIERSERVICES.SERVICENAME,
    ->     COURIERSERVICES.COST
    -> FROM COURIER
    -> LEFT JOIN COURIERSERVICES ON COURIER.SERVICEID = COURIERSERVICES.SERVICEID;
+-----------+------------+-------------+------+
| COURIERID | SENDERNAME | SERVICENAME | COST |
+-----------+------------+-------------+------+
|       101 | Rishitha   | NULL        | NULL |
|       102 | Savitri    | NULL        | NULL |
+-----------+------------+-------------+------+
2 rows in set (0.00 sec)
```

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



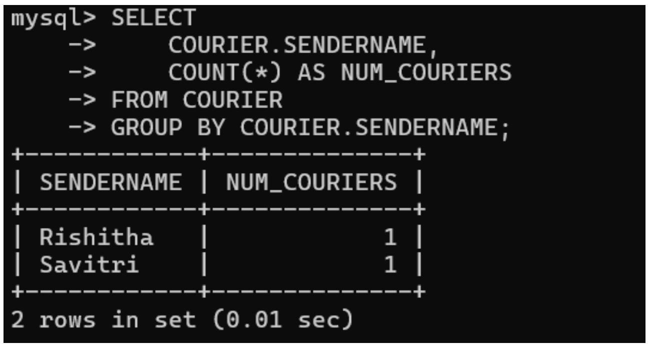
```
mysql> SELECT
    ->     EMPLOYEE.EMPLOYEEID,
    ->     EMPLOYEE.NAME,
    ->     COUNT(COURIER.COURIERID) AS NUM_COURIERS
    -> FROM EMPLOYEE
    -> LEFT JOIN COURIER ON EMPLOYEE.EMPLOYEEID = COURIER.EMPLOYEEID
    -> GROUP BY EMPLOYEE.EMPLOYEEID, EMPLOYEE.NAME;
+------------+-----------+--------------+
| EMPLOYEEID | NAME      | NUM_COURIERS |
+------------+-----------+--------------+
|        201 | Dinakaran |            1 |
|        202 | Harishini |            1 |
+------------+-----------+--------------+
2 rows in set (0.01 sec)
```

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



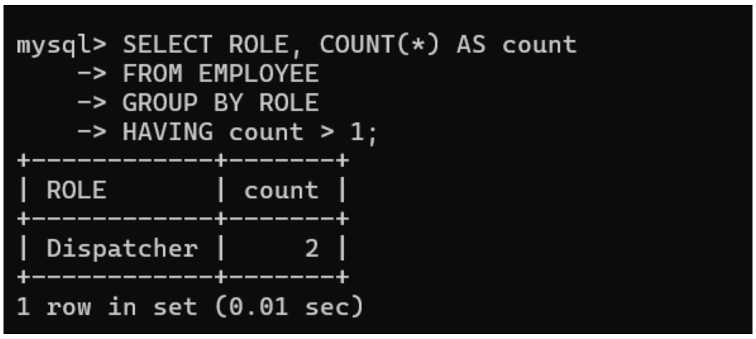
```
mysql> SELECT
    ->     LOCATION.LOCATIONID,
    ->     LOCATION.LOCATIONNAME,
    ->     SUM(PAYMENT.AMOUNT) AS TOTAL_PAYMENT
    -> FROM LOCATION
    -> LEFT JOIN PAYMENT ON LOCATION.LOCATIONID = PAYMENT.LOCATIONID
    -> GROUP BY LOCATION.LOCATIONID, LOCATION.LOCATIONNAME;
+------------+--------------+---------------+
| LOCATIONID | LOCATIONNAME | TOTAL_PAYMENT |
+------------+--------------+---------------+
|        301 | Warehouse A  |       1300.00 |
|        302 | Warehouse B  |       1450.00 |
+------------+--------------+---------------+
