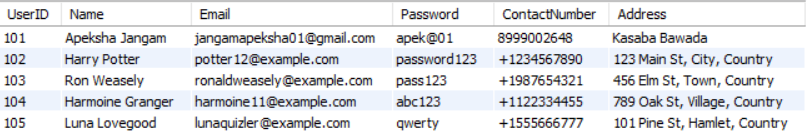
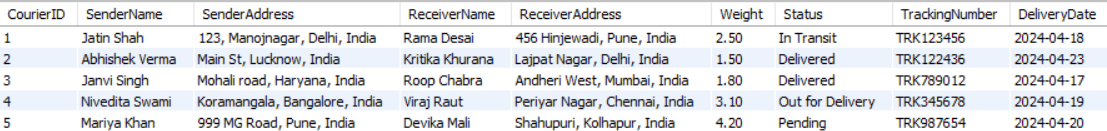
SQL ASSIGNMENT:

TASK 1:

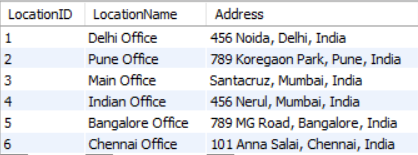
Users Table-



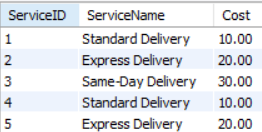
Couriers Table-



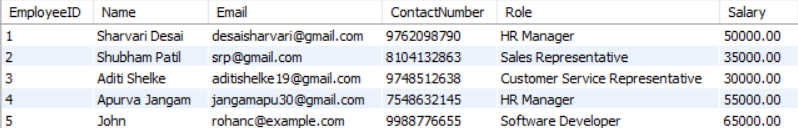
Locations Table-



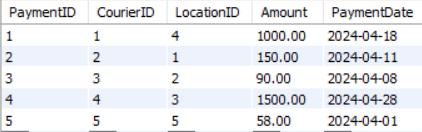
Courier Services Table-



Employees Table-



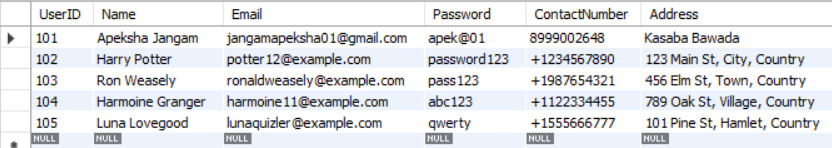
Payments Table-



TASK 2:

1. List all customers:

select \* from users;



2.List all orders for a specific customer:

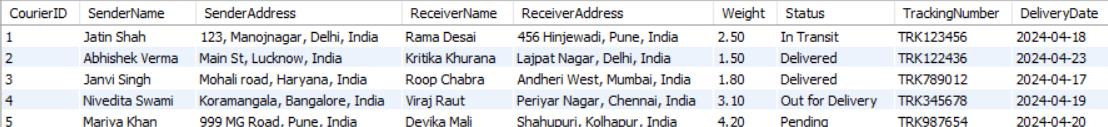
select \* from users

where userid = '104';



3. List all couriers:

select \* from couriers;



4. List all packages for a specific order:

select \* from couriers

where CourierID = 2;



5. List all deliveries for a specific courier:

select \* from couriers

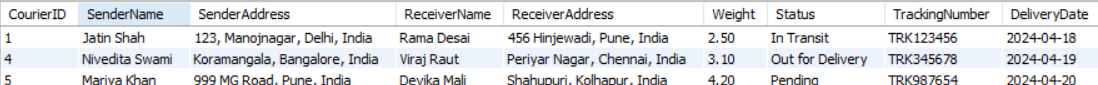
where CourierID = 5;



6. List all undelivered packages:

select \* from Couriers

where not status = 'Delivered';



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

SELECT \*

FROM Couriers

WHERE DATE(DeliveryDate) = CURDATE();



8. List all packages with a specific status:

select \* from Couriers

where status = 'Pending';



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

select count(CourierID) AS TotalPackages

from couriers;



