Task 1: Schema Design

```
-- Schema design of Courier Management System
CREATE TABLE User (
    UserID INT PRIMARY KEY,
    Name VARCHAR(255),
    Email VARCHAR(255) UNIQUE,
    Password VARCHAR(255),
    ContactNumber VARCHAR(20),
    Address TEXT
);


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
);


CREATE TABLE CourierServices (
```

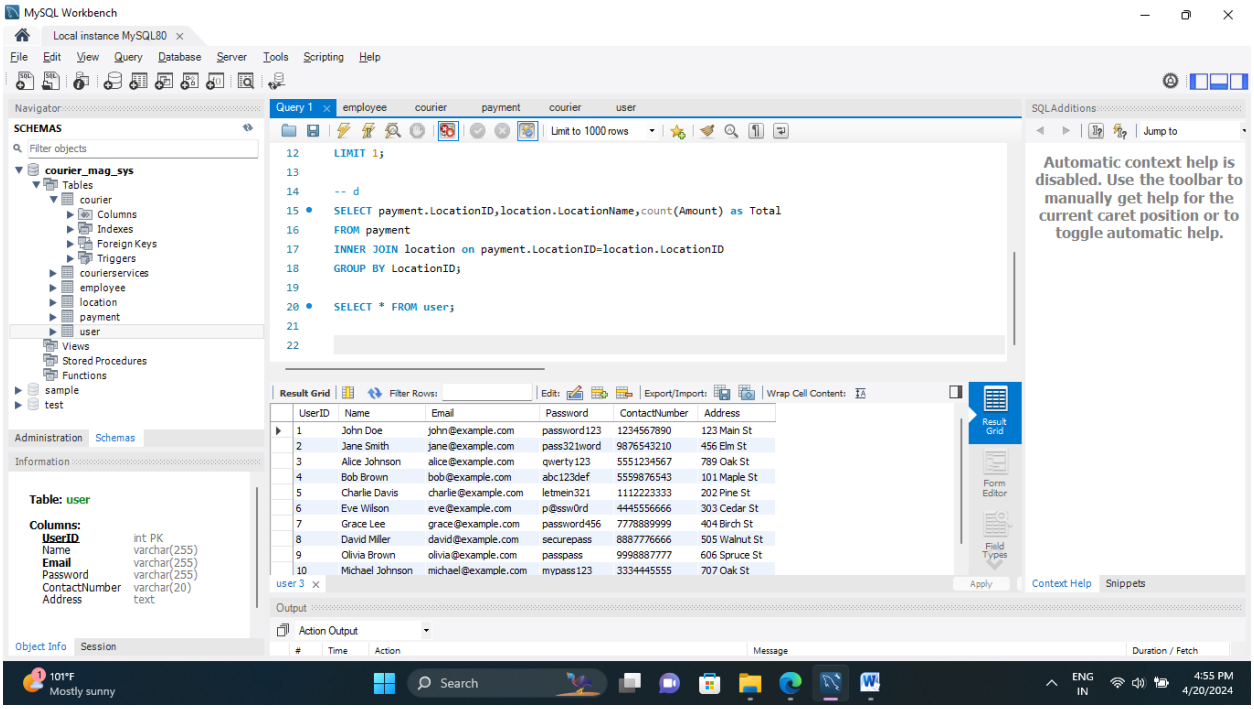
```
    ServiceID INT PRIMARY KEY,
    ServiceName VARCHAR(100),
    Cost DECIMAL(8, 2)
);


CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    Name VARCHAR(255),
    Email VARCHAR(255) UNIQUE,
    ContactNumber VARCHAR(20),
    Role VARCHAR(50),
    Salary DECIMAL(10, 2)
);


CREATE TABLE Location (
    LocationID INT PRIMARY KEY,
    LocationName VARCHAR(100),
    Address TEXT
);


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)
);



--Queries
```

Task 2: Select,Where

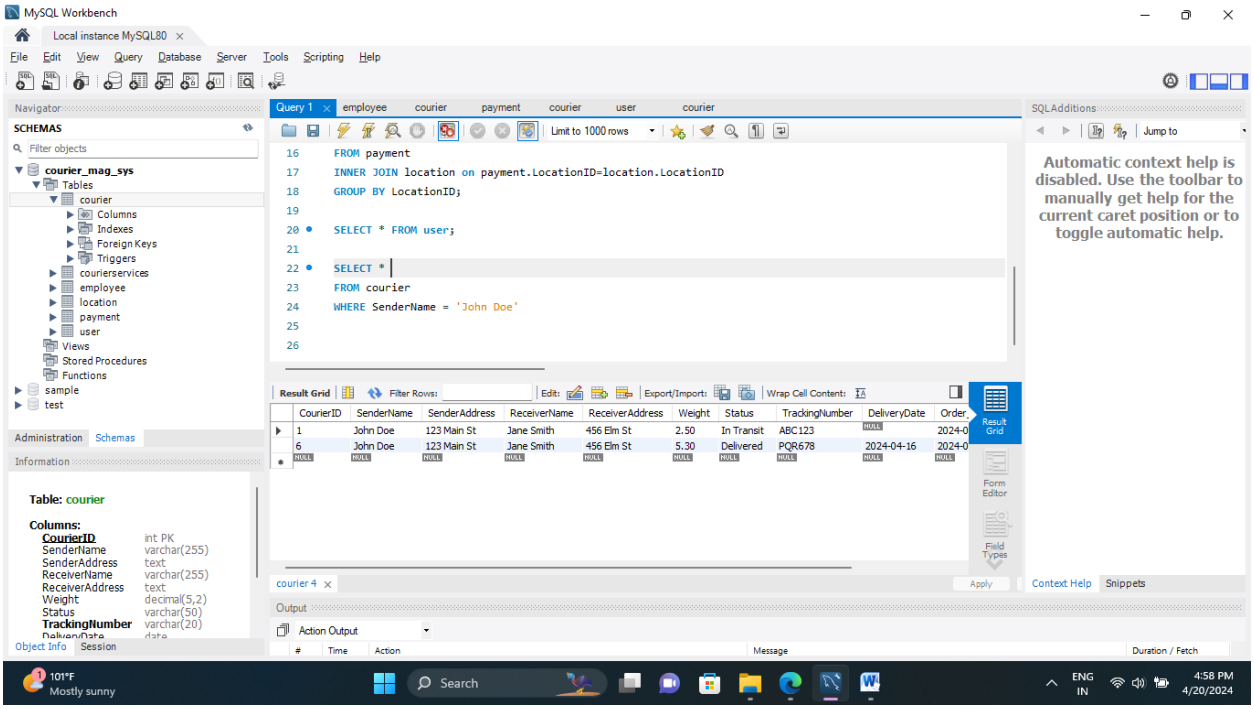
1. List all customers:

SELECT * FROM user;



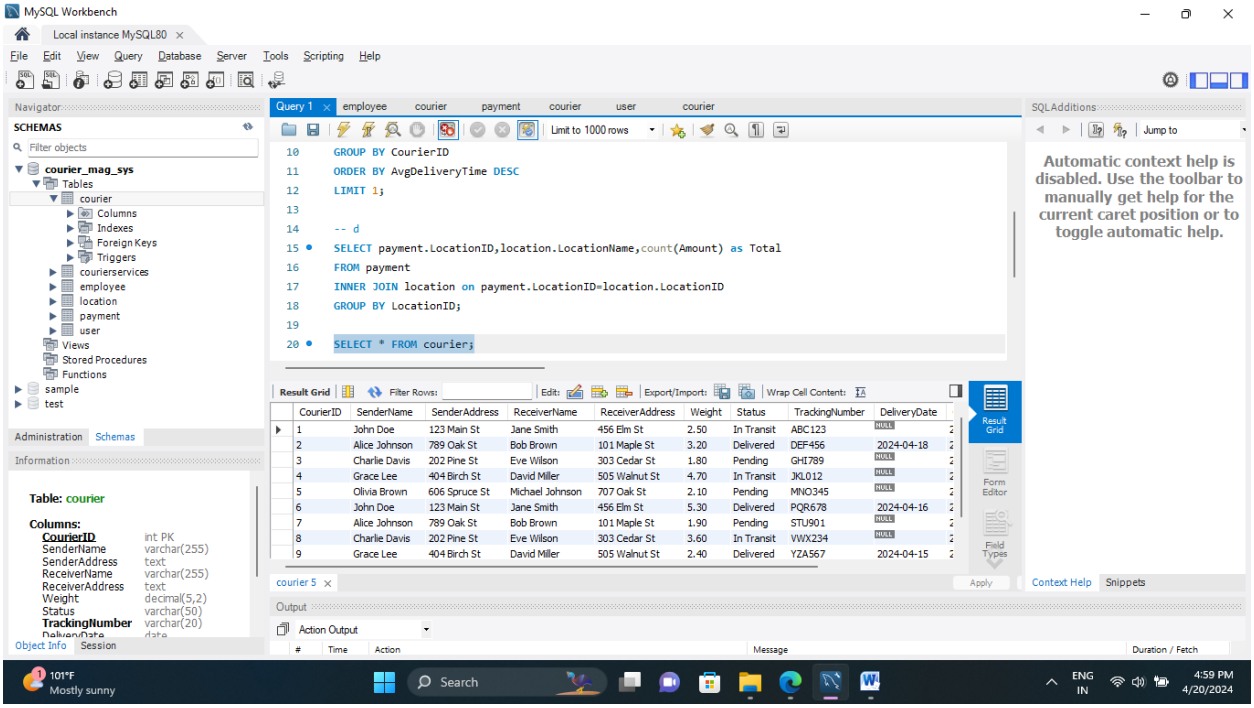
2. List all orders for a specific customer:

```
SELECT *

FROM courier

WHERE SenderName = 'John Doe'
```



3. List all couriers:

SELECT * FROM courier;

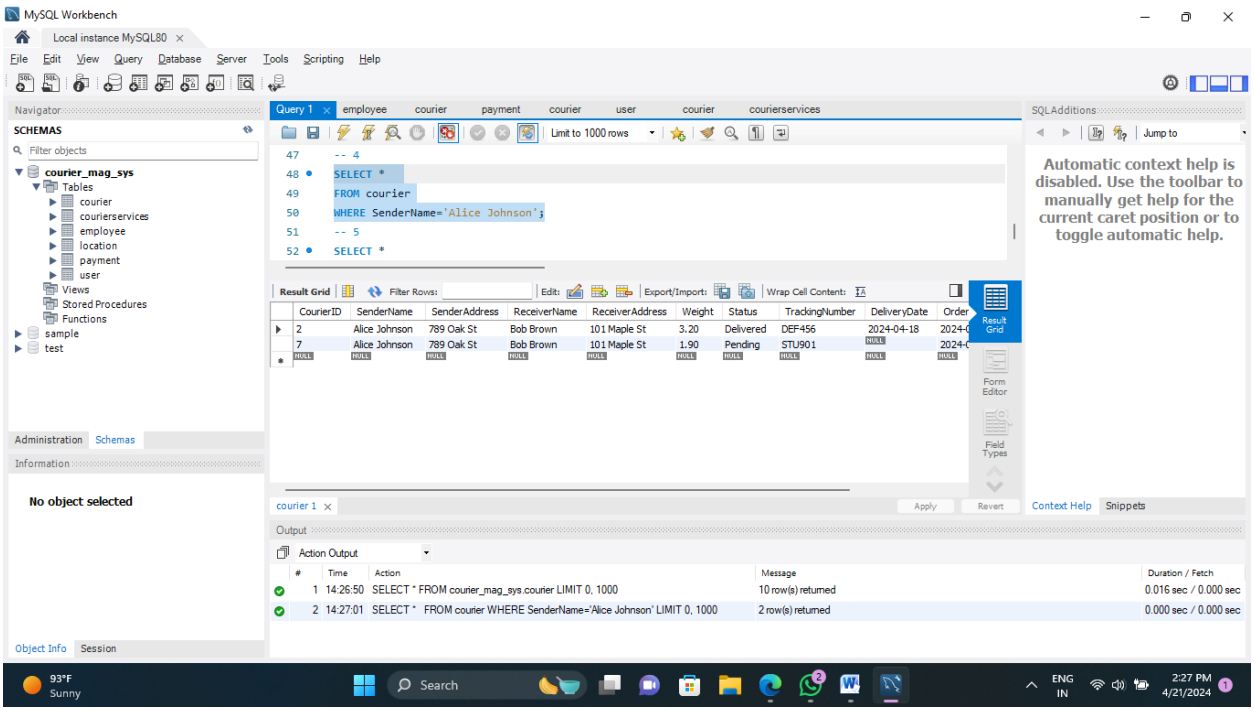


4. List all packages for a specific order:

SELECT *

FROM courier

WHERE SenderName='Alice Johnson';

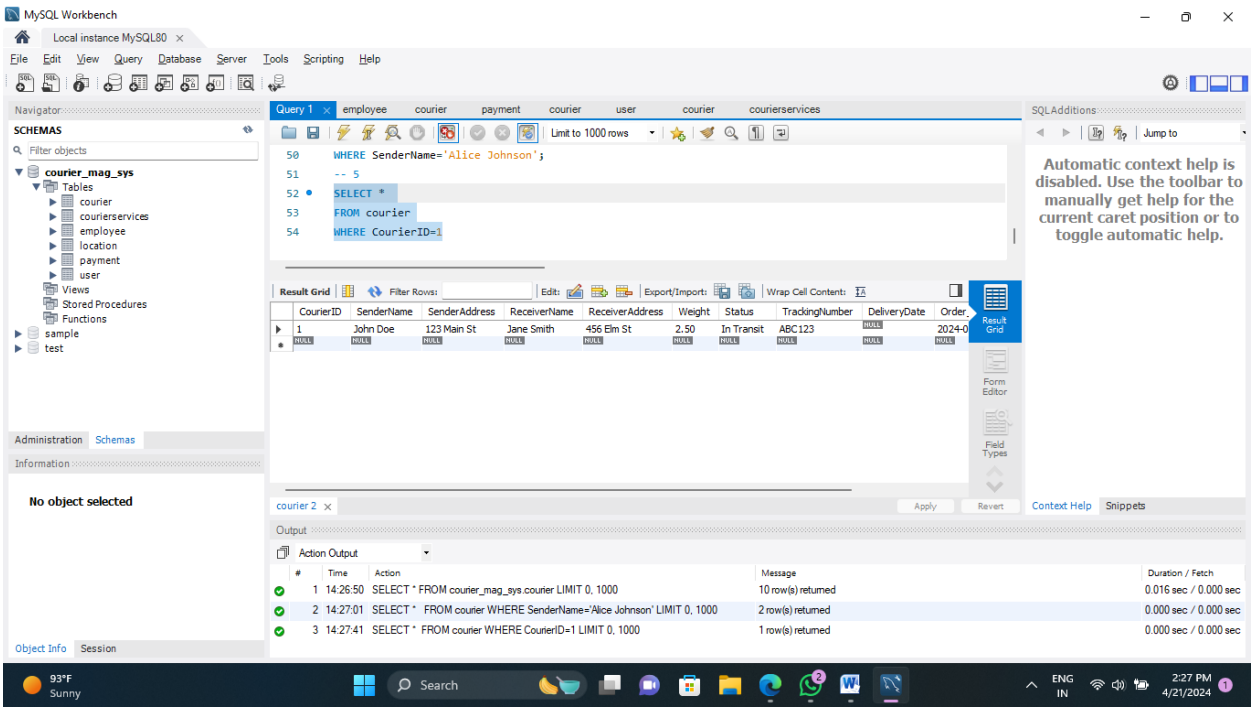


5. List all deliveries for a specific courier:

SELECT *

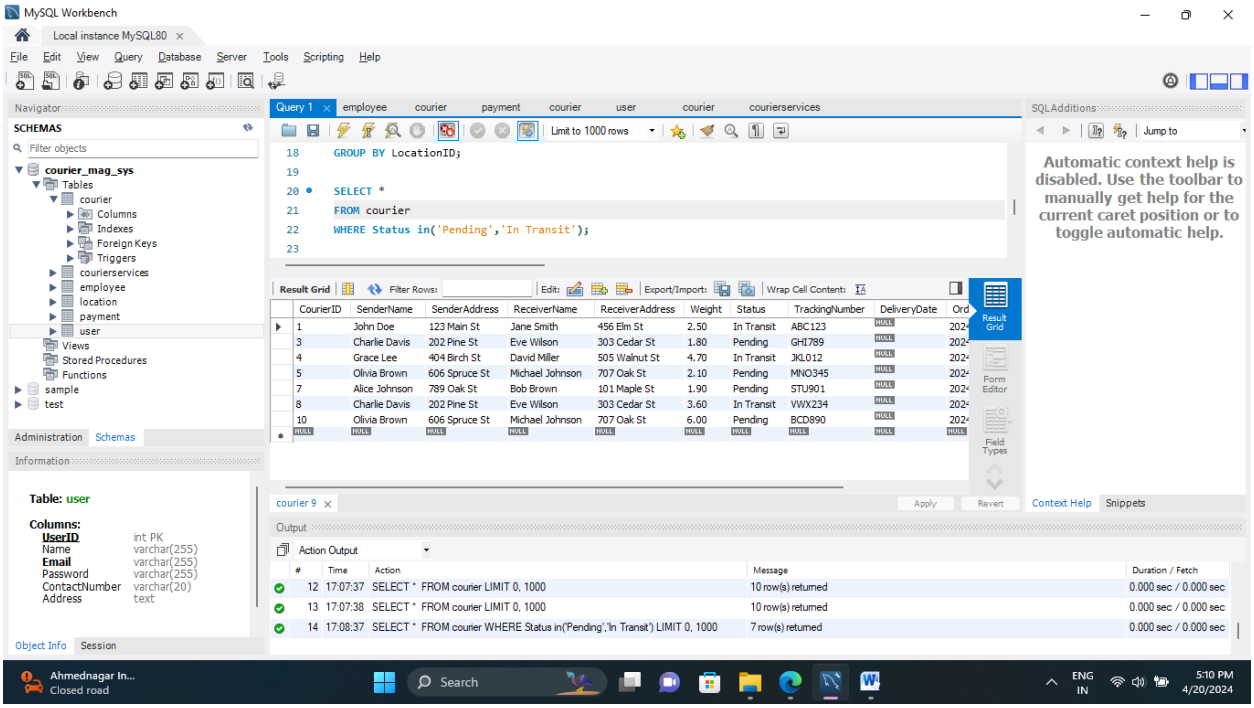
FROM courier

WHERE CourierID=1



6. List all undelivered packages:

```
SELECT *

FROM courier

WHERE Status in('Pending','In Transit');
```

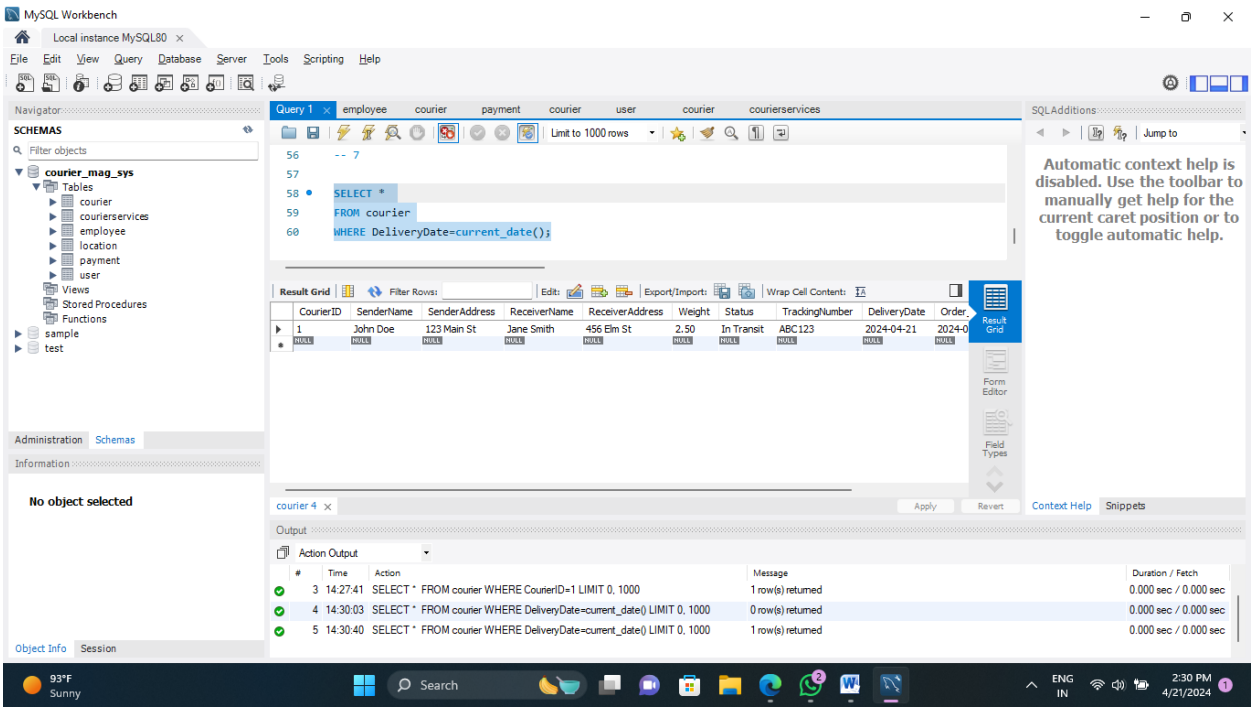


7. List all packages that are scheduled for delivery today:

SELECT *

FROM courier

WHERE DeliveryDate=current_date();