2 rows in set (0.00 sec)
```

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



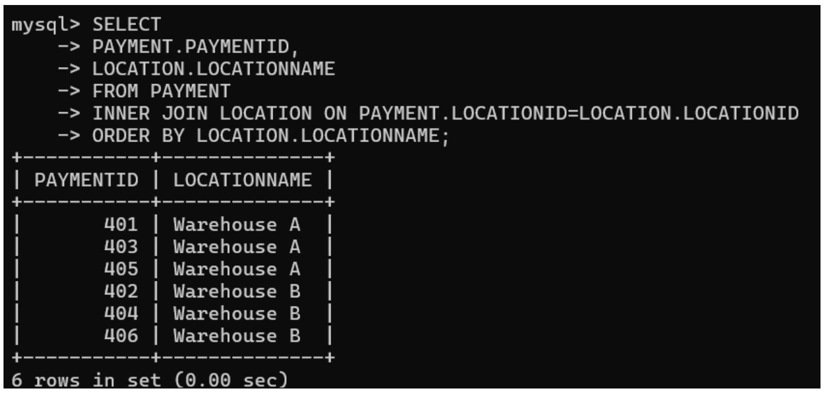
```
mysql> SELECT
    ->     COURIER.SENDERNAME,
    ->     COUNT(*) AS NUM_COURIERS
    -> FROM COURIER
    -> GROUP BY COURIER.SENDERNAME;
+------------+--------------+
| SENDERNAME | NUM_COURIERS |
+------------+--------------+
| Rishitha   |            1 |
| Savitri    |            1 |
+------------+--------------+
2 rows in set (0.01 sec)
```

44. List all employees who share the same role.



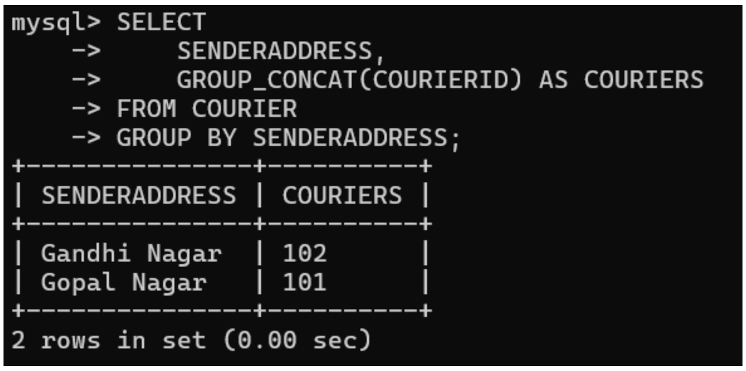
```
mysql> SELECT ROLE, COUNT(*) AS count
    -> FROM EMPLOYEE
    -> GROUP BY ROLE
    -> HAVING count > 1;
+------------+-------+
| ROLE       | count |
+------------+-------+
| Dispatcher |     2 |
+------------+-------+
1 row in set (0.01 sec)
```

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



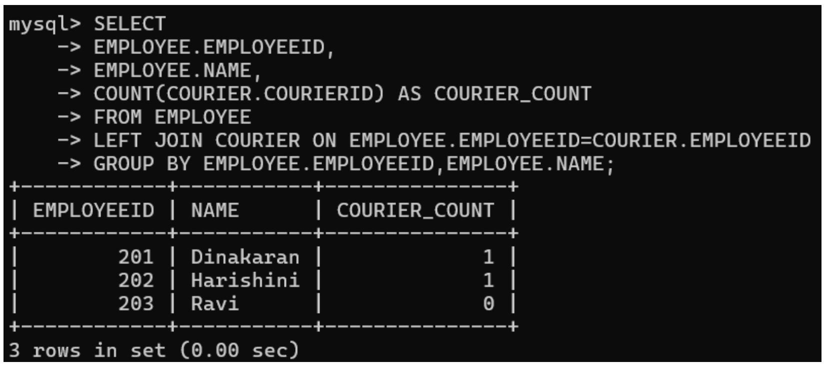
```
mysql> SELECT
    -> PAYMENT.PAYMENTID,
    -> LOCATION.LOCATIONNAME
    -> FROM PAYMENT
    -> INNER JOIN LOCATION ON PAYMENT.LOCATIONID=LOCATION.LOCATIONID
    -> ORDER BY LOCATION.LOCATIONNAME;
+-----------+--------------+
| PAYMENTID | LOCATIONNAME |
+-----------+--------------+
|       401 | Warehouse A  |
|       403 | Warehouse A  |
|       405 | Warehouse A  |
|       402 | Warehouse B  |
|       404 | Warehouse B  |
|       406 | Warehouse B  |
+-----------+--------------+
6 rows in set (0.00 sec)
```

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



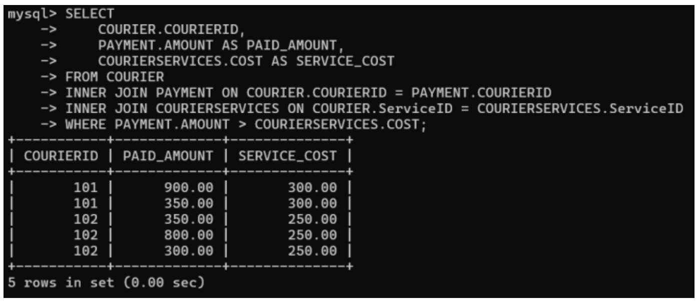
```
mysql> SELECT
    ->     SENDERADDRESS,
    ->     GROUP_CONCAT(COURIERID) AS COURIERS
    -> FROM COURIER
    -> GROUP BY SENDERADDRESS;
+---------------+----------+
| SENDERADDRESS | COURIERS |
+---------------+----------+
| Gandhi Nagar  | 102      |