10. Find the average delivery time for each courier

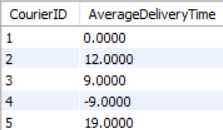
SELECT c.CourierID,

AVG(DATEDIFF(c.DeliveryDate, p.PaymentDate)) AS AverageDeliveryTime

FROM Couriers c

JOIN Payments p ON c.CourierID = p.CourierID

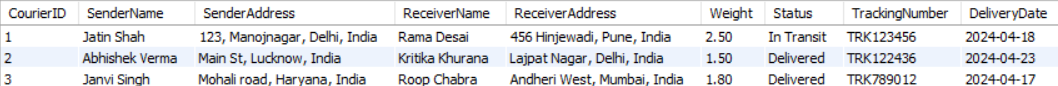
GROUP BY c.CourierID;



11. List all packages with a specific weight range:

select \* from couriers

where weight between 1 and 3;



12. Retrieve employees whose names contain 'John'

select Name from Employees

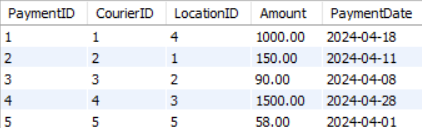
where Name Like 'John%';



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

select \* from Payments

where amount > 50;



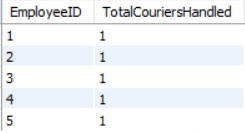
Task 3:

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

SELECT EmployeeID, COUNT(\*) AS TotalCouriersHandled

FROM Couriers

GROUP BY EmployeeID;



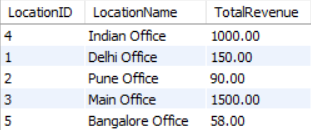
15. Calculate the total revenue generated by each location

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

FROM Locations l

JOIN Payments p ON l.LocationID = p.LocationID

GROUP BY l.LocationID, l.LocationName;



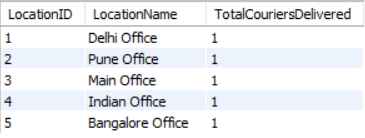
16. Find the total number of couriers delivered to each location.

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

FROM Locations l

JOIN Payments p ON l.LocationID = p.LocationID

GROUP BY l.LocationID, l.LocationName;



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

SELECT c.CourierID, AVG(DATEDIFF(c.DeliveryDate, p.PaymentDate)) AS AvgDeliveryTime

FROM Couriers c

JOIN Payments p ON c.CourierID = p.CourierID

GROUP BY c.CourierID

ORDER BY AvgDeliveryTime DESC

LIMIT 1;



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

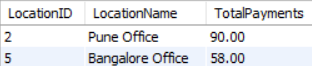
SELECT l.LocationID, l.LocationName, SUM(p.Amount) AS TotalPayments

FROM Locations l

JOIN Payments p ON l.LocationID = p.LocationID

GROUP BY l.LocationID, l.LocationName

HAVING TotalPayments < 100;



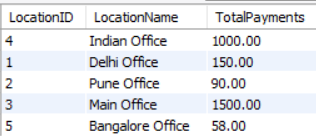
19. Calculate Total Payments per Location

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

FROM Locations l

JOIN Payments p ON l.LocationID = p.LocationID

GROUP BY l.LocationID, l.LocationName;



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

(LocationID = X):

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

FROM Couriers c

JOIN Payments p ON c.CourierID = p.CourierID

WHERE p.LocationID = 3

GROUP BY c.CourierID

HAVING SUM(p.Amount) > 1000;



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

(PaymentDate > 'YYYY-MM-DD'):

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

FROM Couriers c

JOIN Payments p ON c.CourierID = p.CourierID

WHERE p.PaymentDate > '2024-04-18'

GROUP BY c.CourierID

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 TotalPayments

FROM Locations l

JOIN Payments p ON l.LocationID = p.LocationID

WHERE p.PaymentDate > '2024-04-11'

GROUP BY l.LocationID, l.LocationName

HAVING SUM(p.Amount) > 5000;



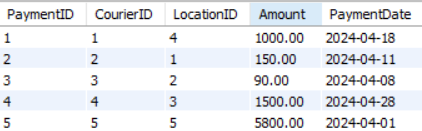
Task 4:

23. Retrieve Payments with Courier Information

SELECT p.\*

FROM Payments p

JOIN Couriers c ON p.CourierID = c.CourierID;

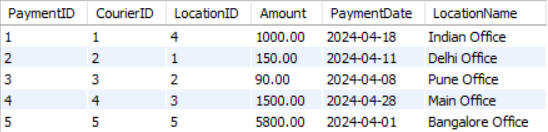


24. Retrieve Payments with Location Information

SELECT p.\*, l.LocationName

FROM Payments p

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



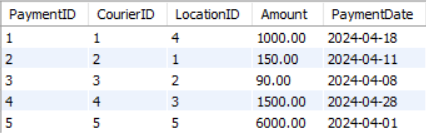
25. Retrieve Payments with Courier and Location Information

