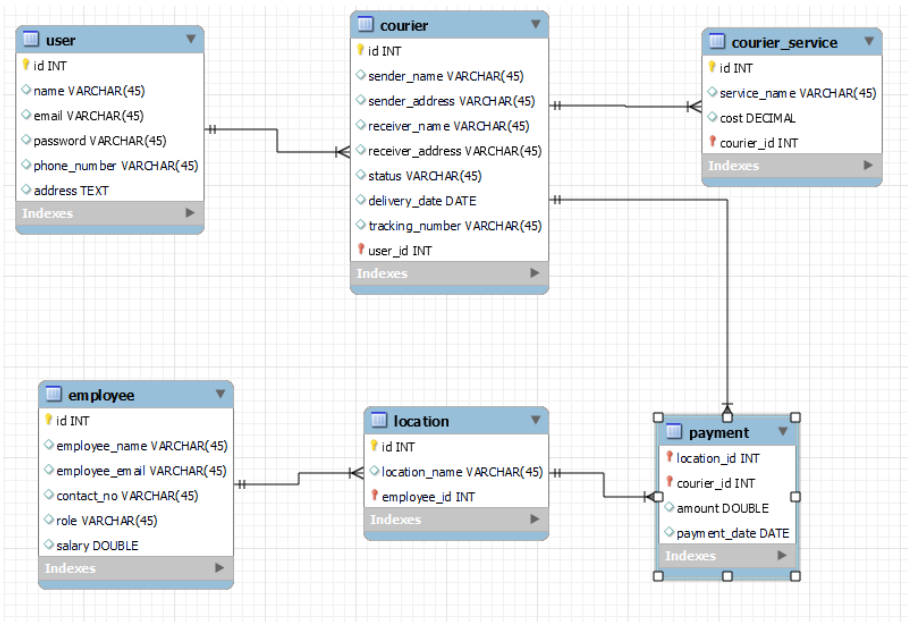
COURIER MANAGEMENT SYSTEM ASSIGNMENT

ER DIAGRAM:



use courier_service;

insert into user(id,name,email,password,phone_number,address)values

(1,'Sheela','sheelaprakash@gamil.com','sheela','45677888999','Bangalore'),

(2,'arohi','arohiharsha@gamil.com','arohi','98765552323','Hyderabad'),

(3,'Anu','Anu65332h@gamil.com','Anu','77803974645','Chennai'),

(4,'Sanjay','sanjayrama@gamil.com','Sanjay','256369283','Chennai');

insert into user(id,name,email,password,phone_number,address)values

(5,'Prakash','prakash5y78982@gamil.com','Prakash','801266532','Bangalore'),

(6,'Harsha','harsha35tg@gamil.com','Harsha','9876675232','Hyderabad');

----

insert into courier(sender_name, sender_address, receiver_name, receiver_address, status,delivery_date,tracking_number,user_id) values

('sheela','Bangalore','anjalai','tirupur','delivered','2024-01-20',6789,1),

('aryan','tirupur','Sanjay','Chennai','Sender','2024-04-01',6782,4),

('Harsha','Hyderabad','Aadhi','hyderabad','delivered','2023-09-06',6770,6),

('anu','Chennai','madhu','Ooty','On-the-way','2024-02-22',6790,3);

```
insert into courier_service(service_name,cost,courier_id) values
('speed','1000',1),
('mhaps','750',4),
('hjkt','150',3);
```

```
insert into employee(employee_name,employee_email,contact_no,role,salary) values
('Divi','divi56gt@gmail.com','9867654234','deliver','20000'),
('sara','sara56gt@gmail.com','9876612345','packaging','17000'),
('Selvakumar','selvakumar45356gt@gmail.com','9983245671','manager','50000'),
('Raju','raju56gt@gmail.com','9844354234','deliver','19000'),
('Rudhran','rudharan456@gmail.com','8965543109','manager','70000');
```

```
insert into location(location_name,employee_id) values
('Kodaikanal',3),
('Chennai',1),
('Coimbatore',2),
('Bangalore',5),
('Chennai',4);
```

```
insert into payment(location_id,courier_id,amount,payment_date) values
(1,4,'1000','2024-01-02'),
(2,1,'350','2023-10-20'),
(3,3,'750','2024-02-10'),
(4,2,'150','2024-03-10');
```