| Gopal Nagar   | 101      |
+---------------+----------+
2 rows in set (0.00 sec)
```

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



```
mysql> SELECT
    -> EMPLOYEE.EMPLOYEEID,
    -> EMPLOYEE.NAME,
    -> COUNT(COURIER.COURIERID) AS COURIER_COUNT
    -> FROM EMPLOYEE
    -> LEFT JOIN COURIER ON EMPLOYEE.EMPLOYEEID=COURIER.EMPLOYEEID
    -> GROUP BY EMPLOYEE.EMPLOYEEID,EMPLOYEE.NAME;
+------------+-----------+---------------+
| EMPLOYEEID | NAME      | COURIER_COUNT |
+------------+-----------+---------------+
|        201 | Dinakaran |             1 |
|        202 | Harishini |             1 |
|        203 | Ravi      |             0 |
+------------+-----------+---------------+
3 rows in set (0.00 sec)
```

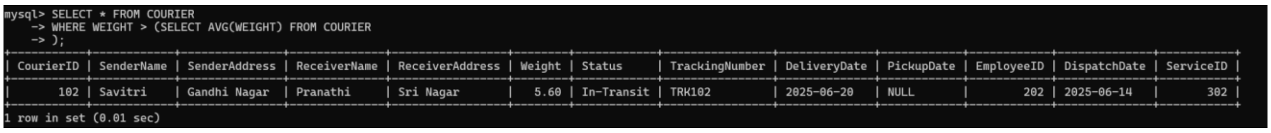
48. Find couriers that were paid an amount greater than the cost of their respective courier services



```
mysql> SELECT
    ->     COURIER.COURIERID,
    ->     PAYMENT.AMOUNT AS PAID_AMOUNT,
    ->     COURIERSERVICES.COST AS SERVICE_COST
    -> FROM COURIER
    -> INNER JOIN PAYMENT ON COURIER.COURIERID = PAYMENT.COURIERID
    -> INNER JOIN COURIERSERVICES ON COURIER.ServiceID = COURIERSERVICES.ServiceID
    -> WHERE PAYMENT.AMOUNT > COURIERSERVICES.COST;
+-----------+-------------+--------------+
| COURIERID | PAID_AMOUNT | SERVICE_COST |
+-----------+-------------+--------------+
|       101 |      900.00 |       300.00 |
|       101 |      350.00 |       300.00 |
|       102 |      350.00 |       250.00 |
|       102 |      800.00 |       250.00 |
|       102 |      300.00 |       250.00 |
+-----------+-------------+--------------+
5 rows in set (0.00 sec)
```

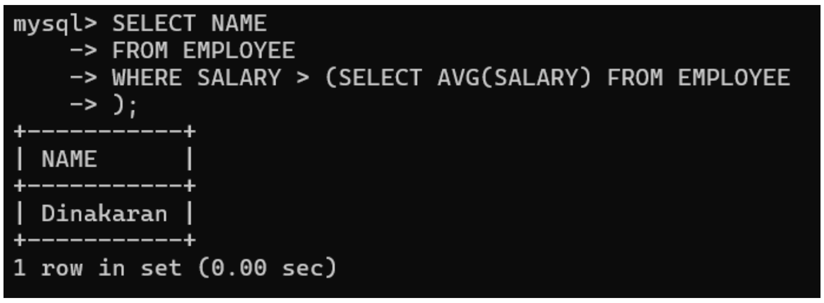
Scope: Inner Queries, Non Equi Joins, Equi joins,Exist,Any,All