SELECT p.\*

FROM Payments p

JOIN Couriers c ON p.CourierID = c.CourierID

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

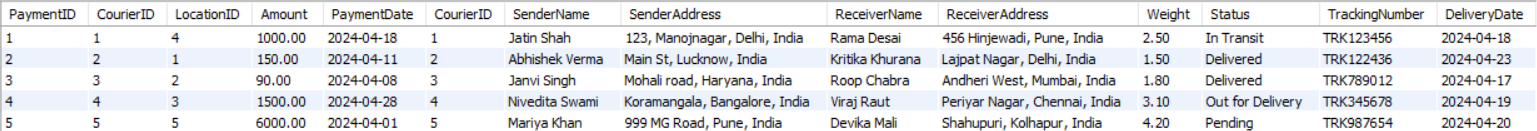


26. List all payments with courier details

SELECT p.\*, c.\*

FROM Payments p

JOIN Couriers c ON p.CourierID = c.CourierID;



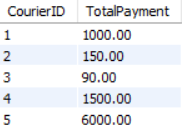
27. Total payments received for each courier

SELECT c.CourierID, SUM(p.Amount) AS TotalPayment

FROM Couriers c

JOIN Payments p ON c.CourierID = p.CourierID

GROUP BY c.CourierID;



28. List payments made on a specific date

SELECT \*

FROM Payments

WHERE PaymentDate = '2024-04-01';



29. Get Courier Information for Each Payment

SELECT p.\*, c.\* // like we could only use paymentid

FROM Payments p

JOIN Couriers c ON p.CourierID = c.CourierID;

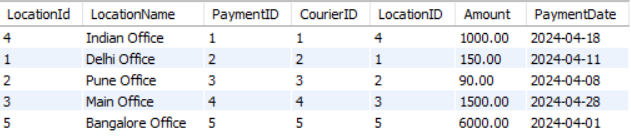


30. Get Payment Details with Location

SELECT l.LocationId, l.LocationName, p.\*

FROM Locations l

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



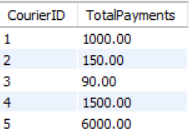
31. Calculating Total Payments for Each Courier

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

FROM Couriers c

JOIN Payments p ON c.CourierID = p.CourierID

GROUP BY c.CourierID;



32. List Payments Within a Date Range

SELECT \*

FROM Payments

WHERE PaymentDate BETWEEN '2024-04-01' AND '2024-04-10';



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

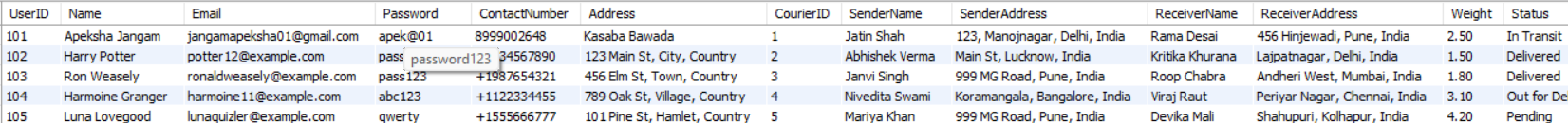
LEFT JOIN Couriers c ON u.UserID = c.UserID

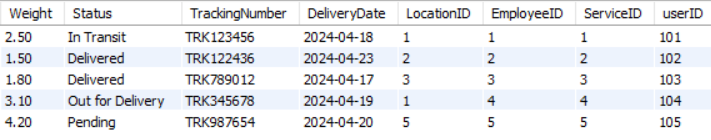
UNION

SELECT u.\*, c.\*

FROM Users u

RIGHT JOIN Couriers c ON u.UserID = c.UserID;





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

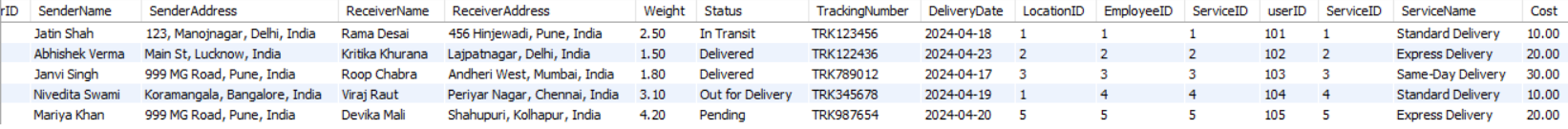
LEFT JOIN Courierservices cs ON c.ServiceID = cs.ServiceID