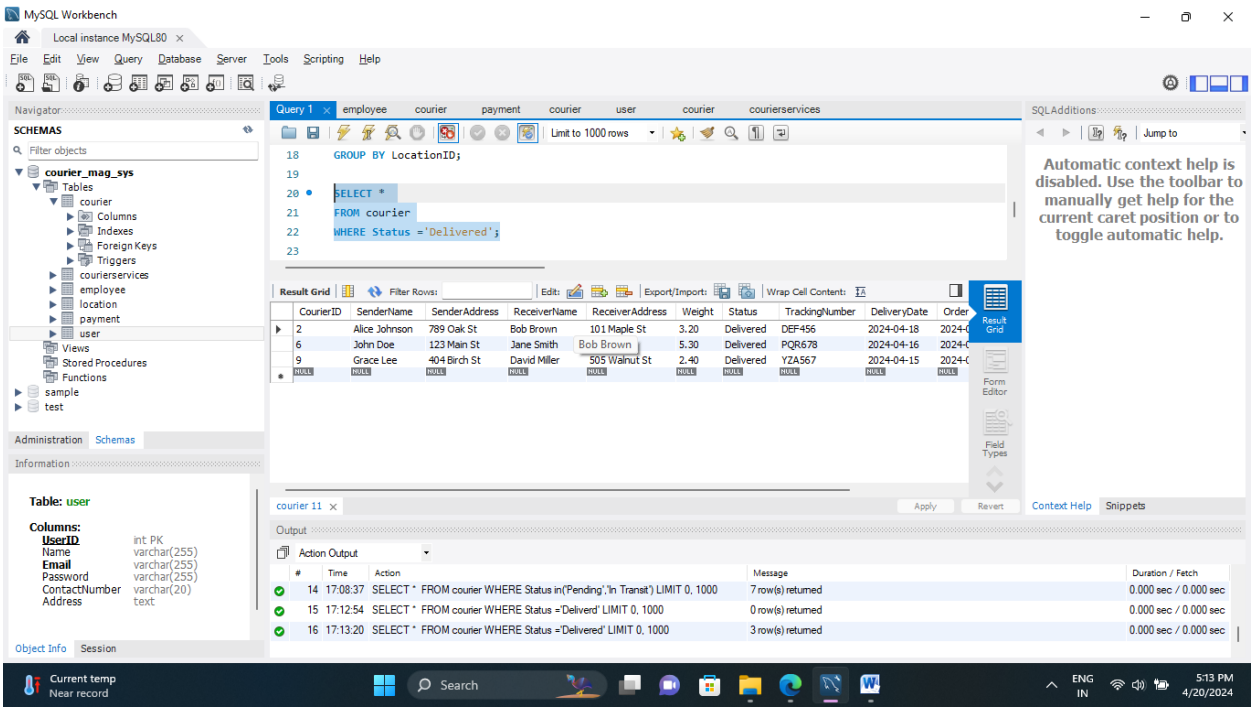


8. List all packages with a specific status:

SELECT *

FROM courier

WHERE Status ='Delivered';

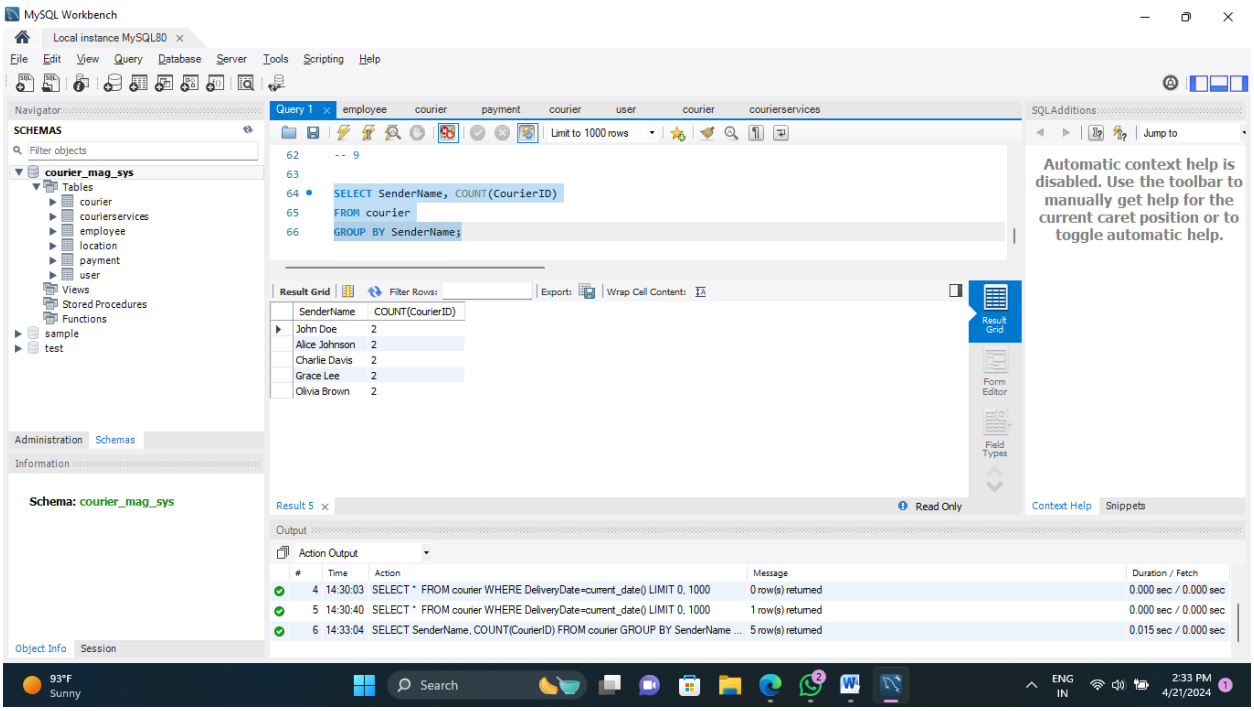


9. Calculate the total number of packages for each courier.

SELECT SenderName, COUNT(CourierID)

FROM courier

GROUP BY SenderName;

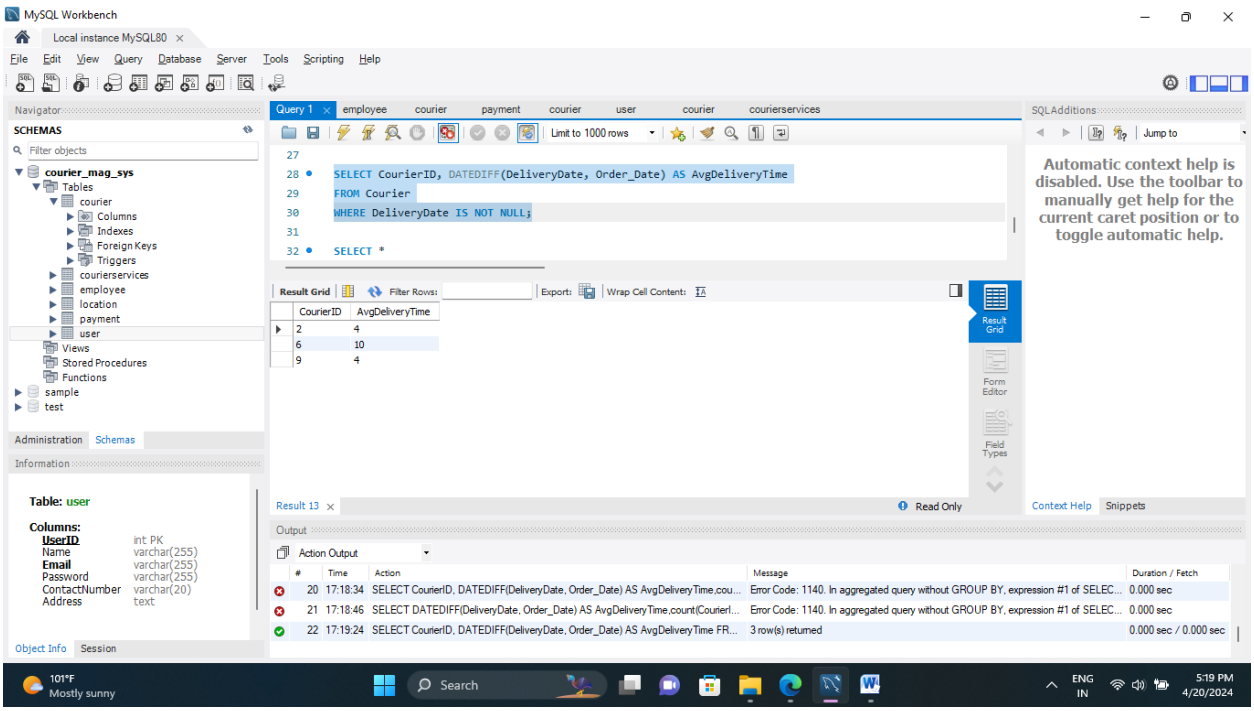


10. Find the average delivery time for each courier

SELECT CourierID, DATEDIFF(DeliveryDate, Order_Date) AS AvgDeliveryTime

FROM Courier

WHERE DeliveryDate IS NOT NULL;

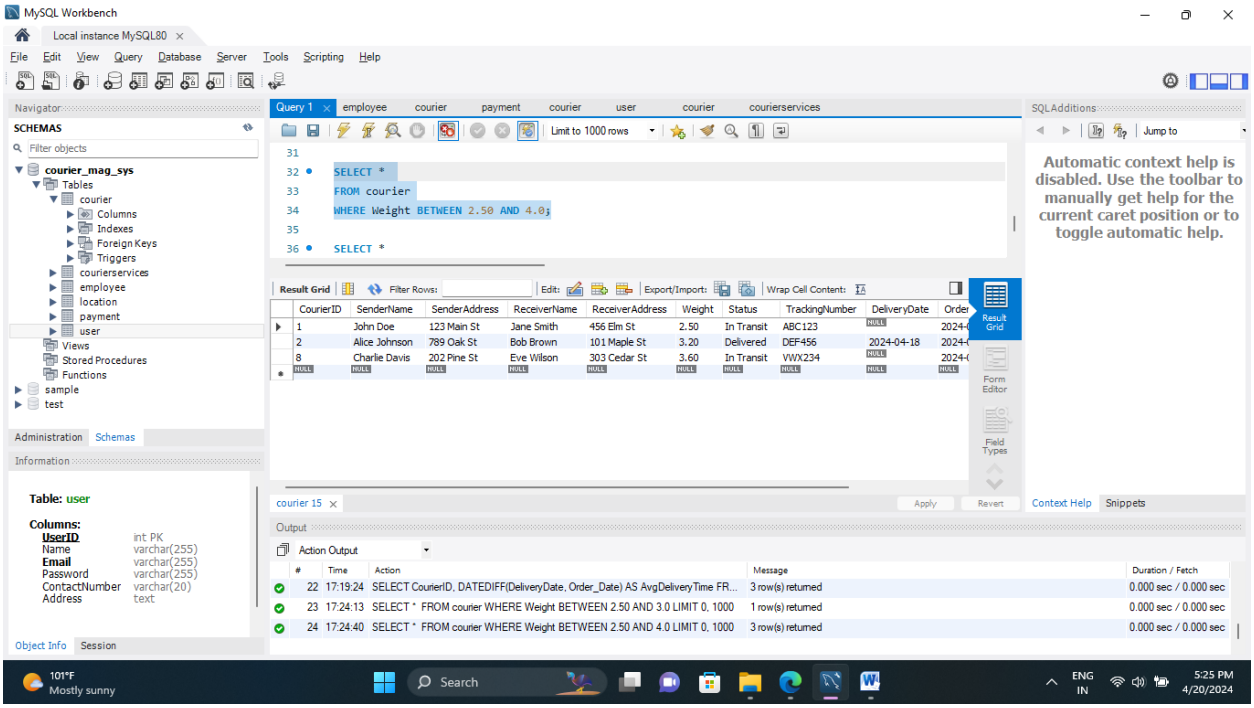


11. List all packages with a specific weight range:

SELECT *

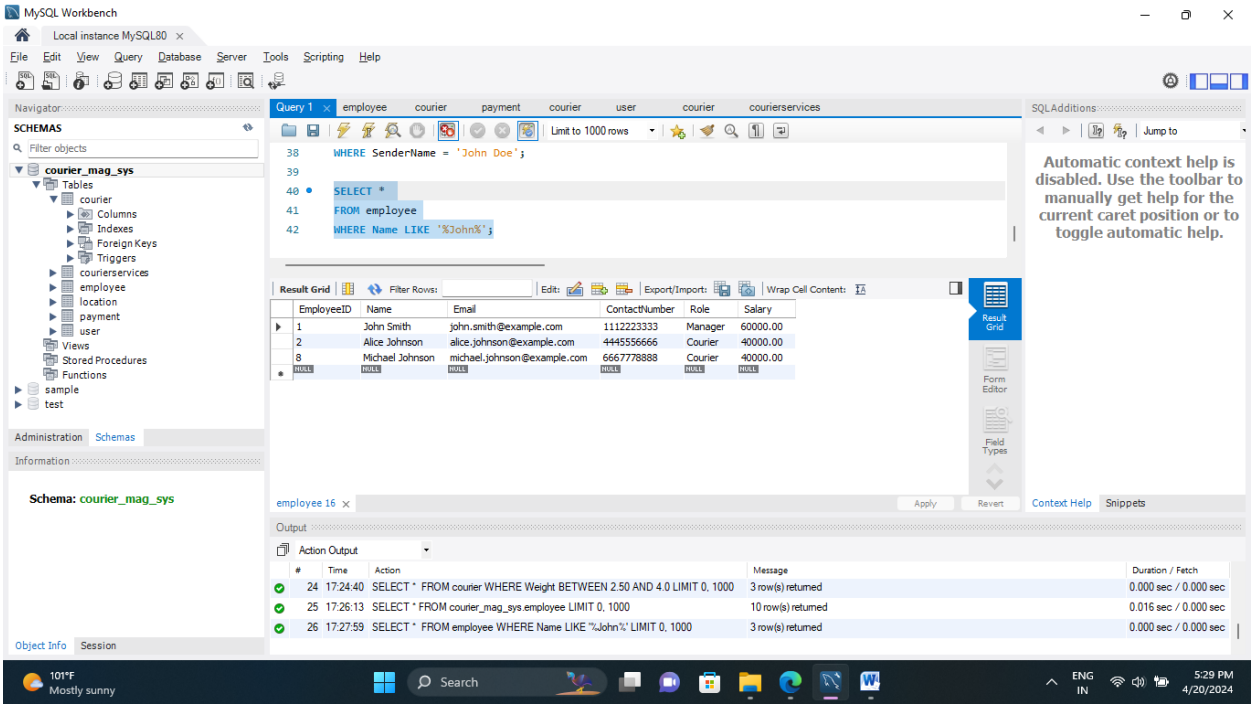
FROM courier

WHERE Weight BETWEEN 2.50 AND 4.0;



12. Retrieve employees whose names contain 'John'

```
SELECT *

FROM employee

WHERE Name LIKE '%John%';
```

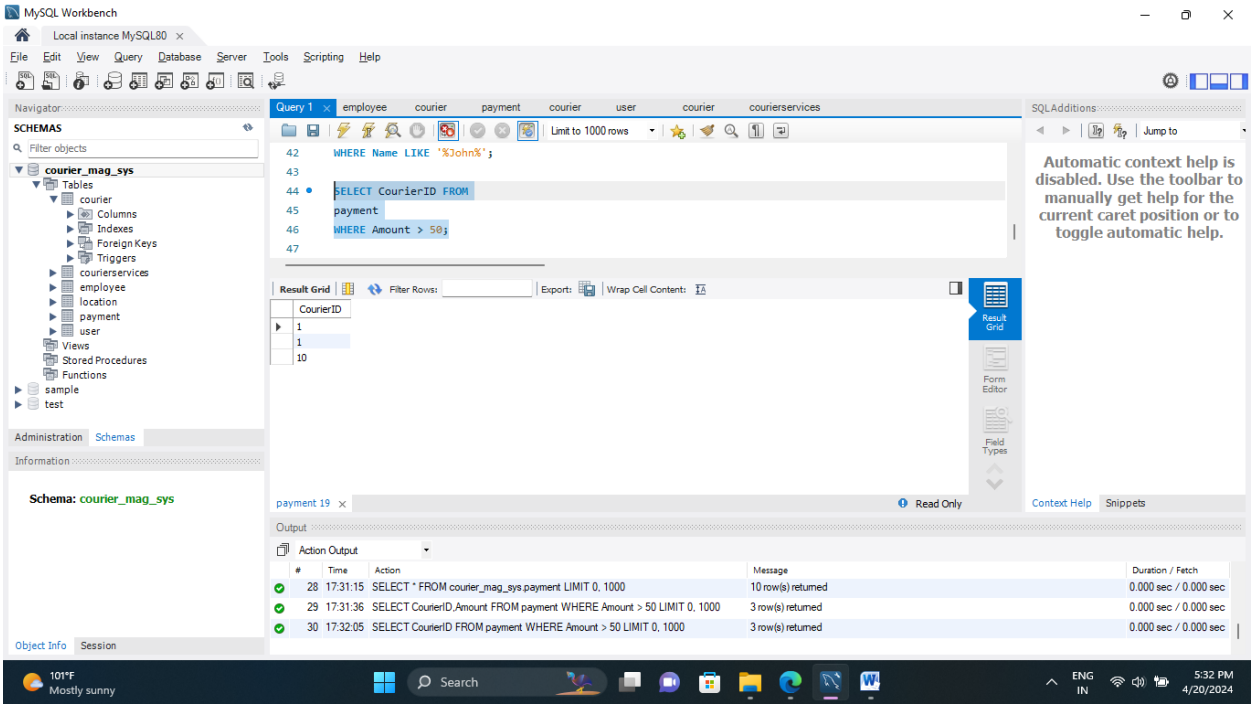


13. Retrieve all courier records with payments greater than $50.

SELECT CourierID FROM

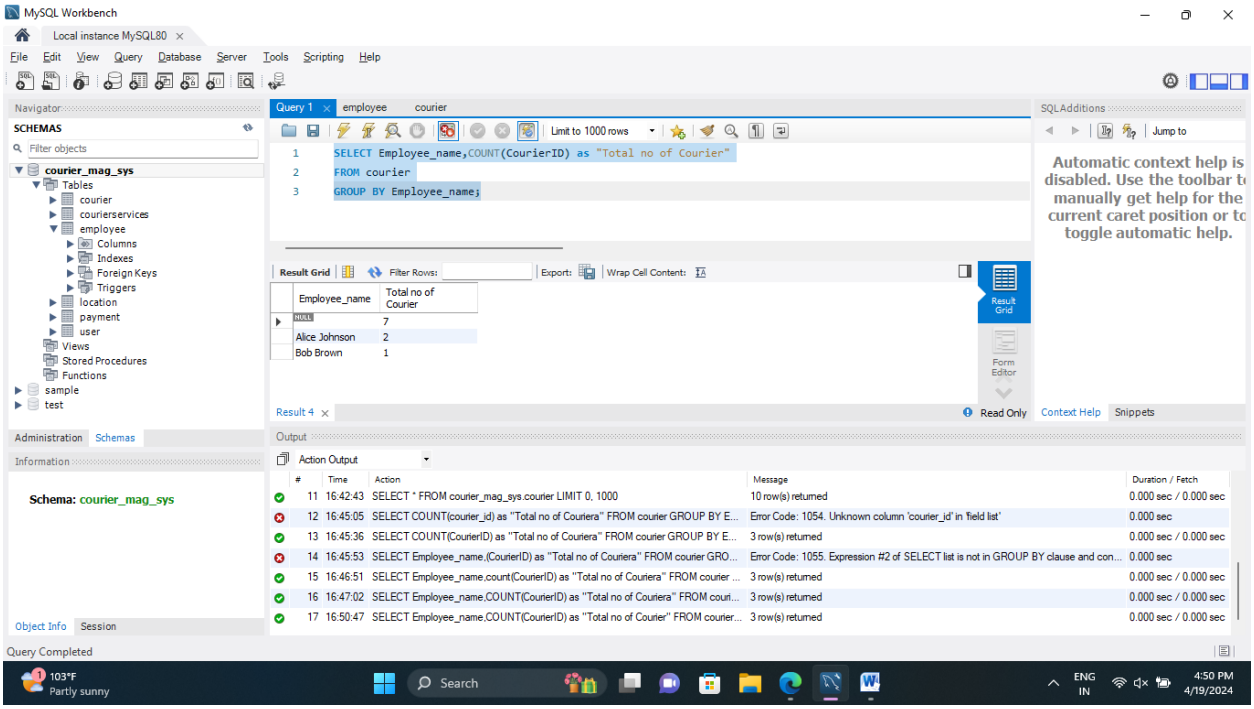
payment

WHERE Amount > 50;