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



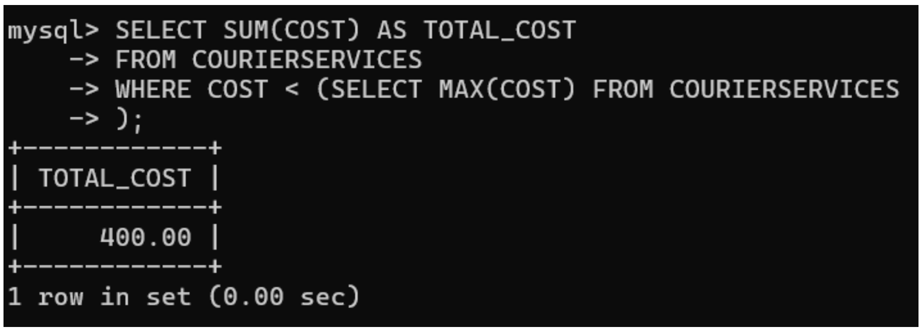
```
mysql> SELECT * FROM COURIER
    -> WHERE WEIGHT > (SELECT AVG(WEIGHT) FROM COURIER
    -> );
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status     | TrackingNumber | DeliveryDate | PickupDate | EmployeeID | DispatchDate | ServiceID |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
|       102 | Savitri    | Gandhi Nagar  | Pranathi     | Sri Nagar       |   5.60 | In-Transit | TRK102         | 2025-06-20   | NULL       |        202 | 2025-06-14   |       302 |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
1 row in set (0.01 sec)
```

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



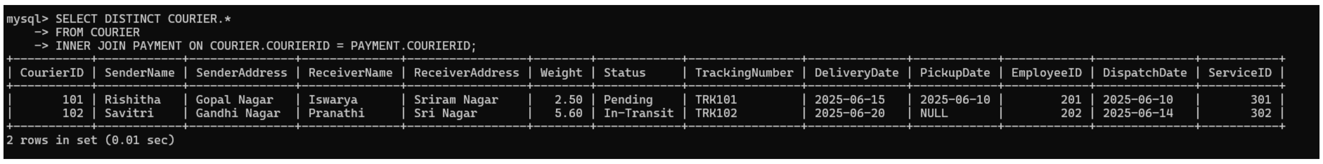
```
mysql> SELECT NAME
    -> FROM EMPLOYEE
    -> WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEE
    -> );
+-----------+
| NAME      |
+-----------+
| Dinakaran |
+-----------+
1 row in set (0.00 sec)
```

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



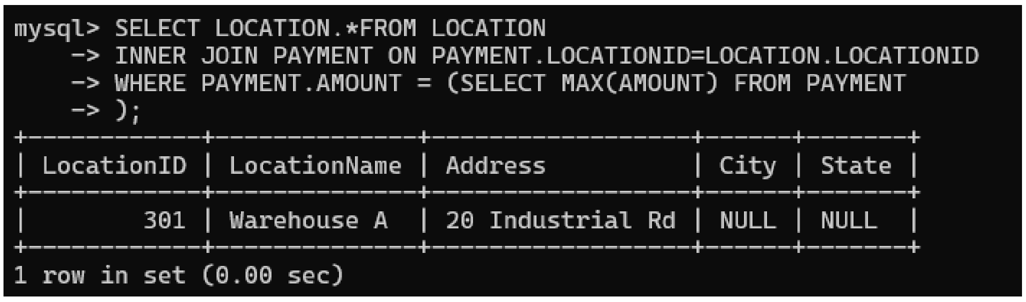
```
mysql> SELECT SUM(COST) AS TOTAL_COST
    -> FROM COURIERSERVICES
    -> WHERE COST < (SELECT MAX(COST) FROM COURIERSERVICES
    -> );
+------------+
| TOTAL_COST |
+------------+
|     400.00 |
+------------+
1 row in set (0.00 sec)
```

52. Find all couriers that have been paid for



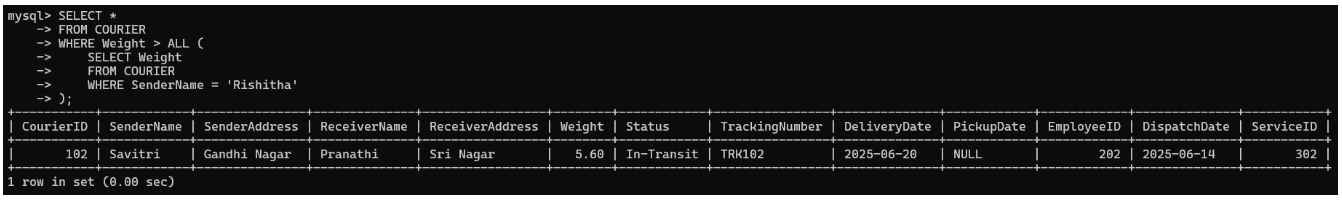
```
mysql> SELECT DISTINCT COURIER.*
    -> FROM COURIER
    -> INNER JOIN PAYMENT ON COURIER.COURIERID = PAYMENT.COURIERID;
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status     | TrackingNumber | DeliveryDate | PickupDate | EmployeeID | DispatchDate | ServiceID |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