UNION

SELECT c.\*, cs.\*

FROM Couriers c

RIGHT 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

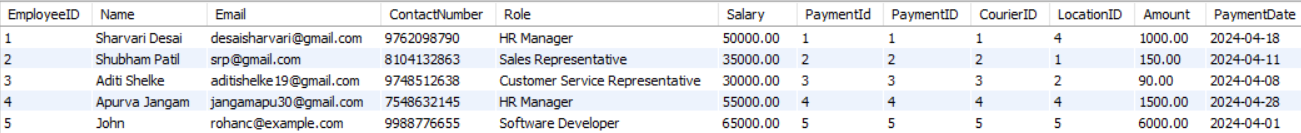
LEFT JOIN Payments p ON e.paymentID = p.paymentID

UNION

SELECT e.\*, p.\*

FROM Employees e

RIGHT JOIN Payments p ON e.paymentID = p.paymentID;

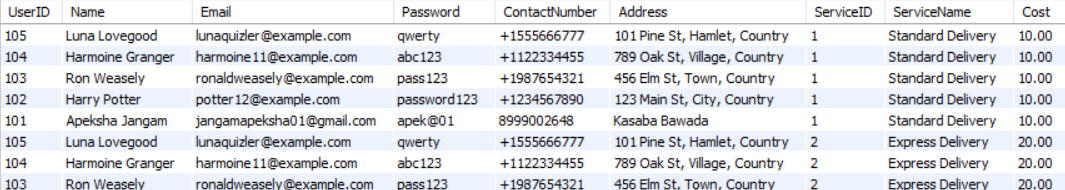


36. List all users and all courier services, showing all possible combinations.

SELECT \*

FROM Users

FULL JOIN Courierservices;

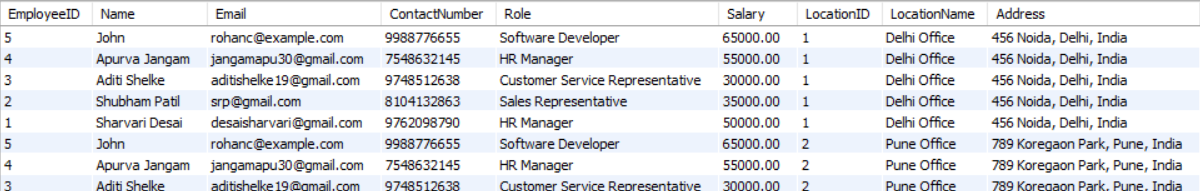


37. List all employees and all locations, showing all possible combinations:

SELECT \*

FROM Employees

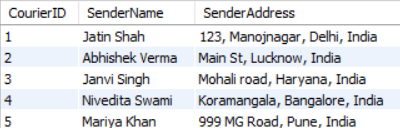
CROSS JOIN Locations;



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

SELECT c.CourierID, c.SenderName, c.SenderAddress

FROM Couriers c;



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

SELECT CourierID, ReceiverName, ReceiverAddress

FROM Couriers;

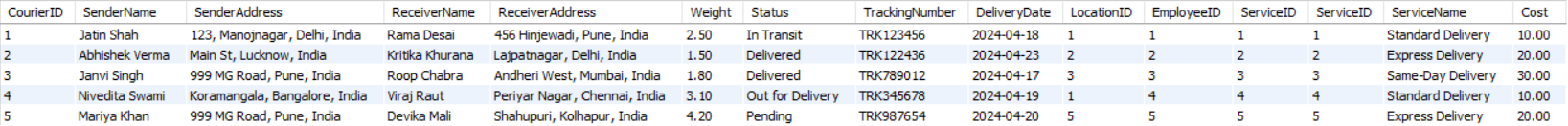


40. Retrieve a list of couriers along with the courier service details (if available):

SELECT c.\*, cs.\*

FROM Couriers c

inner JOIN Courierservices cs ON c.ServiceID = cs.ServiceID;



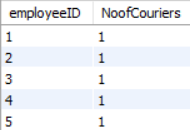
41. Retrieve a list of employees and the number of couriers assigned to each employee:

SELECT e.employeeID, COUNT(DISTINCT c.courierID) AS NoofCouriers

FROM Employees e

RIGHT JOIN Couriers c ON e.employeeID = c.employeeID

GROUP BY e.employeeID;



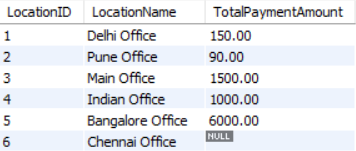
42. Retrieve a list of locations and the total payment amount received at each location:

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

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 c.\* //self join

FROM Couriers c

JOIN Couriers c2 ON c.SenderName = c2.SenderName AND c.CourierID <> c2.CourierID;



44. List all employees who share the same role.

SELECT e.\*

FROM Employees e

JOIN Employees e2 ON e.Role = e2.Role AND e.EmployeeID <> e2.EmployeeID;

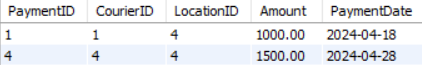


45. Retrieve all payments made for couriers sent from the same location.

Select p.\*

From Payments p

Join Payments p1 ON p.LocationID = p1.LocationID AND p.courierId <> p1.courierID;

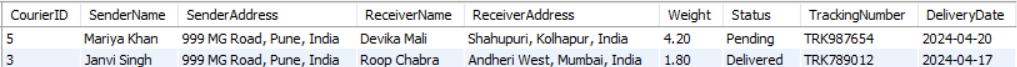


46. Retrieve all couriers sent from the same location (based on SenderAddress).

SELECT c.\*

FROM Couriers c

JOIN Couriers c2 ON c.SenderAddress = c2.SenderAddress AND c.CourierID <> c2.CourierID;



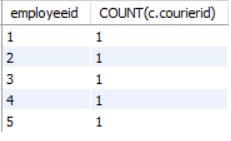
47. List employees and the number of couriers they have delivered:

select e.employeeid, COUNT(c.courierid)

from couriers c

join employees e on c.employeeid = e.employeeid

group by e.employeeid;



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

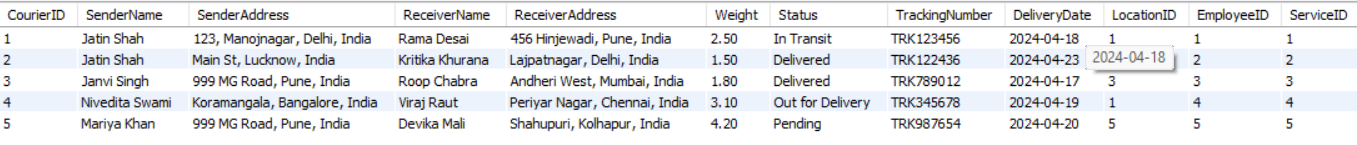
select c.\*

from couriers c

join payments p on c.courierID = p.courierID

join courierservices cs on c.serviceId = cs.serviceId

where p.amount > cs.cost

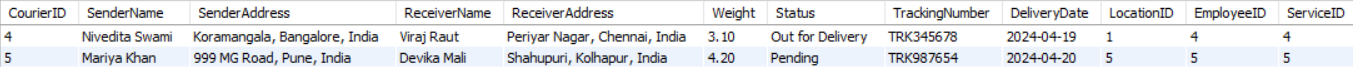


49. Find couriers that have a weight greater than the average weight of all couriers

select \*

from couriers

where weight > (select avg(weight) from couriers);

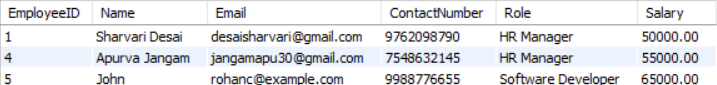


50. Find the names of all employees who have a salary greater than the average salary:

select \*

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(Cost)

FROM CourierServices

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



52. Find all couriers that have been paid for

SELECT DISTINCT c.\*

FROM Couriers c

INNER JOIN Payments p ON c.CourierID = p.CourierID;



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

SELECT LocationID

FROM Payments

GROUP BY LocationID

HAVING SUM(Amount) = (

SELECT MAX(TotalAmount)

FROM (

SELECT SUM(Amount) AS TotalAmount

FROM Payments

GROUP BY LocationID

) AS subquery

);



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

SELECT \*

FROM Couriers

WHERE Weight > ALL (SELECT Weight FROM Couriers WHERE SenderName = 'Janvi singh');