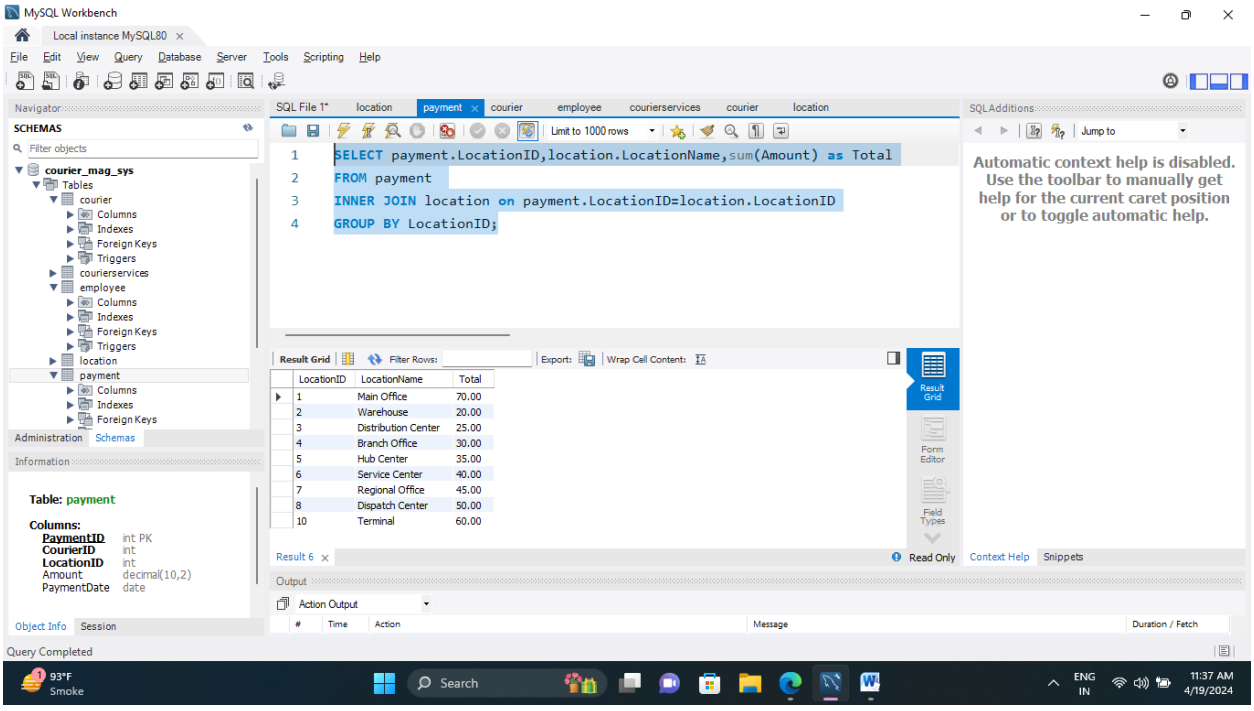
14. Find the total number of couriers handled by each employee.

```
SELECT Employee_name,COUNT(CourierID) as "Total no of Courier"
FROM courier
GROUP BY Employee_name;
```



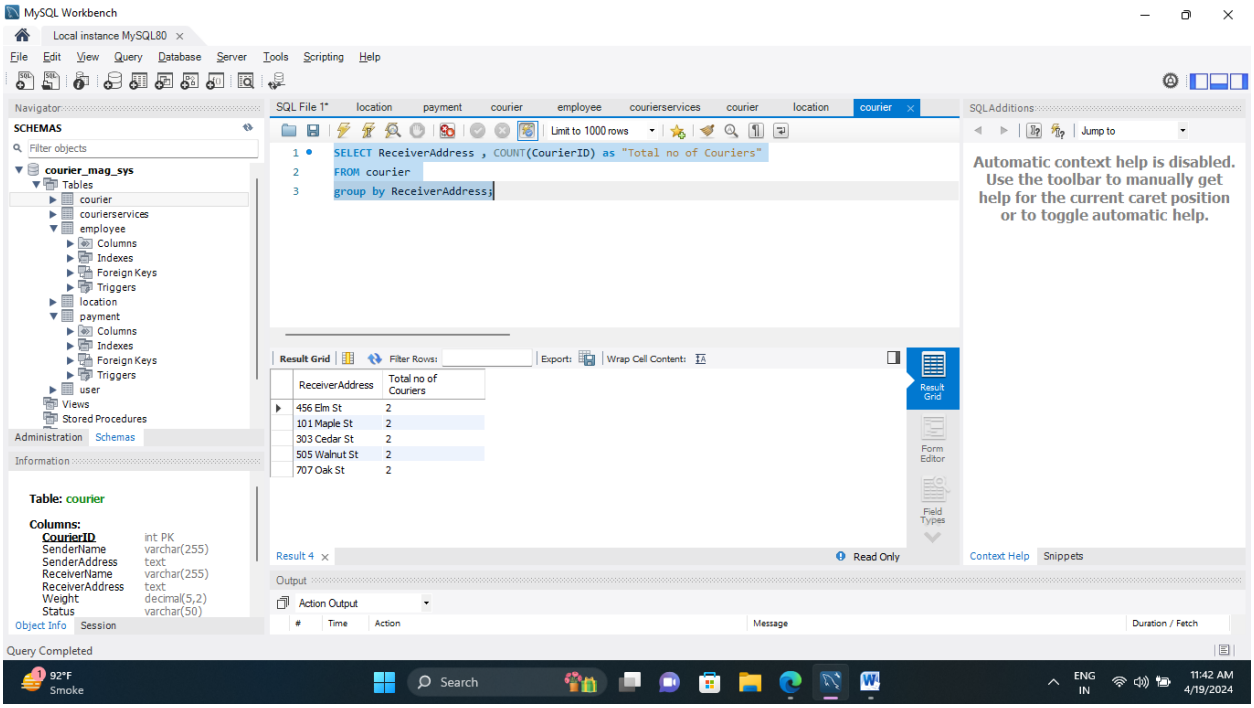
15. Calculate the total revenue generated by each location

```
SELECT payment.LocationID,location.LocationName,sum(Amount) as Total
FROM payment
INNER JOIN location on payment.LocationID=location.LocationID
GROUP BY LocationID;
```



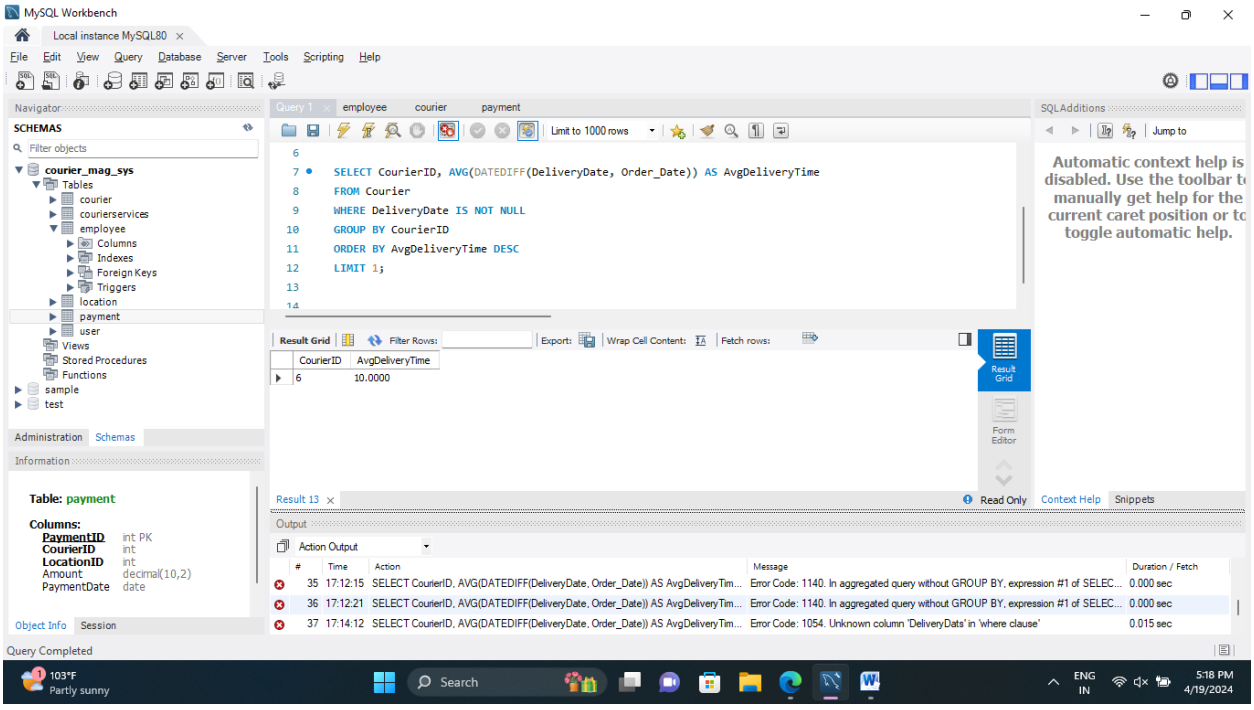
16. Find the total number of couriers delivered to each location.

```
SELECT ReceiverAddress , COUNT(CourierID) as "Total no of Couriers"
FROM courier
group by ReceiverAddress;
```



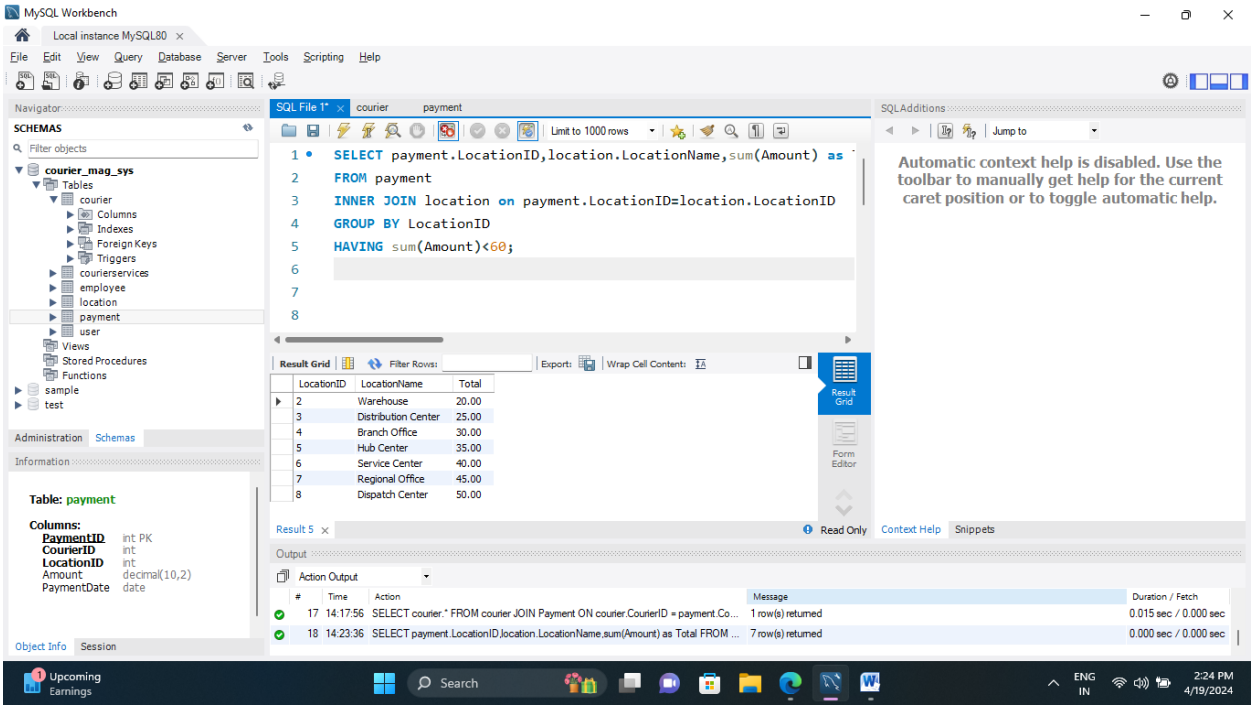
17. Find the courier with the highest average delivery time:

```
SELECT CourierID, AVG(DATEDIFF(DeliveryDate, Order_Date)) AS AvgDeliveryTime
FROM Courier
WHERE DeliveryDate IS NOT NULL
GROUP BY CourierID
ORDER BY AvgDeliveryTime DESC
LIMIT 1;
```



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

```
SELECT payment.LocationID,location.LocationName,sum(Amount) as Total
FROM payment
INNER JOIN location on payment.LocationID=location.LocationID
GROUP BY LocationID
HAVING sum(Amount)<60;
```



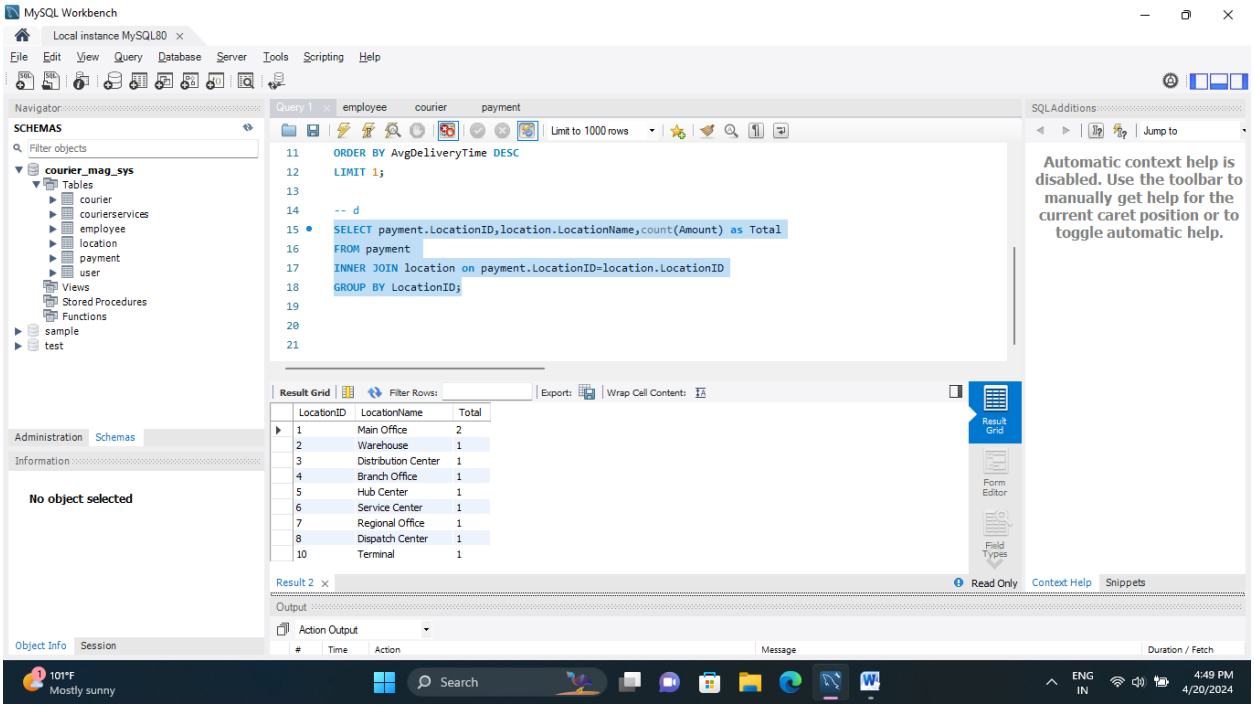
19. Calculate Total Payments per Location

SELECT payment.LocationID,location.LocationName,count(Amount) as Total

FROM payment

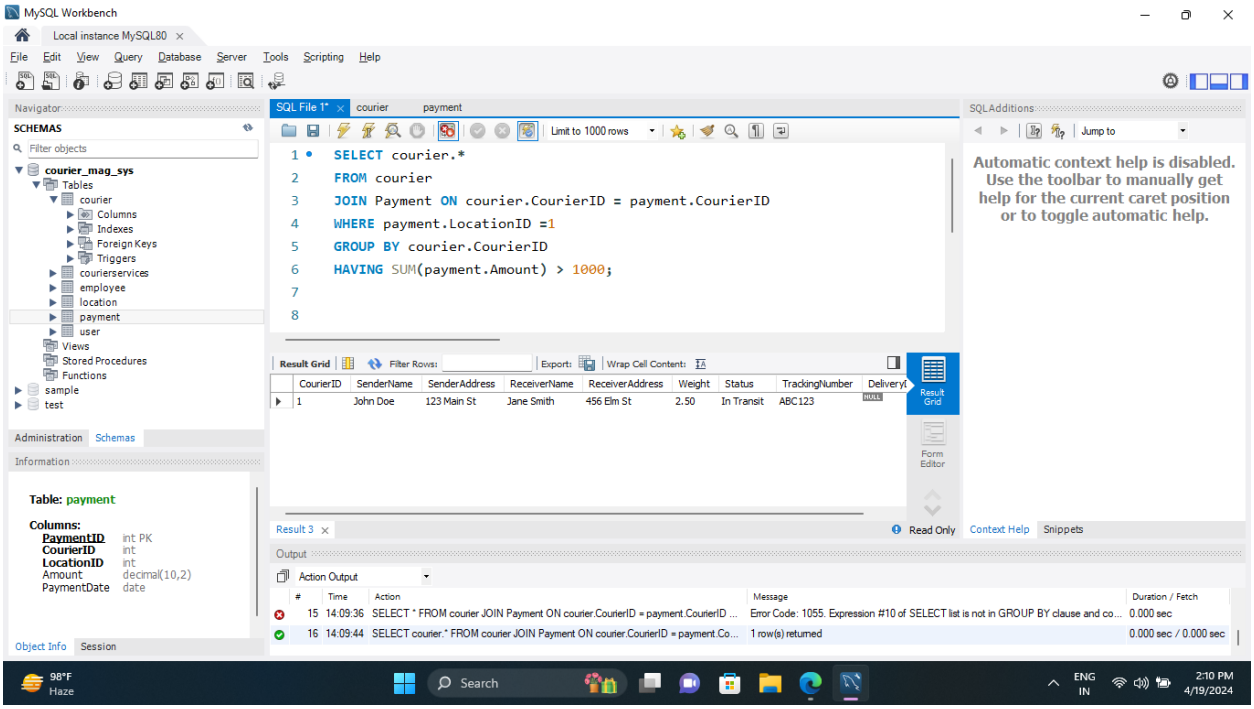
INNER JOIN location on payment.LocationID=location.LocationID

GROUP BY LocationID;