|       101 | Rishitha   | Gopal Nagar   | Iswarya      | Sriram Nagar    |   2.50 | Pending    | TRK101         | 2025-06-15   | 2025-06-10 |        201 | 2025-06-10   |       301 |
|       102 | Savitri    | Gandhi Nagar  | Pranathi     | Sri Nagar       |   5.60 | In-Transit | TRK102         | 2025-06-20   | NULL       |        202 | 2025-06-14   |       302 |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
2 rows in set (0.01 sec)
```

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



```
mysql> SELECT LOCATION.*FROM LOCATION
    -> INNER JOIN PAYMENT ON PAYMENT.LOCATIONID=LOCATION.LOCATIONID
    -> WHERE PAYMENT.AMOUNT = (SELECT MAX(AMOUNT) FROM PAYMENT
    -> );
+------------+--------------+------------------+------+-------+
| LocationID | LocationName | Address          | City | State |
+------------+--------------+------------------+------+-------+
|        301 | Warehouse A  | 20 Industrial Rd | NULL | NULL  |
+------------+--------------+------------------+------+-------+
1 row in set (0.00 sec)
```

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



```
mysql> SELECT *
    -> FROM COURIER
    -> WHERE Weight > ALL (
    ->     SELECT Weight
    ->     FROM COURIER
    ->     WHERE SenderName = 'Rishitha'
    -> );
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status     | TrackingNumber | DeliveryDate | PickupDate | EmployeeID | DispatchDate | ServiceID |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
|       102 | Savitri    | Gandhi Nagar  | Pranathi     | Sri Nagar       |   5.60 | In-Transit | TRK102         | 2025-06-20   | NULL       |        202 | 2025-06-14   |       302 |
+-----------+------------+---------------+--------------+-----------------+--------+------------+----------------+--------------+------------+------------+--------------+-----------+
1 row in set (0.00 sec)
```