Task 2: Select,Where

Solve the following queries in the Schema that you have created above

1. List all customers:

```
select * from user;
```

2. List all orders for a specific customer:

```
select * from courier where user_id = 3;
```

3. List all couriers:

```
select * from courier;
```

4. List all packages for a specific order:

```
select * from courier_service where courier_id = 3;
```

5. List all deliveries for a specific courier:

```
select * from courier_service where courier_name='speed';
```

6. List all undelivered packages:

```
select * from courier where status = 'undelivered';
```

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

```
select *
from courier
where delivery_date = 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 c.courier_name, count(s.id) as total_packages
from courier_service s
join courier c on s.courier_id = c.id
group by c.courier_name;
```

10. Find the average delivery time for each courier

```
select c.courier_name, avg(s.delivery_duration) as average_delivery_time
from courier_service s
join courier c on s.courier_id = c.id
group by c.courier_name;
```

11. List all packages with a specific weight range:

```
select *

from packages

where weight >= min_weight

and weight <= max_weight;
```

12. Retrieve employees whose names contain 'John'

```
select * from employee where employee_name like '%john%';
```

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

```
select * from payment where amount > 50;
```

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

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

```
select e.id, e.employee_name, count(c.id) as total_couriers_handled

from employee e

join courier c on e.id = c.employee_id

group by e.id, e.employee_name;
```

15. Calculate the total revenue generated by each location

```
select l.id, l.location_name, sum(p.amount) as total_revenue

from payment p

join location l on p.location_id=l.id

group by l.id, l.location_name;
```

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

```
select l.id, l.location_name, count(c.id) as total_couriers_delivered

from courier c

join payment p on c.id = p.courier_id

join location l on l.id = p.location_id

group by l.id, l.location_name;
```

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

```
select c.id,  avg(c.delivery_date) as average_delivery_time
from courier c
group by c.id
order by average_delivery_time desc
limit 1;
```

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

```
select l.id, l.location_name, sum(p.amount) as total_payment
from payment p
join location l on p.location_id = l.id
group by l.id, l.location_name
having total_payment < p.amount;
```

19. Calculate Total Payments per Location

```
select l.id, l.location_name, sum(p.amount) as total_payments
from payment p
join location l on p.location_id = l.id
group by l.id, l.location_name;
```

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

```
select c.id,  sum(p.amount) as total_payments
from payment p
join courier c on p.courier_id = c.id
where p.location_id = 2
group by c.id,
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.id,  sum(p.amount) as total_payments
from payment p
join courier c on p.courier_id = c.id
```

```
where p.payment_date > ' 2024-02-10'

group by c.id, c.sender_name

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 p.location_id, l.location_name, sum(p.amount) as total_payments

from payment p

join location l on p.location_id = l.id

where p.payment_date > '2021-03-10'

group by p.location_id, l.location_name

having sum(p.amount) > 5000;
```

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

23. Retrieve Payments with Courier Information

```
select p.amount, p.payment_date, c.courier_name, c.tracking_number

from payment p

join courier c on p.courier_id = c.id;
```

24. Retrieve Payments with Location Information

```
select  p.amount, p.payment_date, l.location_name

from payment p

join location l on p.location_id = l.id;
```

25. Retrieve Payments with Courier and Location Information

```
select p.amount, p.payment_date, c.courier_name, c.tracking_number, l.location_name

from payment p

join courier c on p.courier_id = c.id

join location l on p.location_id = l.id;
```

26. List all payments with courier details

```
select  p.amount, p.payment_date, c.courier_name, c.tracking_number

from payment p
```

join courier c on p.courier_id = c.id;