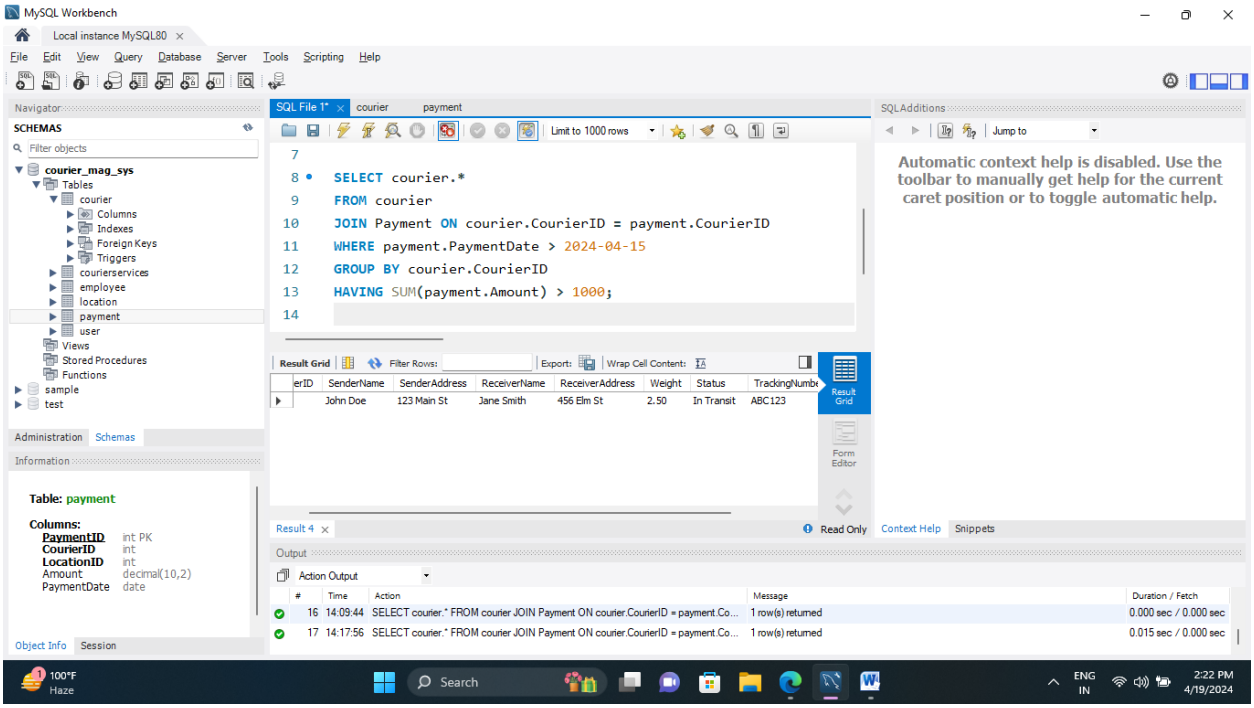
20. Retrieve couriers who have received payments totaling more than $1000 in a specific location (LocationID = X):

```
SELECT courier.*
FROM courier
INNER JOIN Payment ON courier.CourierID = payment.CourierID
WHERE payment.LocationID =1
GROUP BY courier.CourierID
HAVING SUM(payment.Amount) > 1000;
```



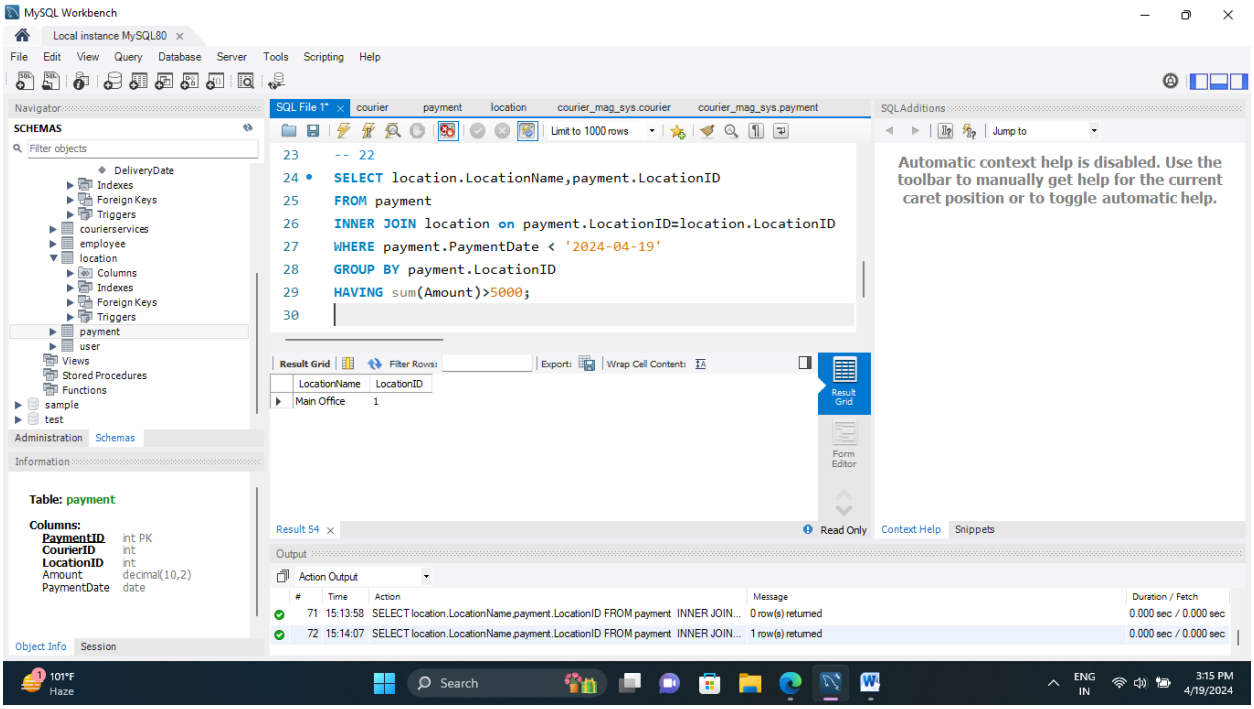
21. Retrieve couriers who have received payments totaling more than $1000 after a certain date (PaymentDate > 'YYYY-MM-DD'):

```
SELECT courier.*
FROM courier
JOIN Payment ON courier.CourierID = payment.CourierID
WHERE payment.PaymentDate > 2024-04-15
GROUP BY courier.CourierID
HAVING SUM(payment.Amount) > 1000;
```



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

```
SELECT location.LocationName,payment.LocationID
FROM payment
INNER JOIN location on payment.LocationID=location.LocationID
WHERE payment.PaymentDate < '2024-04-19'
GROUP BY payment.LocationID
HAVING sum(Amount)>5000;
```

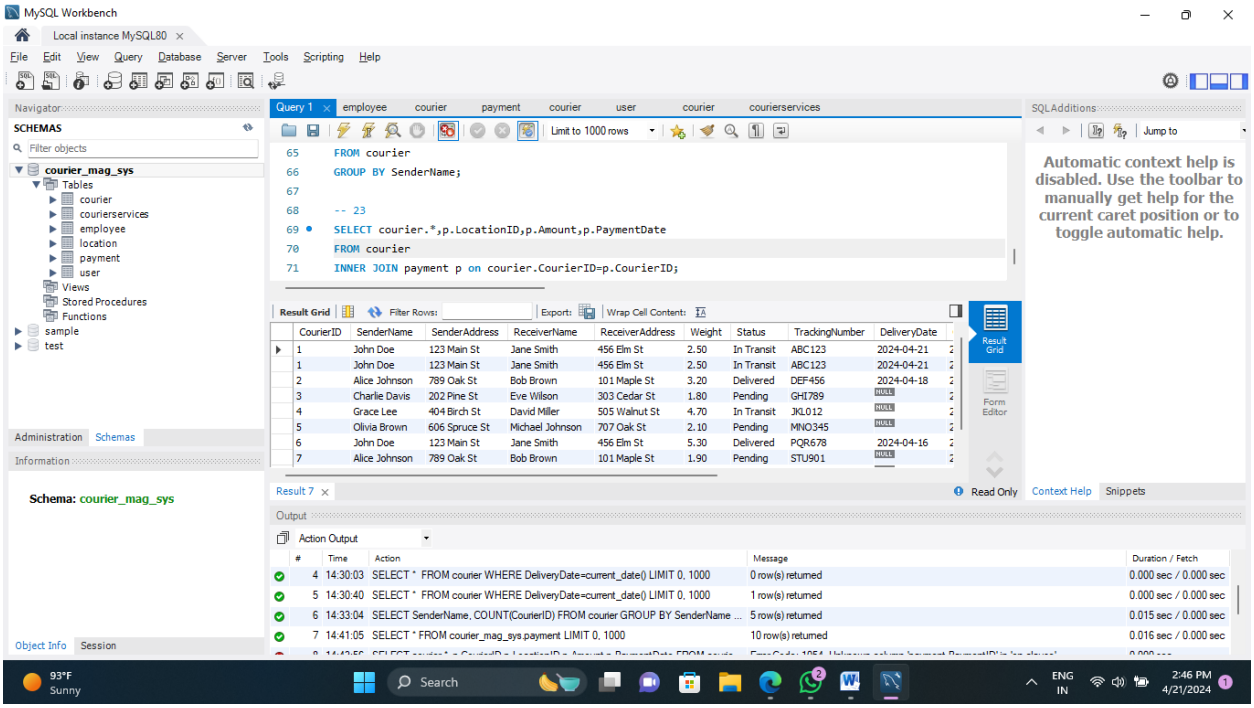


23. Retrieve Payments with Courier Information

SELECT courier.*,p.LocationID,p.Amount,p.PaymentDate

FROM courier

INNER JOIN payment p on courier.CourierID=p.CourierID;

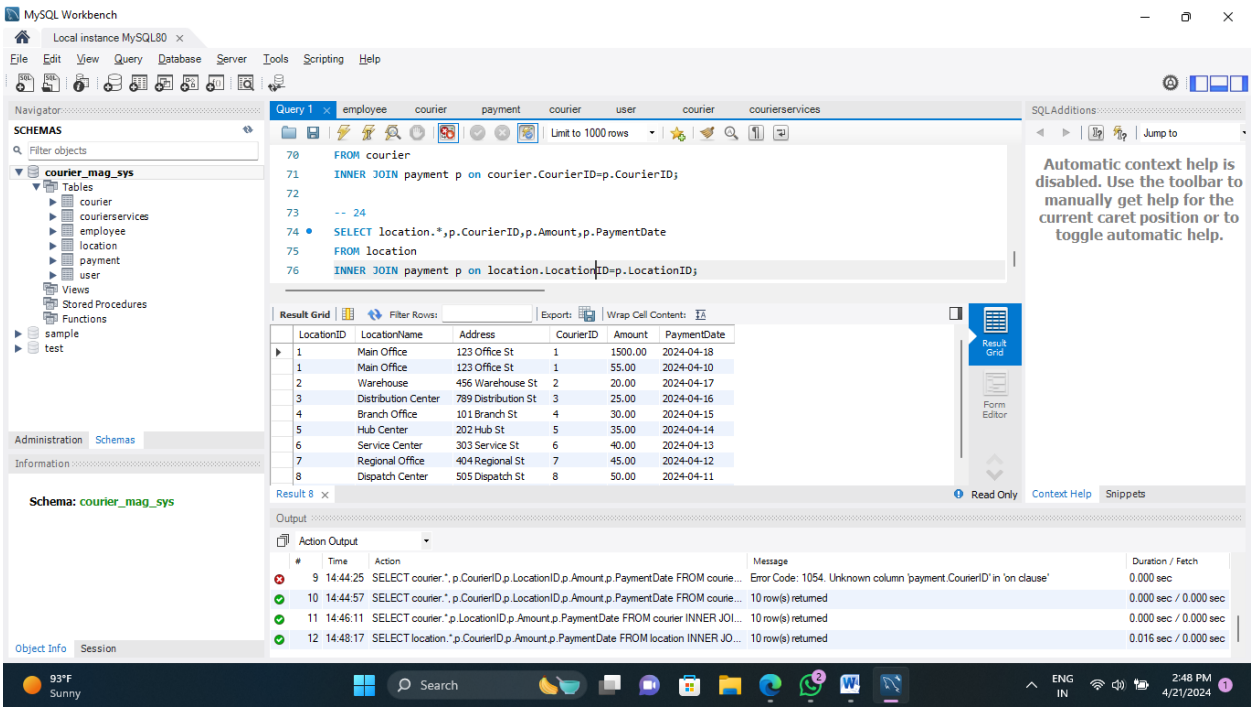


24. Retrieve Payments with Location Information

SELECT location.*,p.CourierID,p.Amount,p.PaymentDate

FROM location

INNER JOIN payment p on location.LocationID=p.LocationID;



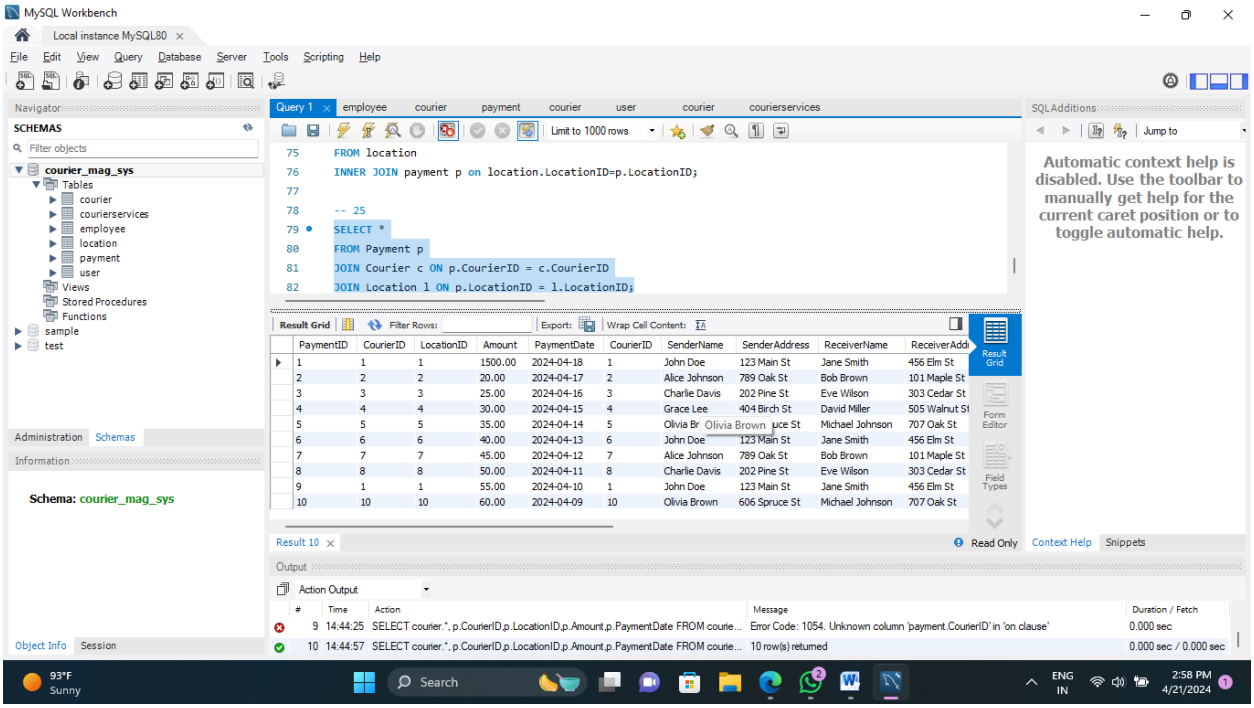
25. Retrieve Payments with Courier and Location Information

SELECT *

FROM Payment p

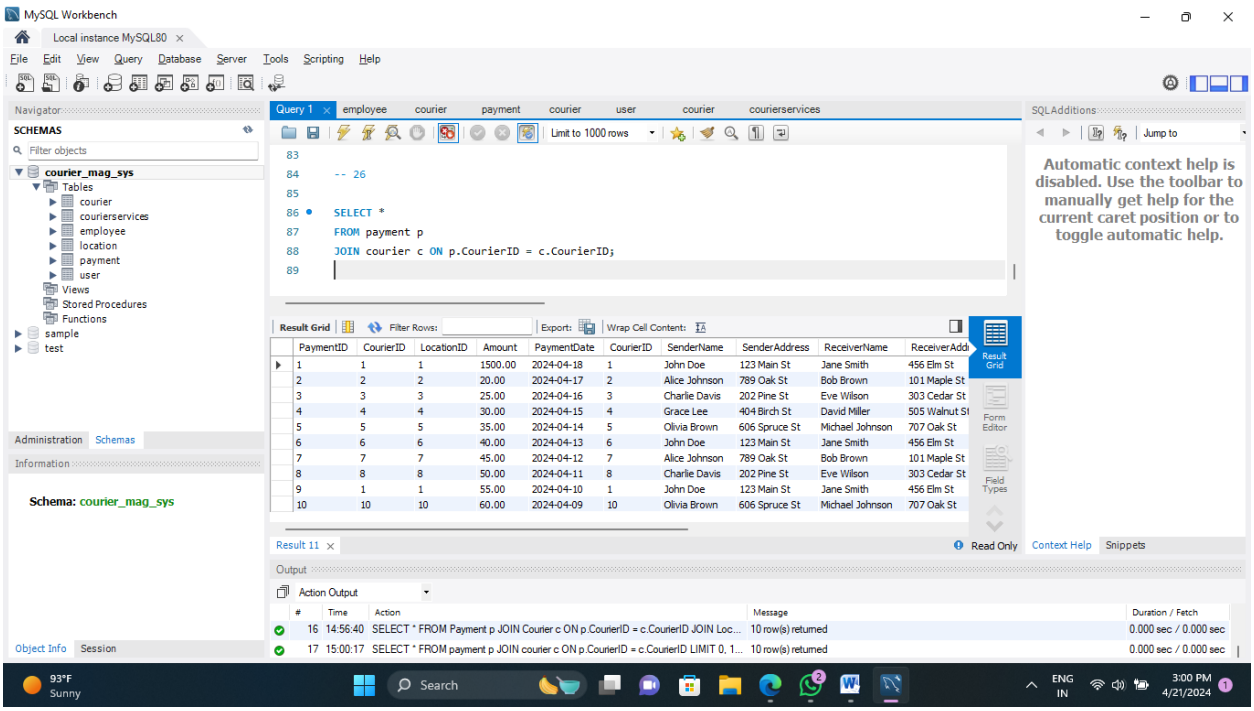
JOIN Courier c ON p.CourierID = c.CourierID

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



26. List all payments with courier details

```
SELECT *

FROM payment p

JOIN courier c ON p.CourierID = c.CourierID;
```

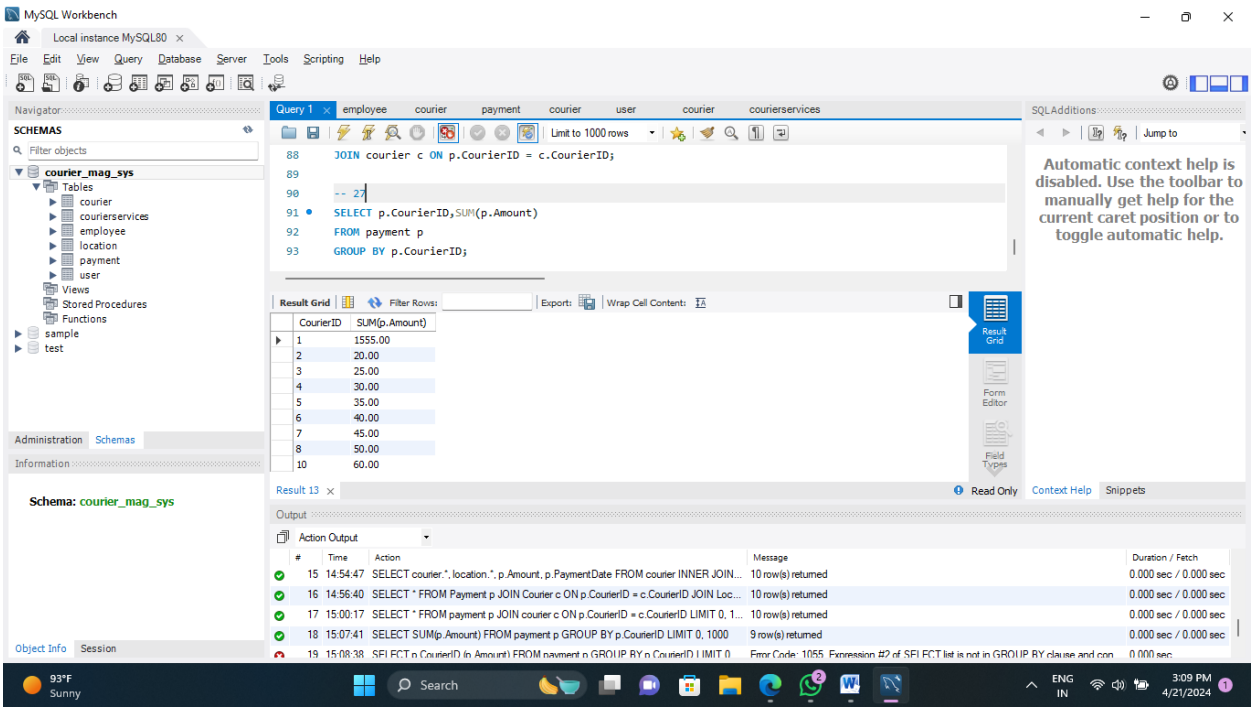


27. Total payments received for each courier

SELECT p.CourierID,SUM(p.Amount)

FROM payment p

GROUP BY p.CourierID;

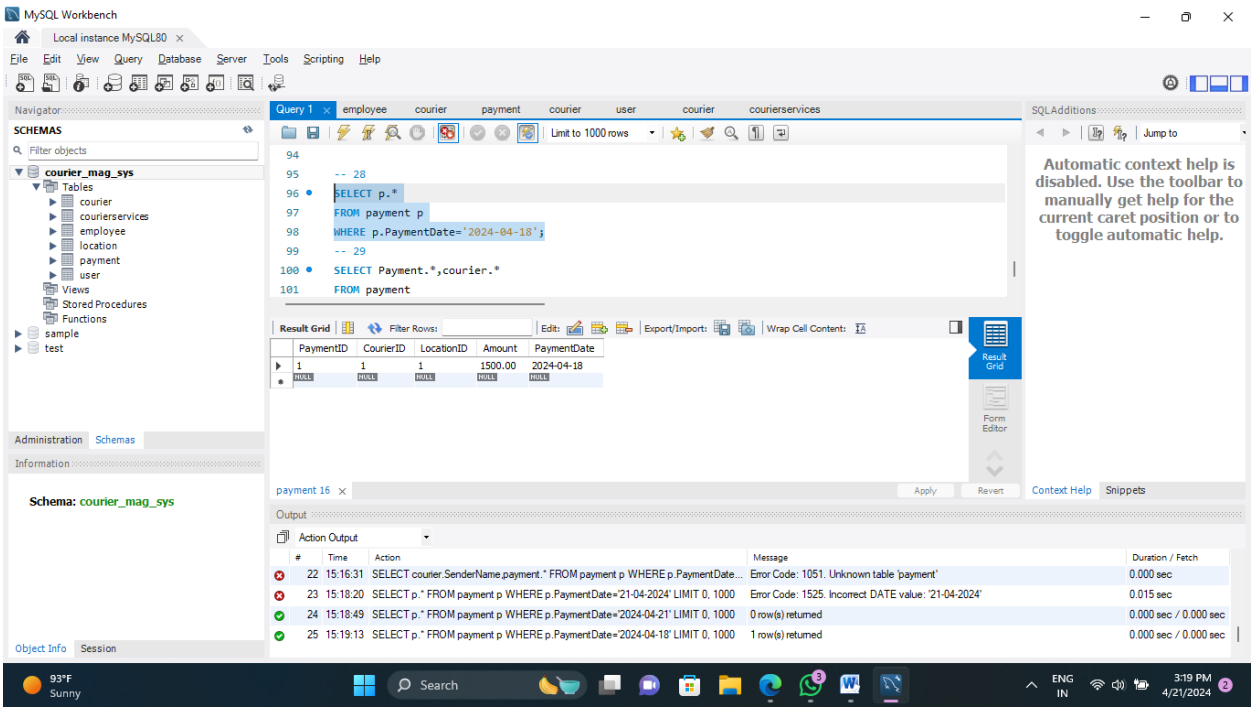


28. List payments made on a specific date

SELECT p.*

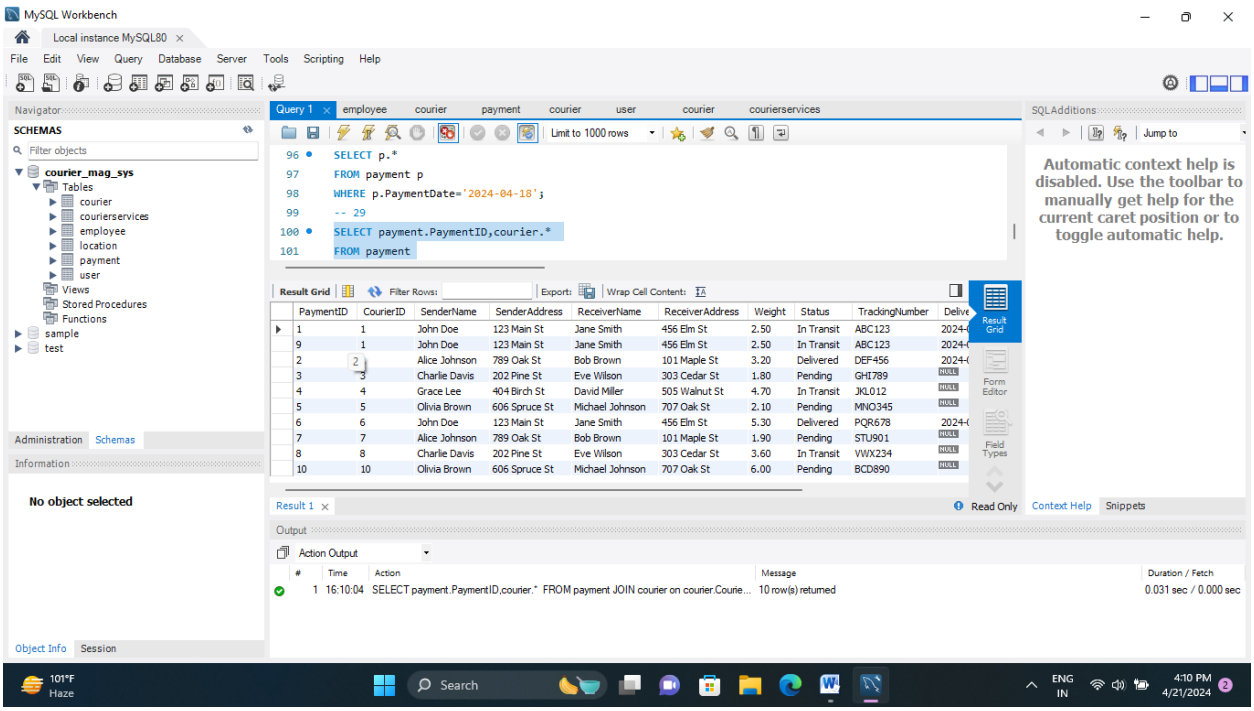
FROM payment p

WHERE p.PaymentDate='2024-04-18';



29. Get Courier Information for Each Payment

```
SELECT payment.PaymentID,courier.*

FROM payment

JOIN courier on courier.CourierID=payment.CourierID;
```

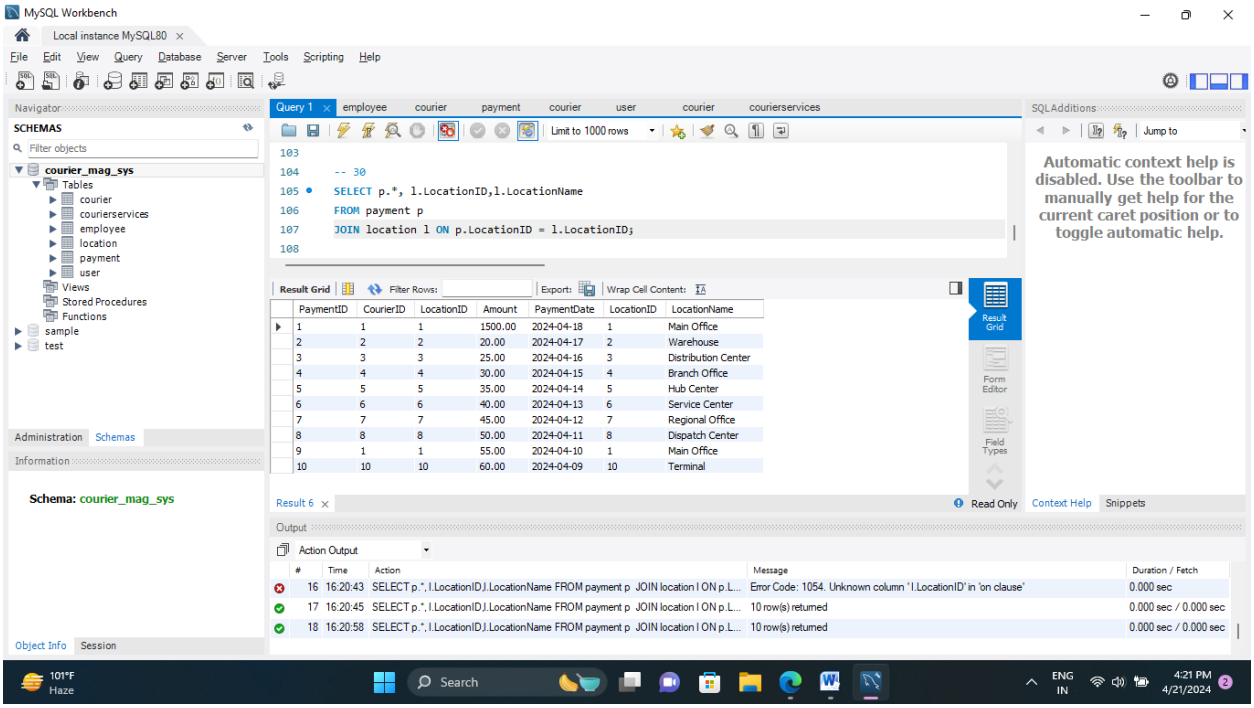


30. Get Payment Details with Location

SELECT p.*, l.LocationID,l.LocationName

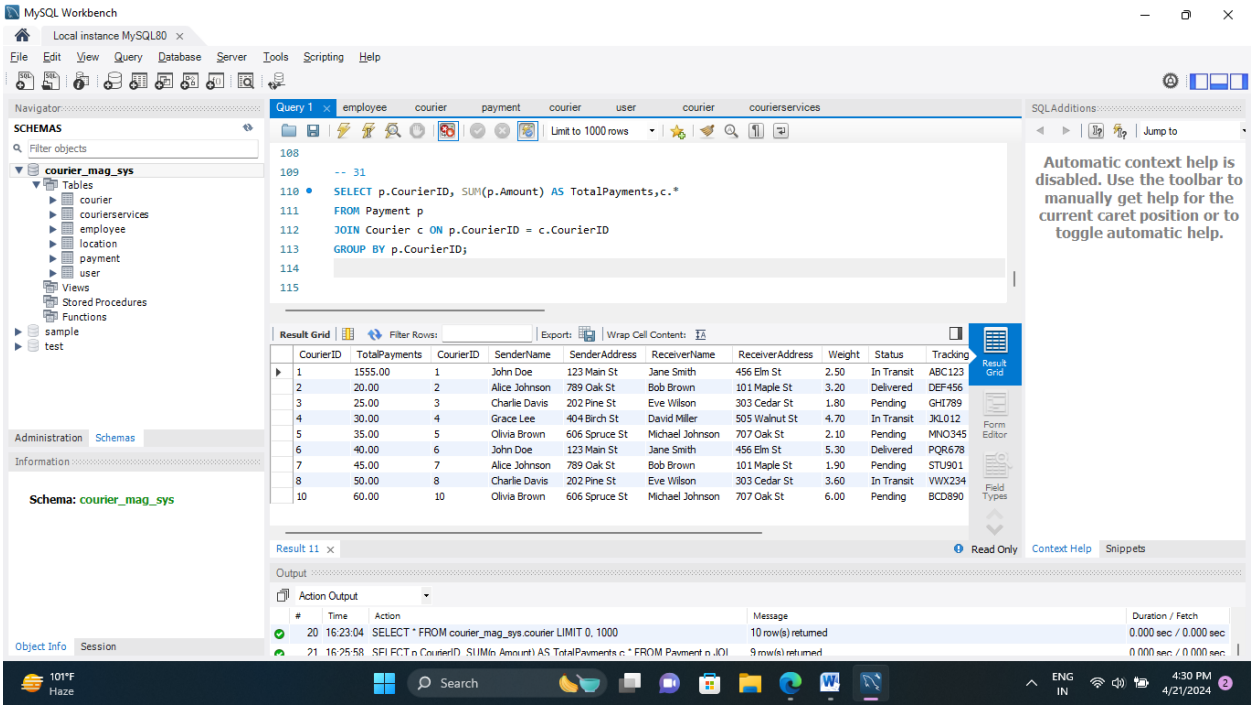
FROM payment p

JOIN location l ON p.LocationID = l.LocationID;



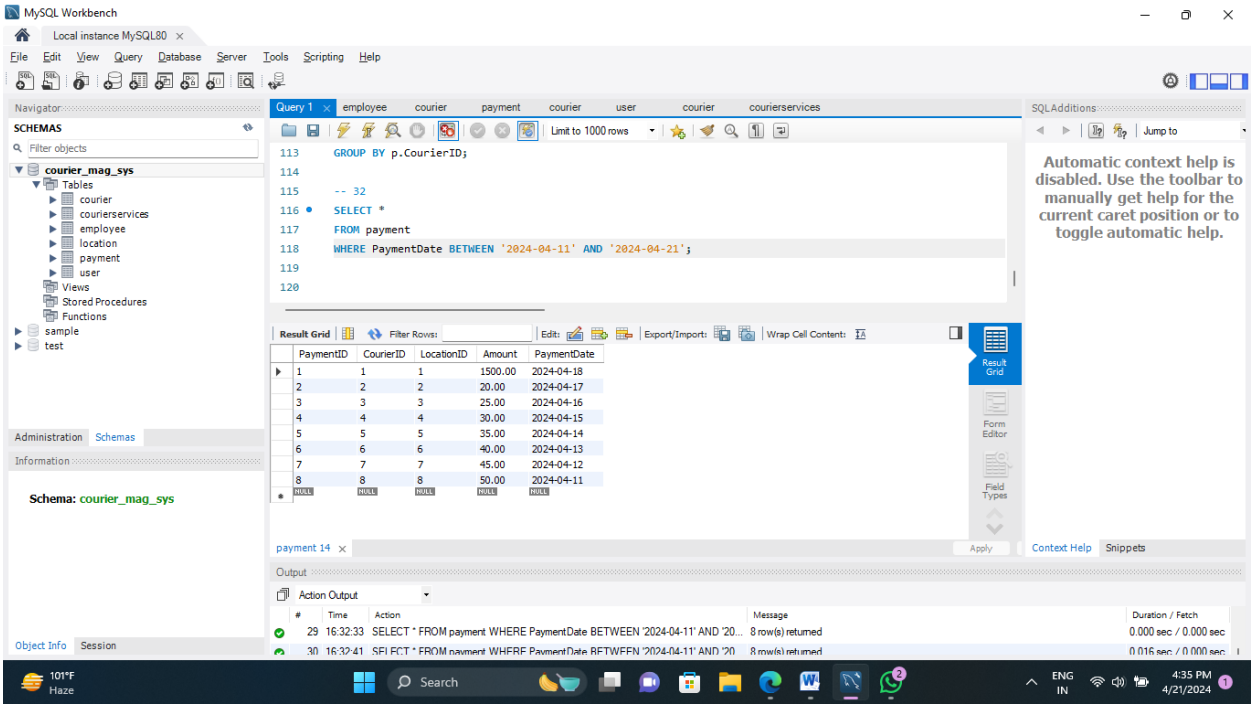
31. Calculating Total Payments for Each Courier

```
SELECT p.CourierID, SUM(p.Amount) AS TotalPayments,c.*
FROM Payment p
JOIN Courier c ON p.CourierID = c.CourierID
GROUP BY p.CourierID;
```



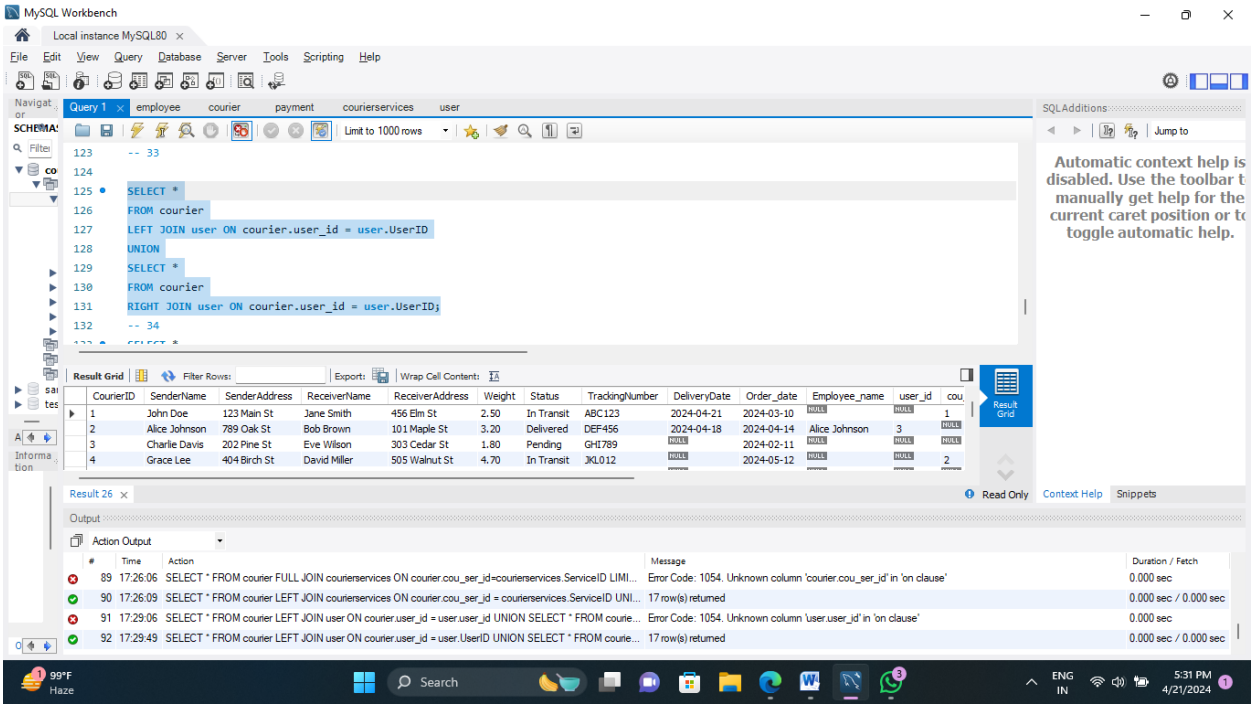
32. List Payments Within a Date Range

```
SELECT *

FROM payment

WHERE PaymentDate BETWEEN '2024-04-11' AND '2024-04-21';
```