27. Total payments received for each courier

select c.courier_name, sum(p.amount) as total_payments_received

from courier c

join payment p on c.id = p.courier_id

group by c.courier_name;

28. List payments made on a specific date

select *

from payment

where payment_date = '2024-03-10';

29. Get Courier Information for Each Payment

select p.amount, p.payment_date, c.courier_name, c.tracking_number

from payment p

left join courier c on p.courier_id = c.payment_id;

30. Get Payment Details with Location

select p.amount, p.payment_date, l.location_name

from payment p

left join location l on p.location_id = l.id;

31. Calculating Total Payments for Each Courier

select c.courier_name, sum(p.amount) as total_payments

from courier c

join payment p on c.id = p.courier_id

group by c.courier_name;

32. List Payments Within a Date Range

select *

from payment

where payment_date between '2024-01-01' and '2024-01-31';

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

select *

from user u

join courier c on u.id = c.user_id;

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

select *

from courier c

JOIN courier_service s ON c.id = s.courier_id;

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

select *

from employee e

join location l on e.id = l.employee_id

join payment p on l.id = p.location_id;

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

select u.id, u.user_name, c.id, c.courier_name

from user u

join courier c;

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

select e.id, e.employee_name, l.id, l.location_name

from employee e

join location l;

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

select s.service_name as couriername, c.sender_name as sendername, c.sender_address as senderaddress

from courier c

join courier_service s on c.id = s.courier_id;

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

select s.service_name as couriername, c.receiver_name as receivername, c.receiver_address as receiveraddress

from courier c

join courier_service s on c.id = s.courier_id;

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

```
select c.id as courierid, s.service_name as servicename
from courier c
JOIN Courier_Service s ON c.id = s.courier_id;
```

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

```
select e.employee_name as employeename, count(c.id) as numberofcouriers
from employee e
JOIN Courier c ON c.employee_id = c.id
GROUP BY e.id, e.employee_name;
```

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

```
select l.location_name as locationname, sum(p.amount) as totalpaymentamount
from location l
join payment p on l.id = p.location_id
group by l.id, l.location_name;
```

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

```
SELECT c.*
FROM courier c
--group by sender_name;
```

44. List all employees who share the same role.

```
select employee_name,role
from employee
where role = (select role from employee where id = 'employee id');
```

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

```
select p.*
from payment p
join location l on p.location_id=l.id
where l.location_name = 'chennai';
```

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

```
select c.*
from courier c
```

join courier c2 on c1.senderaddress = c2.senderaddress

where c2.courierid = 'specific courier id';

-- NOTE: From q-43 to q-46 I was unable get an relevant solution.I couldn't be able to think a method to solve the questions.

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

SELECT e.employee_name AS EmployeeName, COUNT(c.id) AS NumberOfCouriersDelivered

FROM employee e

LEFT JOIN courier c ON e.courier_id = c.id

GROUP BY e.id, e.employee_name;

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

SELECT c.*

FROM courier c

JOIN payment p ON c.id = p.courier_id

JOIN courier_service s ON c.id = s.courier_id

WHERE p.amount > s.cost;

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

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

SELECT *

FROM 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 employee_name

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 SUM(cost) AS TotalCost
FROM courier_service
WHERE cost < (
  SELECT MAX(cost)
  FROM courier_service
);
```

52. Find all couriers that have been paid for

```
SELECT * FROM courier
WHERE id IN (SELECT DISTINCT courier_id FROM payment);
```

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

```
SELECT location_id
FROM payment
GROUP BY location_id
HAVING SUM(amount) = (
  SELECT MAX(total_amount)
  FROM (
    SELECT SUM(amount) AS total_amount
    FROM payment
    GROUP BY location_id
  ) AS max_amount_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 courier
WHERE weight > (
  SELECT MAX(weight)
  FROM courier
  WHERE sender_name = 'sheela' );
```