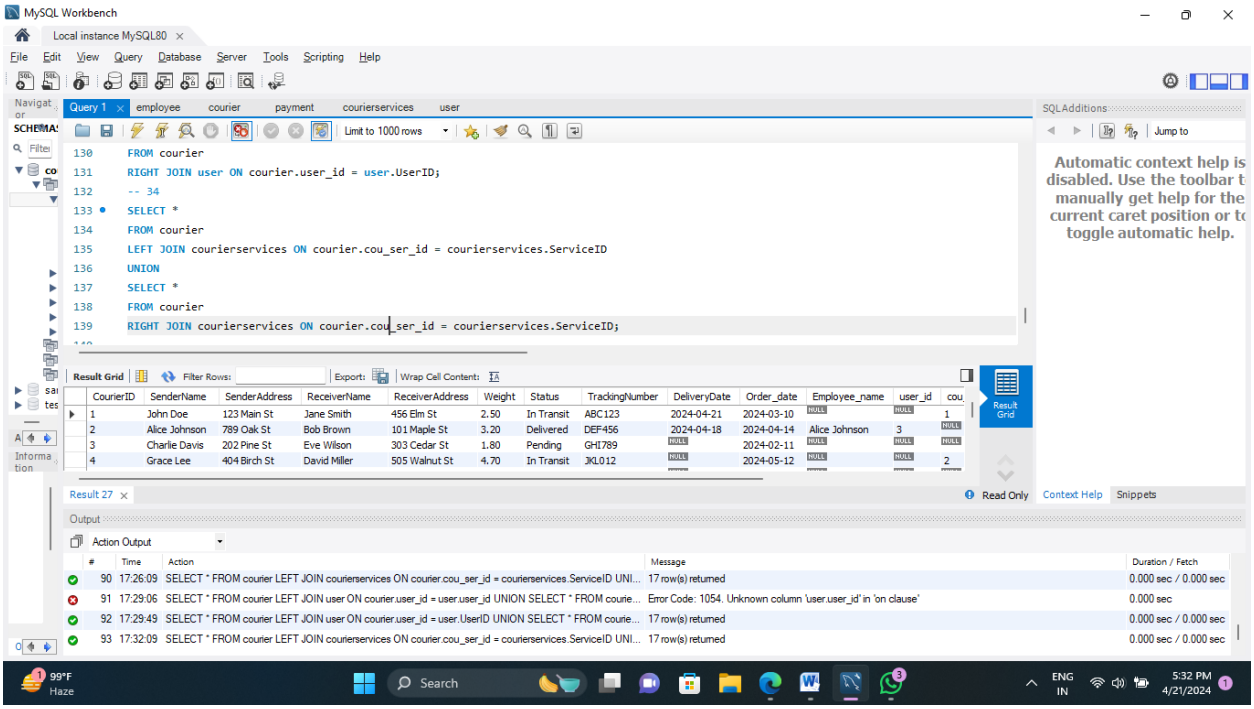
33. Retrieve a list of all users and their corresponding courier records, including cases where there are no matches on either side

```
SELECT *
FROM courier
LEFT JOIN user ON courier.user_id = user.UserID
UNION
SELECT *
FROM courier
RIGHT JOIN user ON courier.user_id = user.UserID;
```



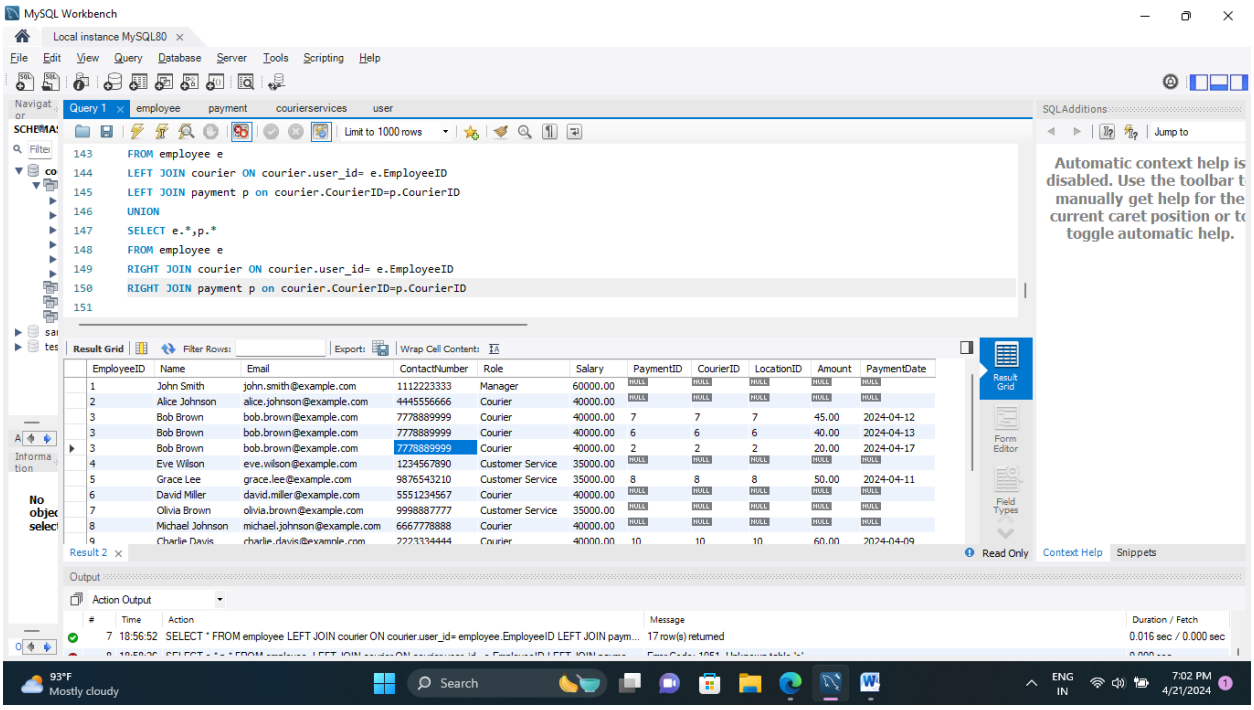
34. Retrieve a list of all couriers and their corresponding services, including cases where there are no matches on either side

```
SELECT *
FROM courier
LEFT JOIN courierservices ON courier.cou_ser_id = courierservices.ServiceID
UNION
SELECT *
FROM courier
RIGHT JOIN courierservices ON courier.cou_ser_id = courierservices.ServiceID;
```



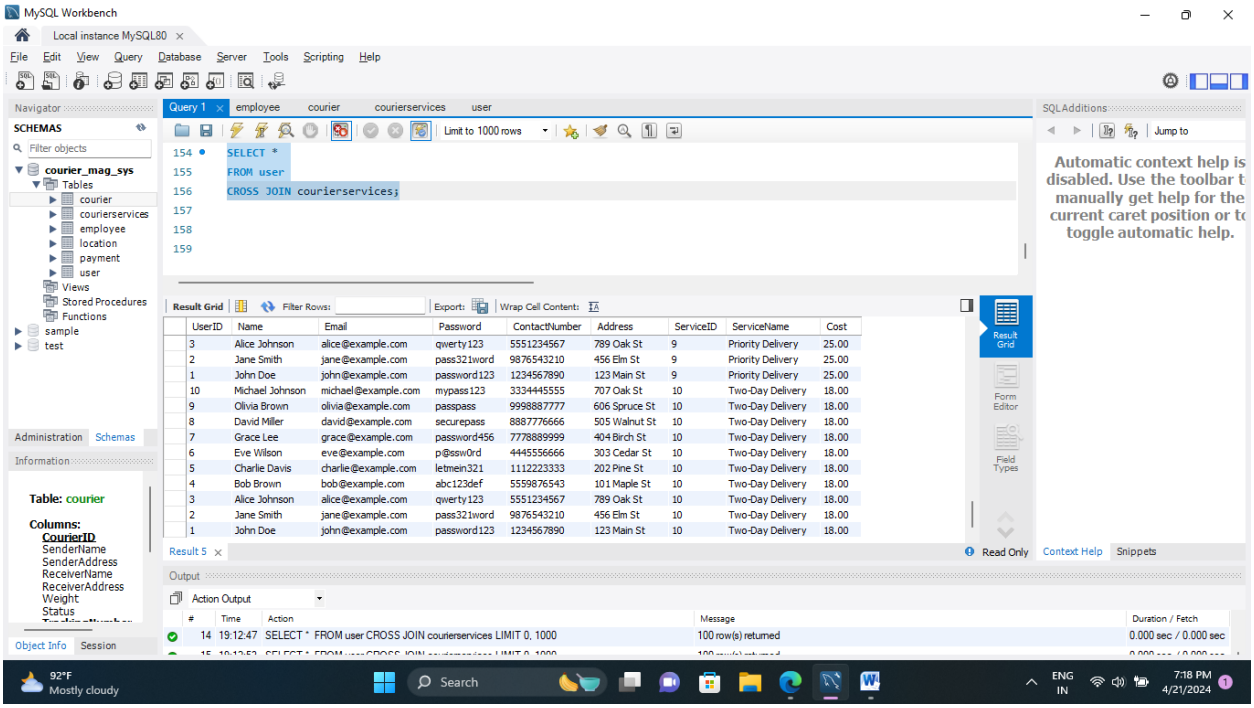
35. Retrieve a list of all employees and their corresponding payments, including cases where there are no matches on either side

```
SELECT e.*,p.*
FROM employee e
LEFT JOIN courier ON courier.emp_id= e.EmployeeID
LEFT JOIN payment p on courier.CourierID=p.CourierID
UNION
SELECT e.*,p.*
FROM employee e
RIGHT JOIN courier ON courier.emp_id= e.EmployeeID
RIGHT JOIN payment p on courier.CourierID=p.CourierID
```



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

```
SELECT *
FROM user
CROSS JOIN courierservices;
```

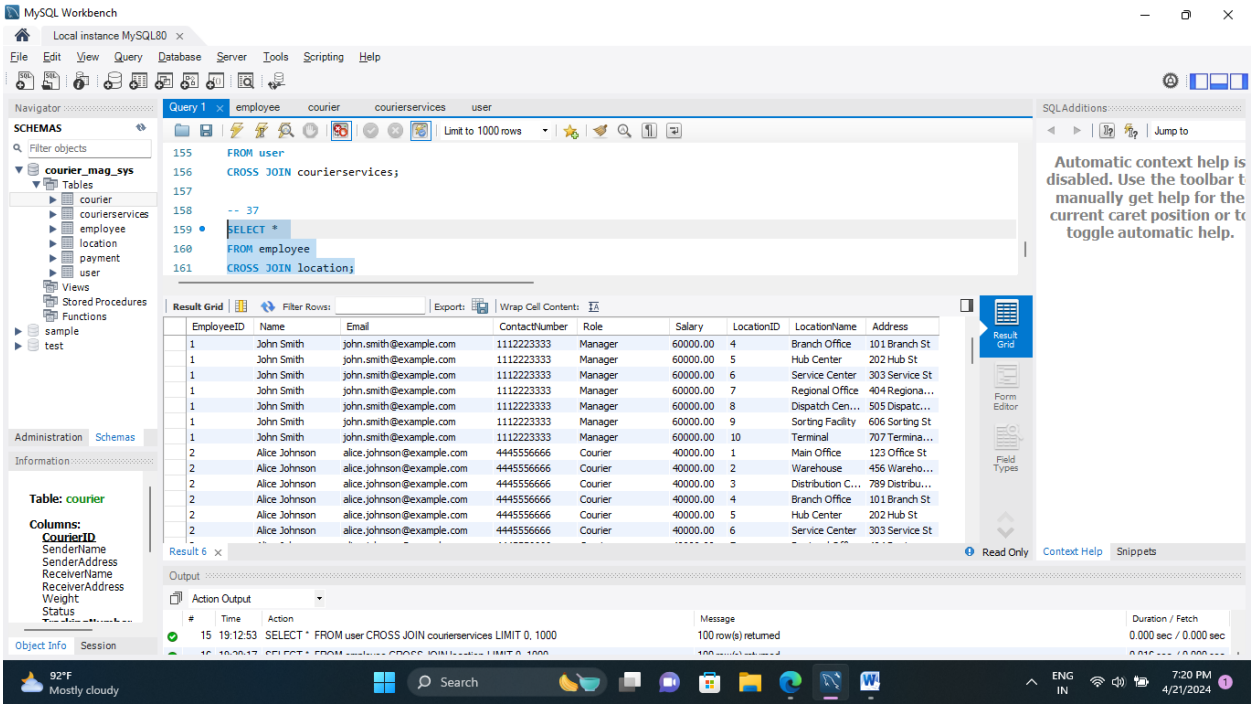


37. List all employees and all locations, showing all possible combinations:

SELECT *

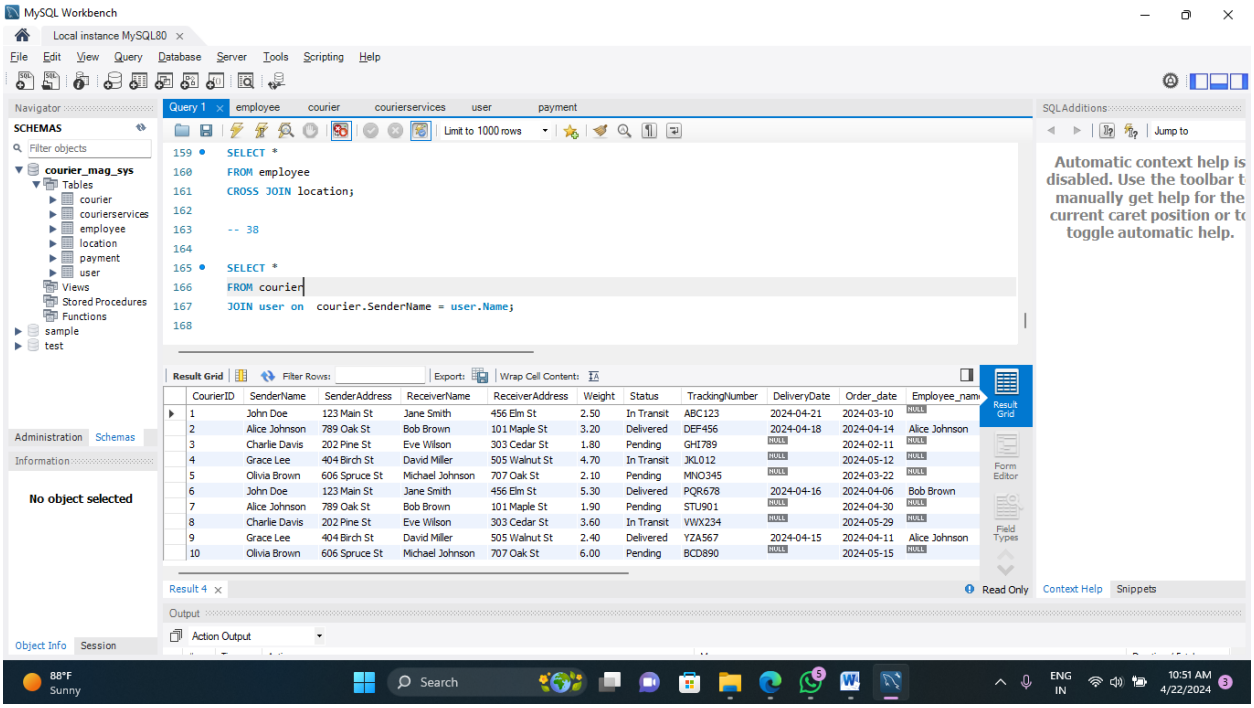
FROM employee

CROSS JOIN location;



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

```
SELECT *
FROM courier
LEFT JOIN user on  courier.SenderName = user.Name;
```

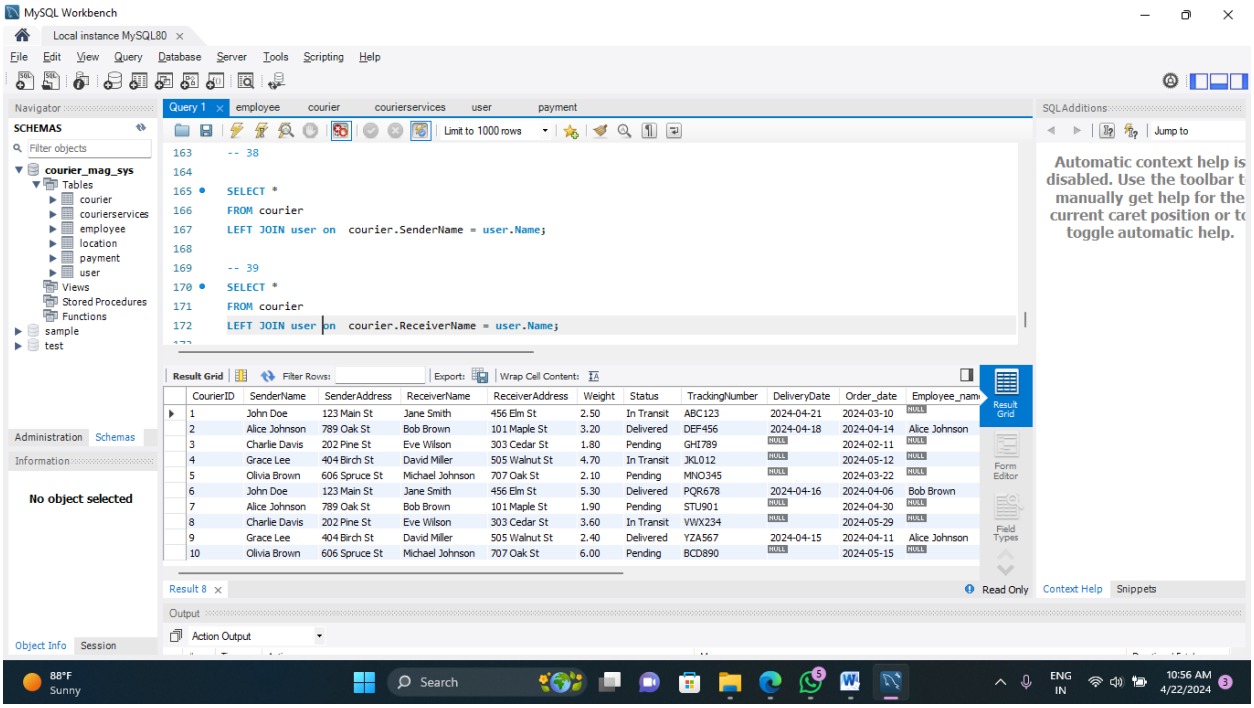


39. Retrieve a list of couriers and their corresponding receiver information (if available):

SELECT *

FROM courier

LEFT JOIN user on courier.ReceiverName = user.Name;

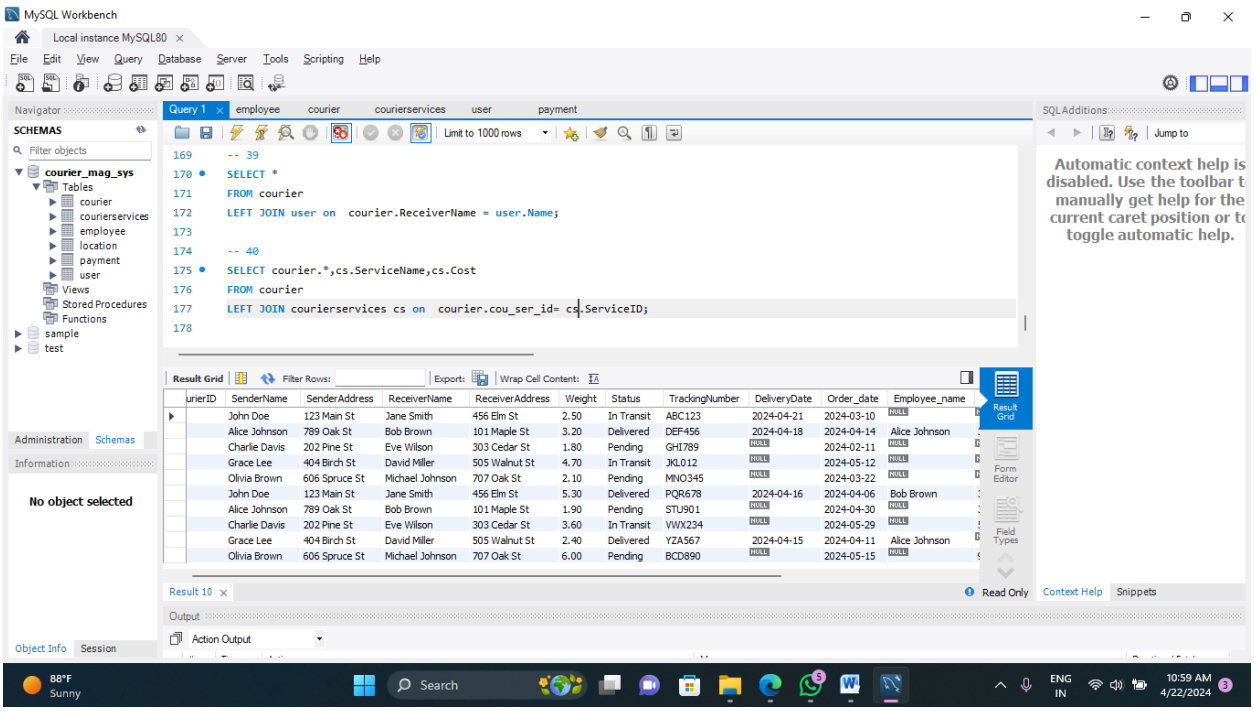


40. Retrieve a list of couriers along with the courier service details (if available):

SELECT courier.*,cs.ServiceName,cs.Cost

FROM courier

LEFT JOIN courierservices cs on courier.cou_ser_id= cs.ServiceID;



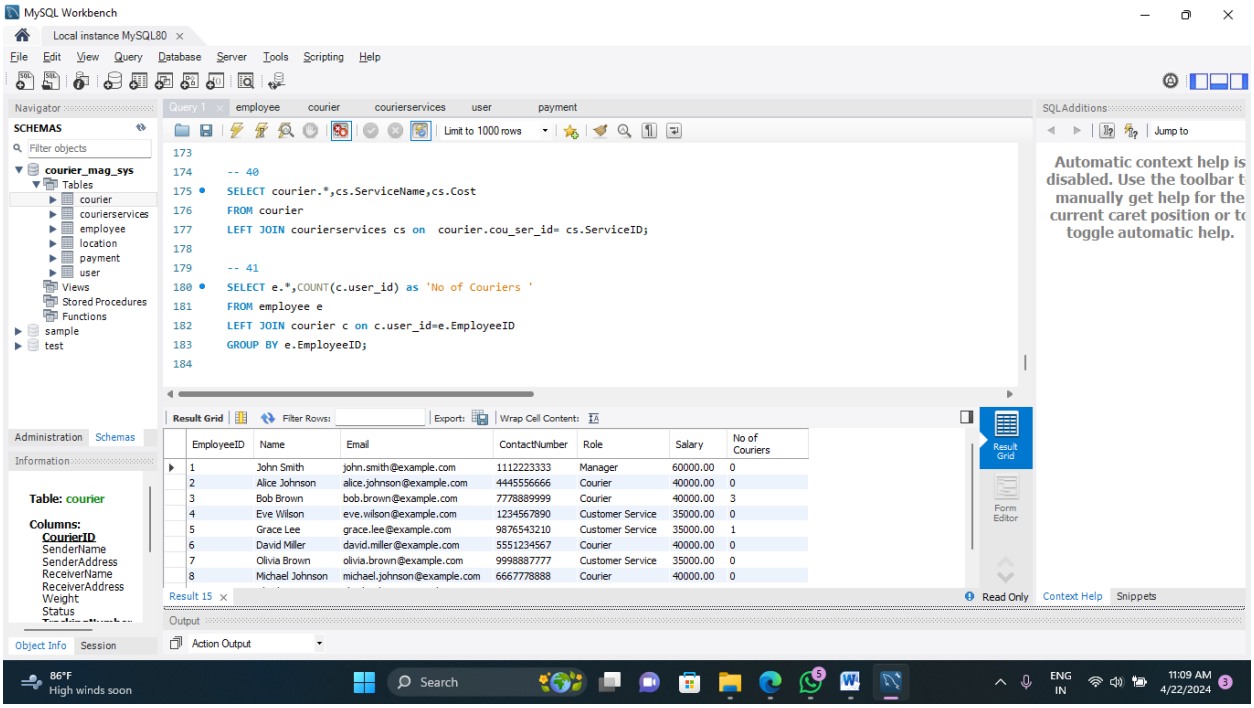
41. Retrieve a list of employees and the number of couriers assigned to each employee:

SELECT e.*,COUNT(c.emp_id) as 'No of Couriers '

FROM employee e

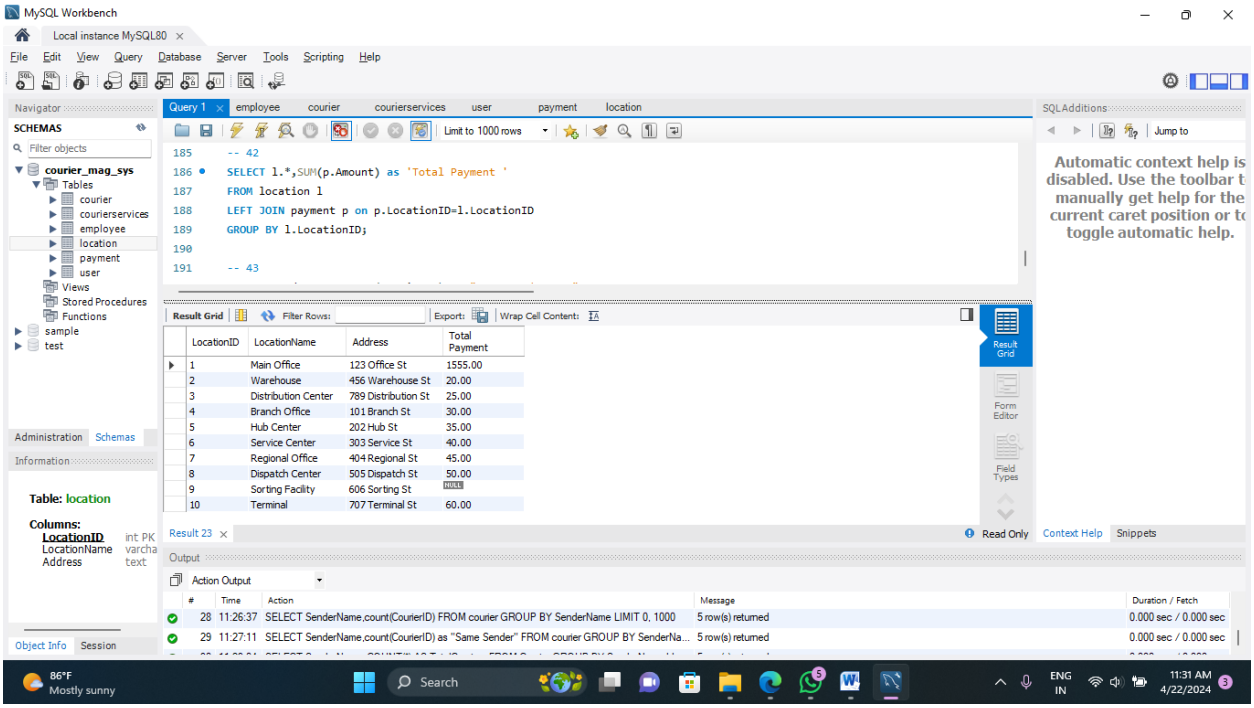
LEFT JOIN courier c on c.emp_id=e.EmployeeID

GROUP BY e.EmployeeID;



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

```
SELECT l.*,SUM(p.Amount) as 'Total Payment '
FROM location l
LEFT JOIN payment p on p.LocationID=l.LocationID
GROUP BY l.LocationID;
```

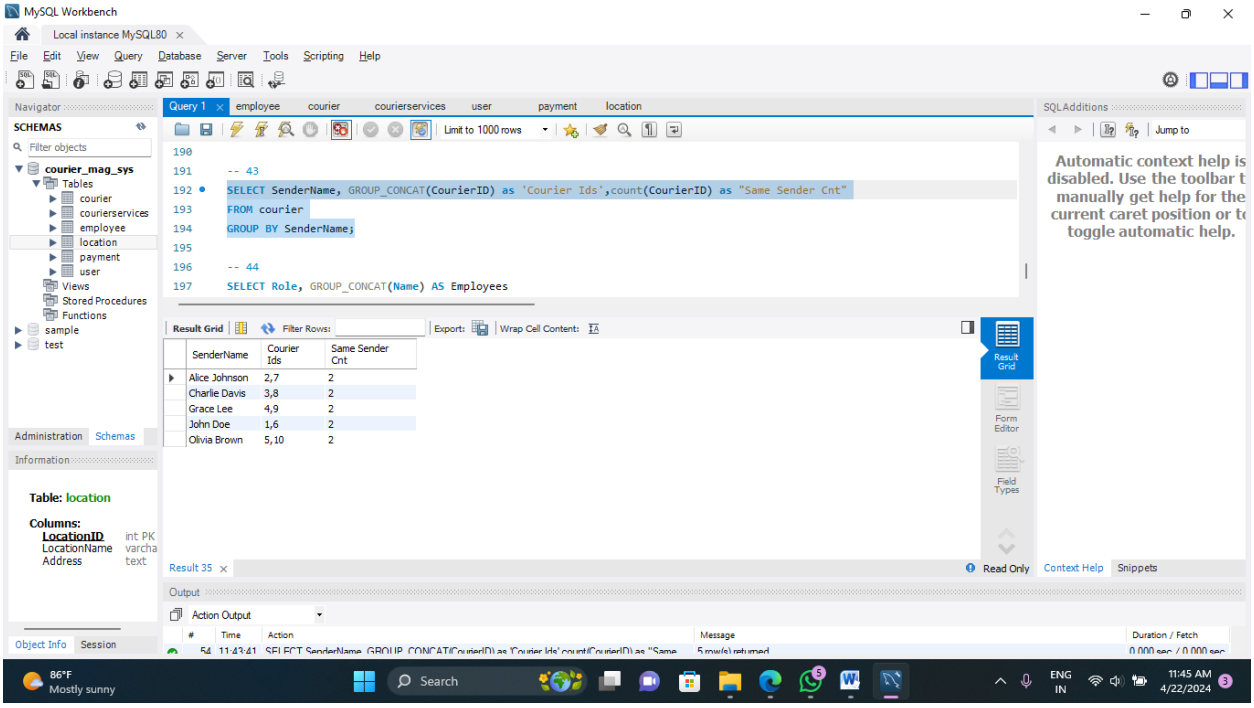


43. Retrieve all couriers sent by the same sender (based on SenderName).

SELECT SenderName, GROUP_CONCAT(CourierID) as 'Courier Ids',count(CourierID) as "Same Sender Cnt"

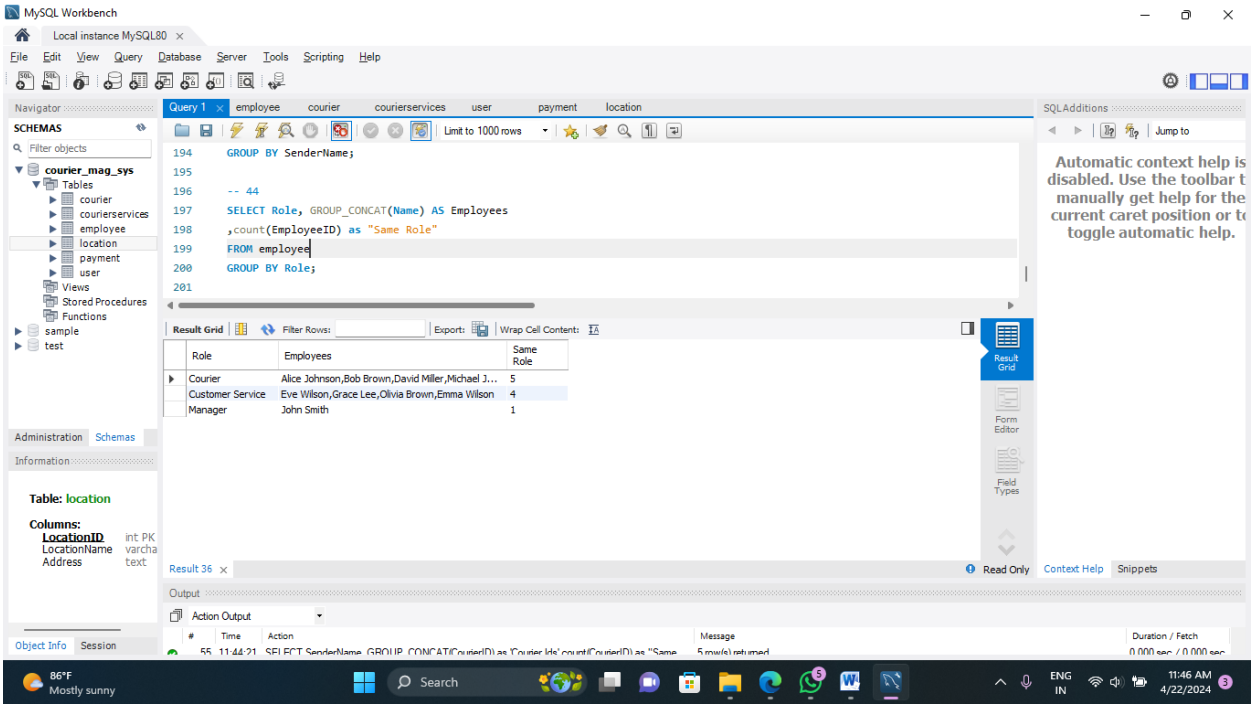
FROM courier

GROUP BY SenderName;



44. List all employees who share the same role.

```
SELECT Role, GROUP_CONCAT(Name) AS Employees
,count(EmployeeID) as "Same Role"
FROM employee
GROUP BY Role;
```

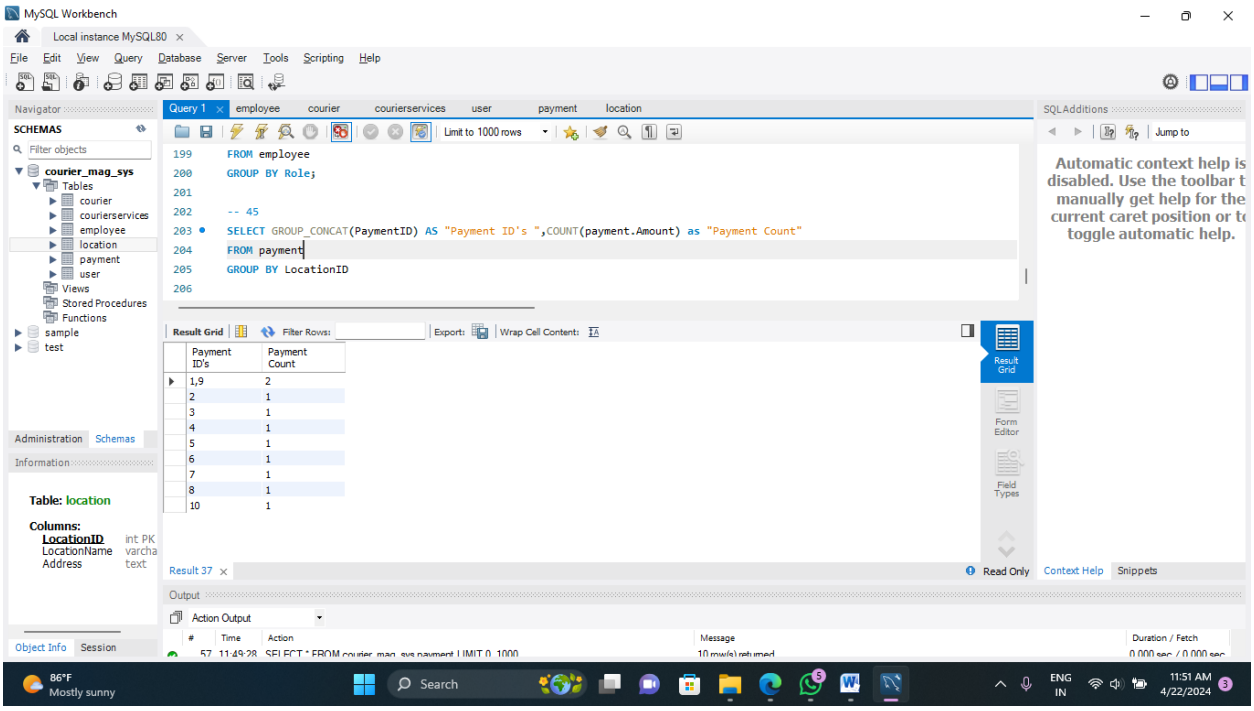


45. Retrieve all payments made for couriers sent from the same location.

SELECT GROUP_CONCAT(PaymentID) AS "Payment ID's ",COUNT(payment.Amount) as "Payment Count"

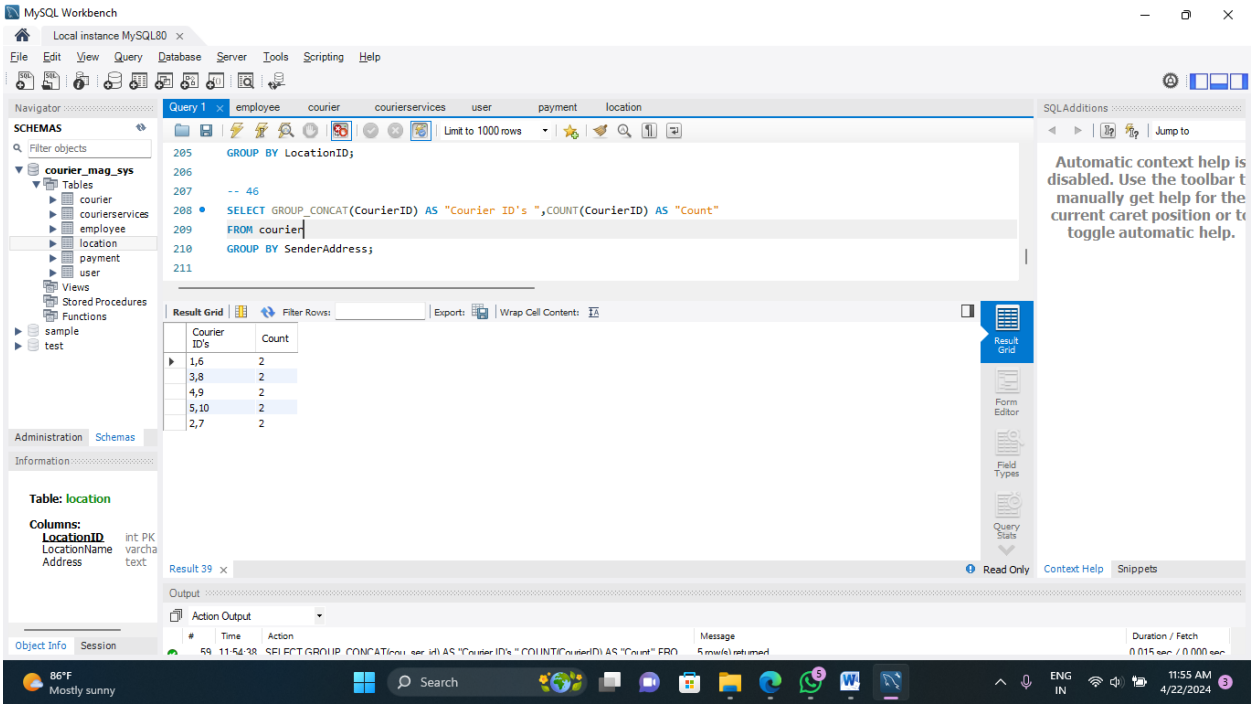
FROM payment

GROUP BY LocationID



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

```
SELECT GROUP_CONCAT(CourierID) AS "Courier ID's ",COUNT(CourierID) AS "Count"
FROM courier
GROUP BY SenderAddress;
```



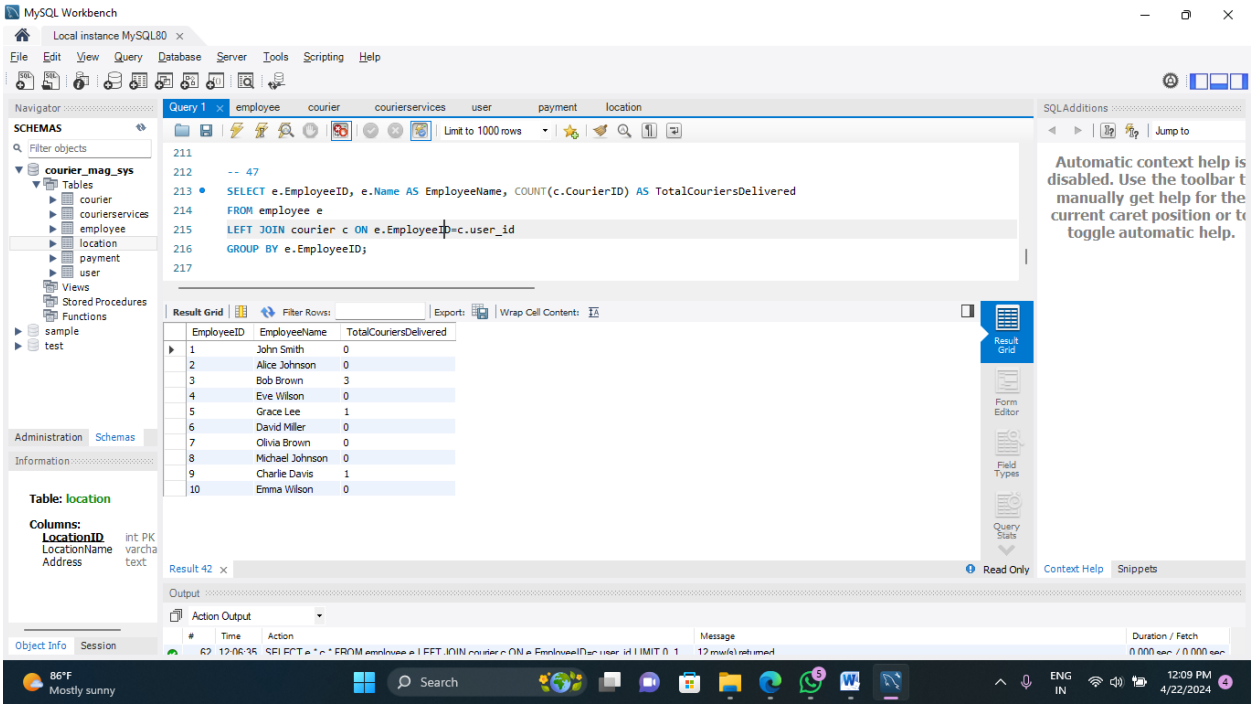
47. List employees and the number of couriers they have delivered:

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

FROM employee e

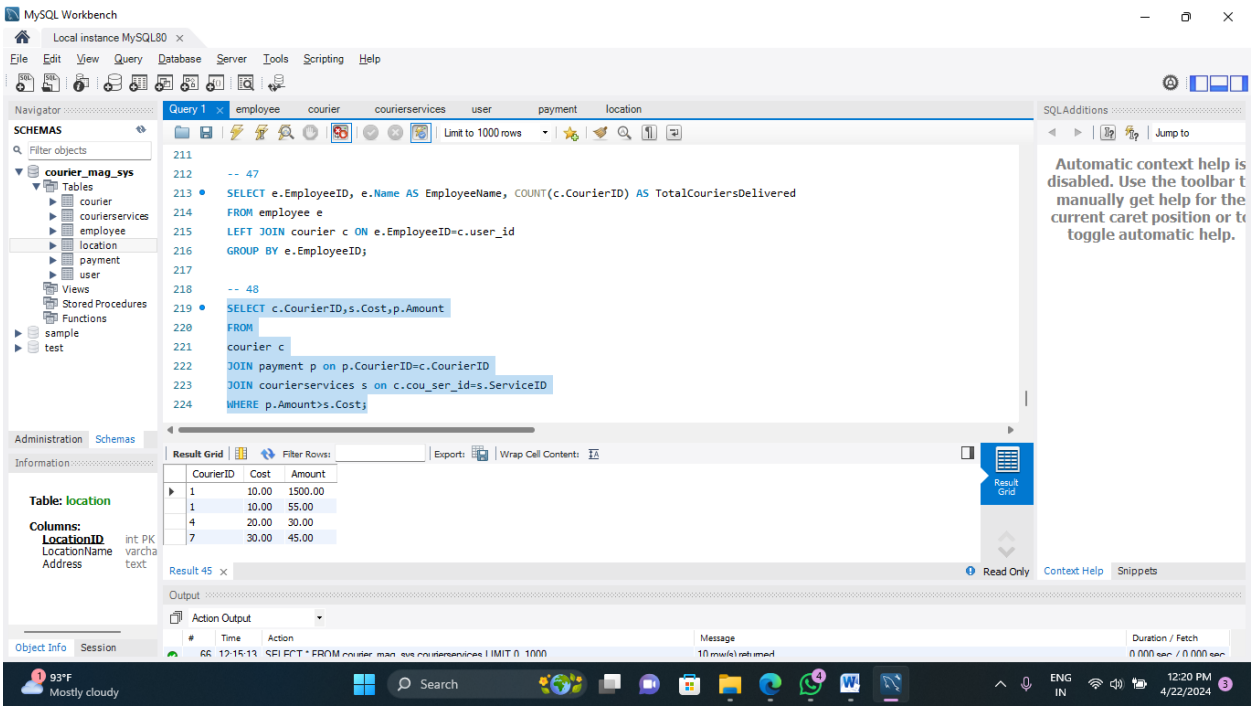
LEFT JOIN courier c ON e.EmployeeID=c.emp_id

GROUP BY e.EmployeeID;



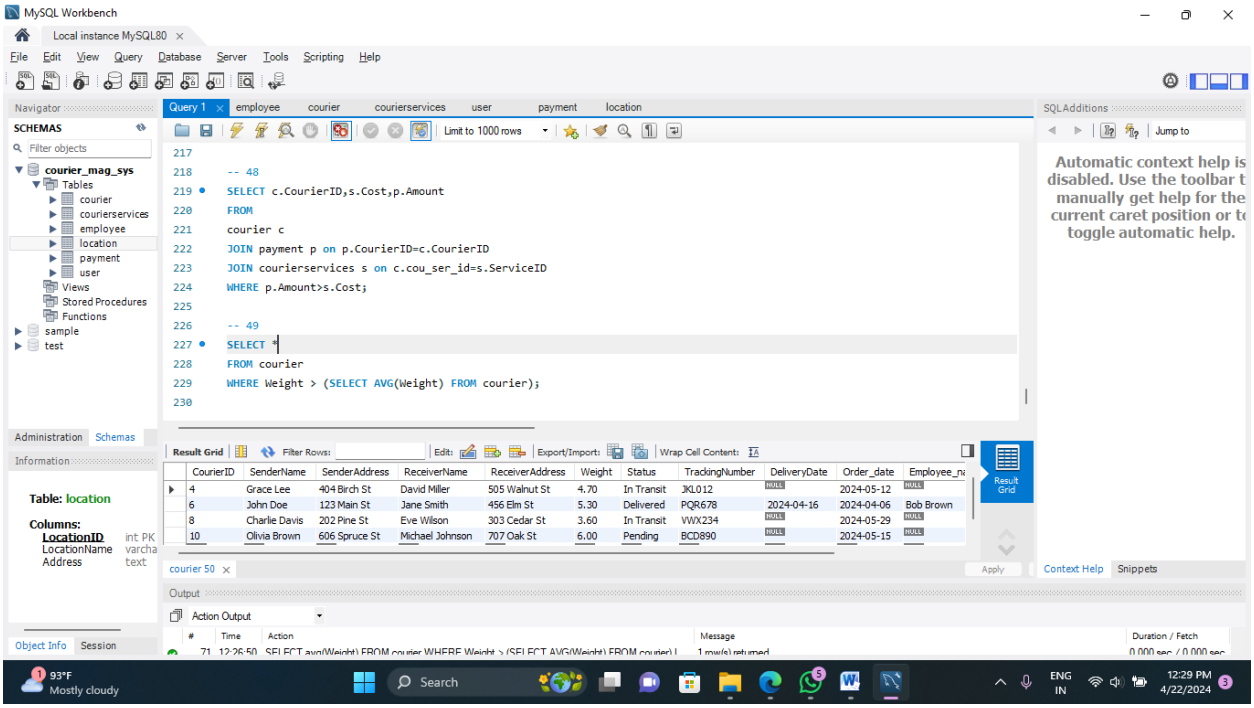
48. Find couriers that were paid an amount greater than the cost of their respective courier services

```
SELECT c.CourierID,s.Cost,p.Amount
FROM
courier c
JOIN payment p on p.CourierID=c.CourierID
JOIN courierservices s on c.cou_ser_id=s.ServiceID
WHERE p.Amount>s.Cost;
```



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

```
SELECT *

FROM courier

WHERE Weight > (SELECT AVG(Weight) FROM courier);
```

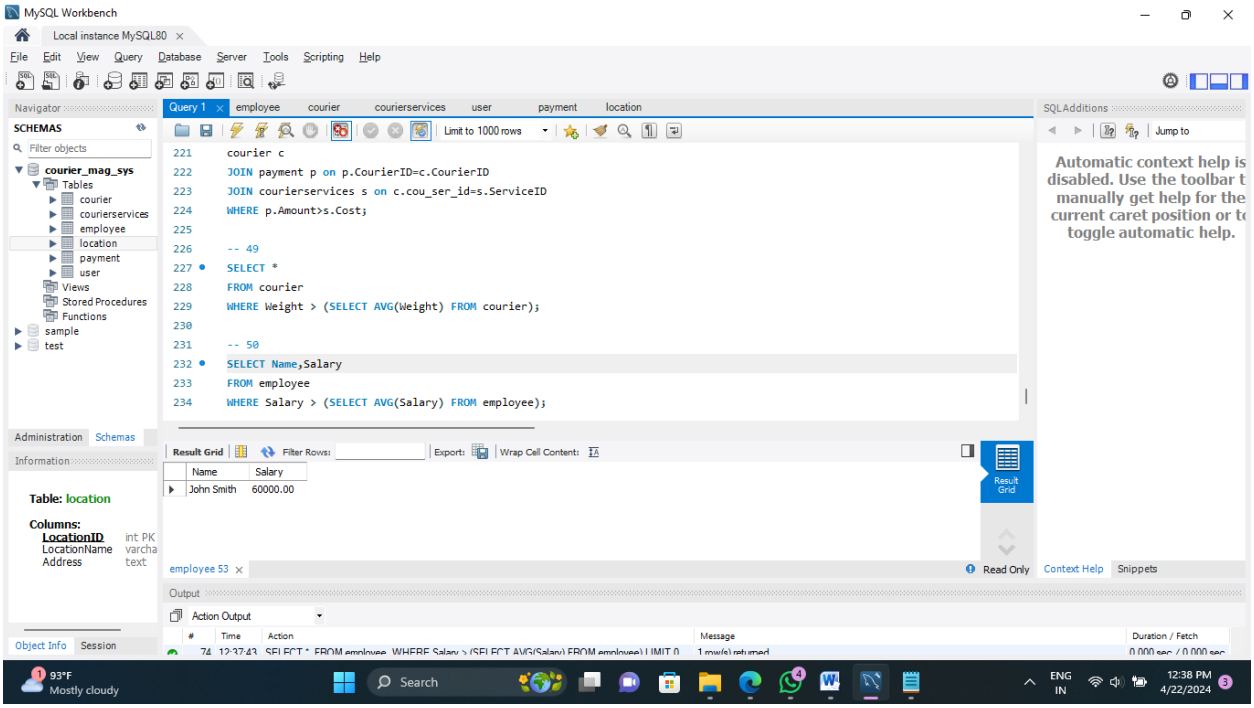


50. Find the names of all employees who have a salary greater than the average salary:

SELECT Name,Salary

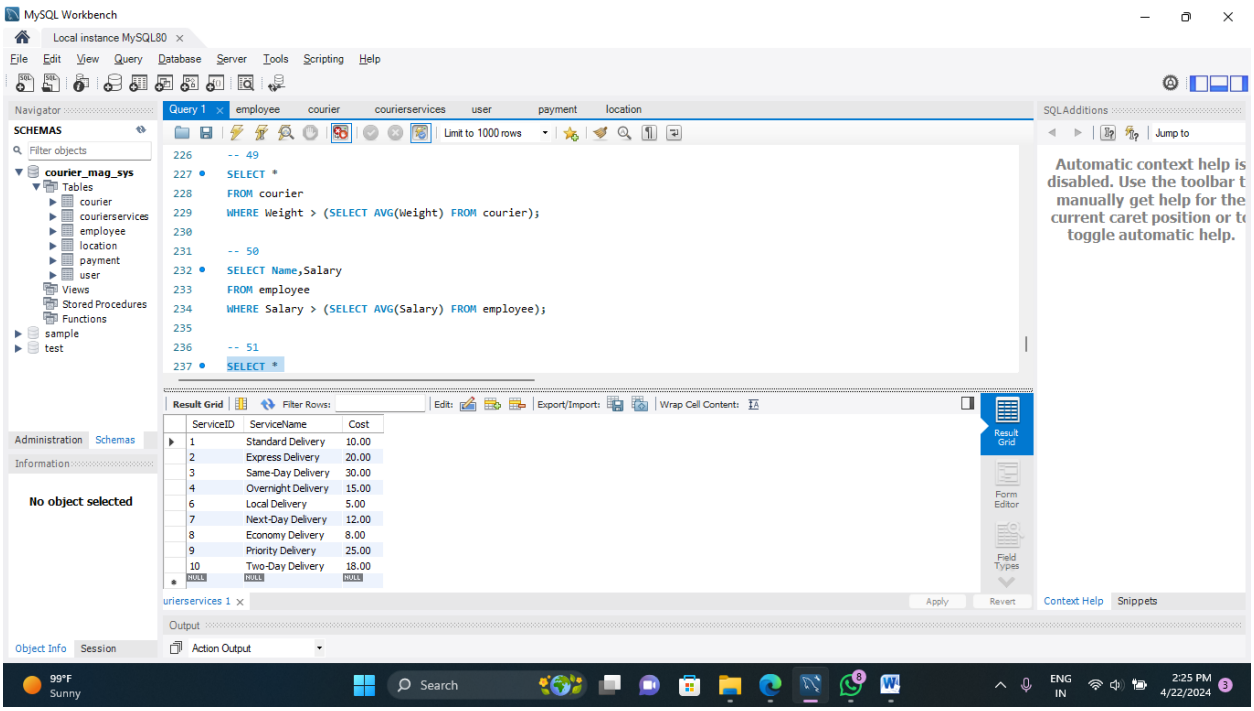
FROM employee

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



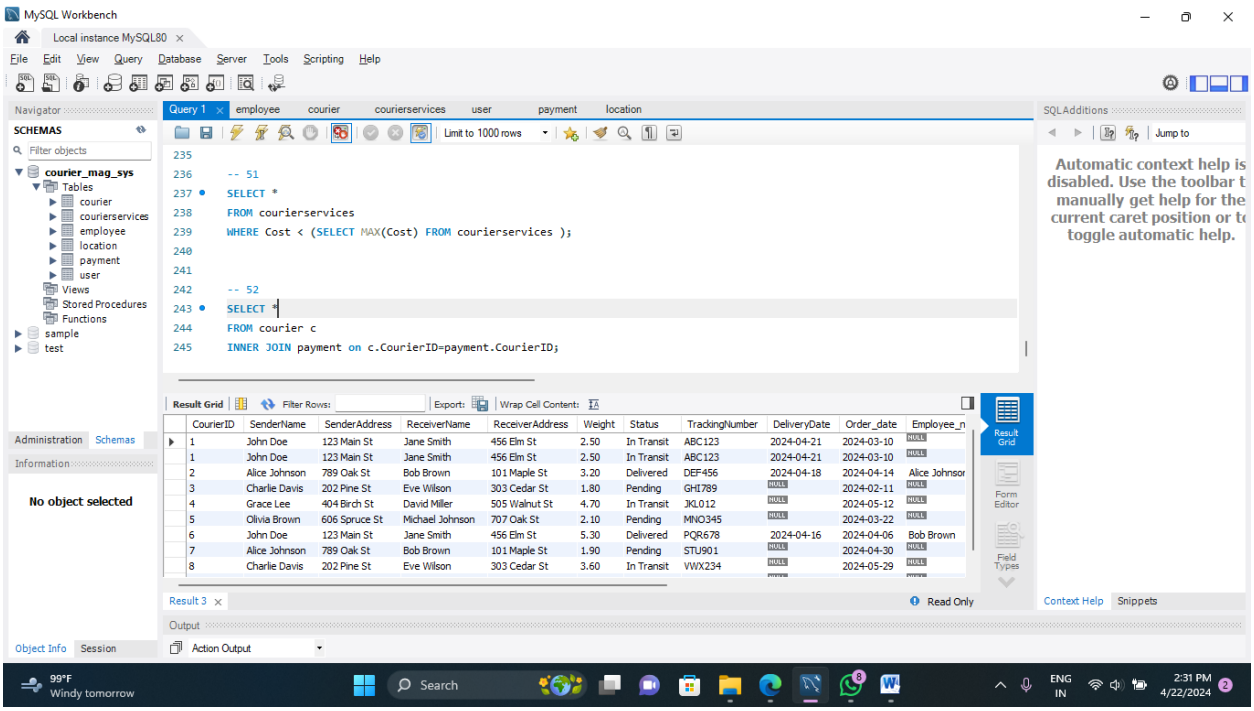
51. Find the total cost of all courier services where the cost is less than the maximum cost

```
SELECT *

FROM courierservices

WHERE Cost < (SELECT MAX(Cost) FROM courierservices );
```



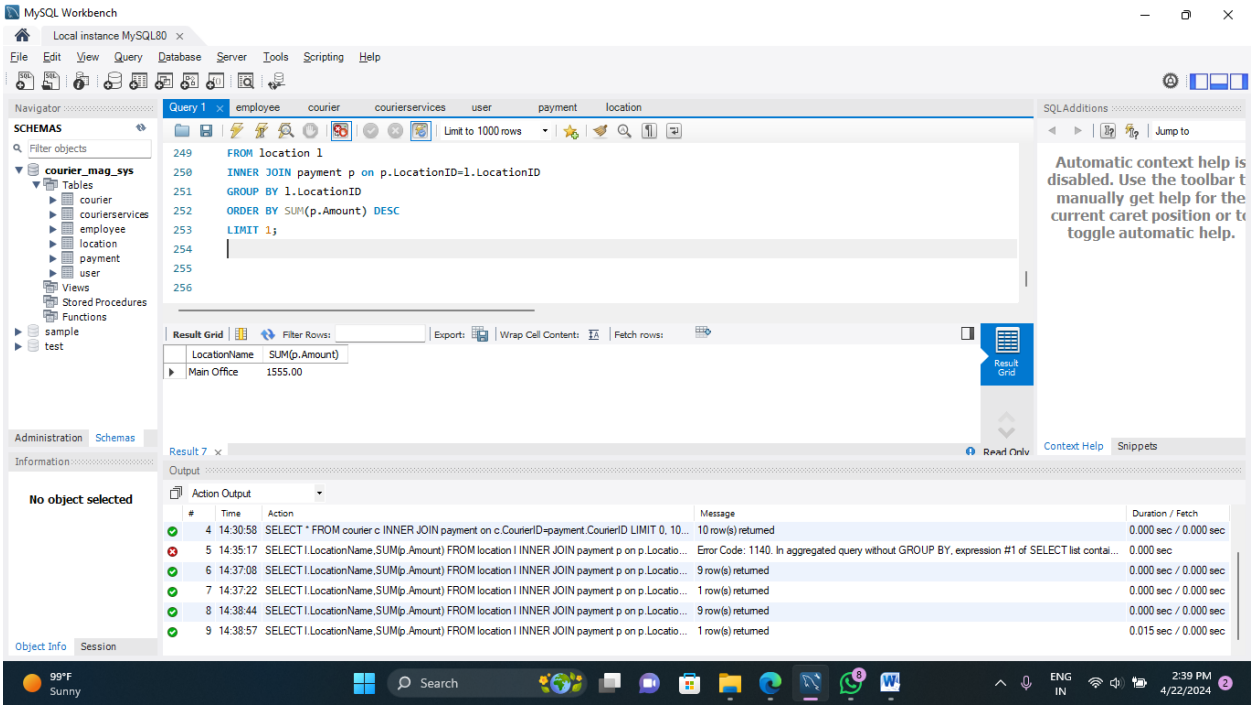
52. Find all couriers that have been paid for

```
SELECT *
FROM courier c
JOIN payment on c.CourierID=payment.CourierID;
```



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

```
SELECT l.LocationName,SUM(p.Amount)
FROM location l
INNER JOIN payment p on p.LocationID=l.LocationID
GROUP BY l.LocationID
ORDER BY SUM(p.Amount) DESC
LIMIT 1;
```



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

```
SELECT
*
FROM courier
WHERE Weight > (SELECT SUM(Weight)
                FROM courier
        WHERE SenderName='Alice Johnson')
```