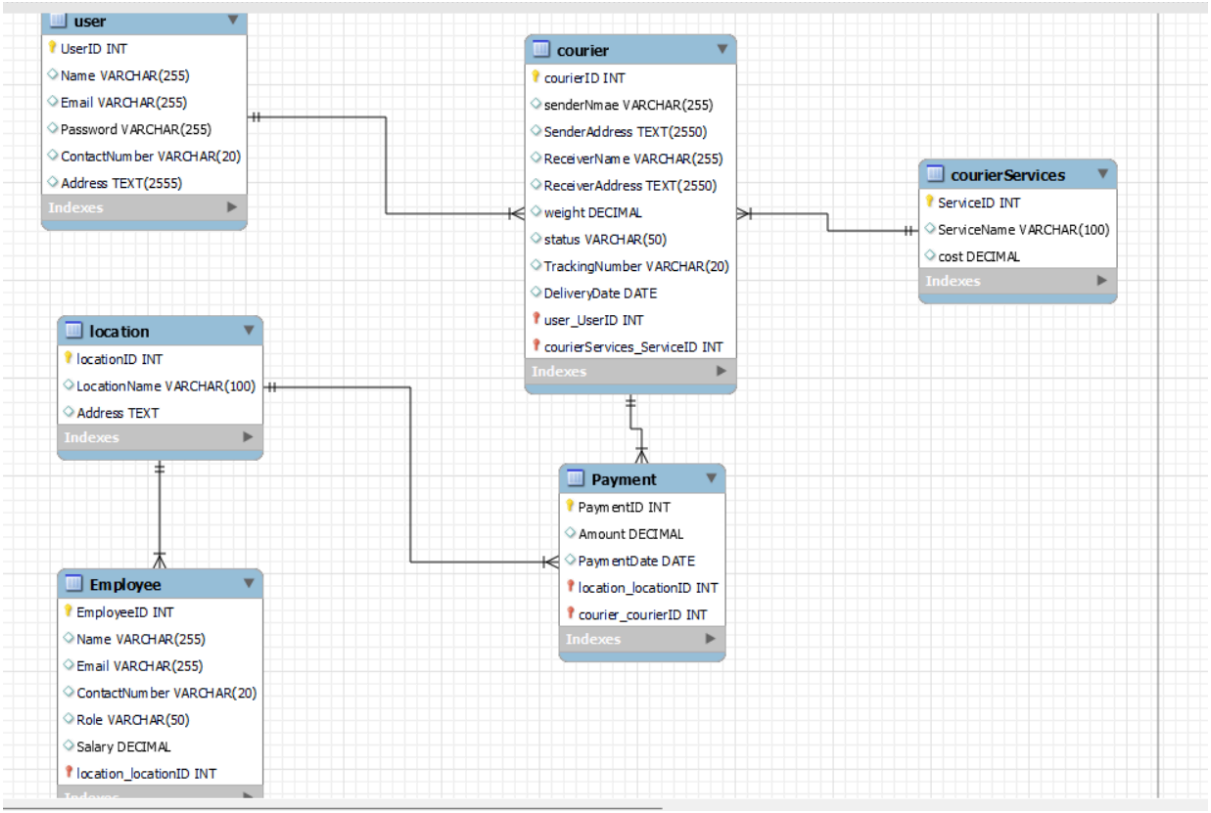
Courier Management system:

ER DIAGRAM :



#courier db scripts

-- MySQL Workbench Forward Engineering

-- ---
-- Schema Couriermanagement_db1
-- ---

-- ---
-- Schema Couriermanagement_db1
-- ---
CREATE SCHEMA IF NOT EXISTS `Couriermanagement_db1` DEFAULT CHARACTER SET utf8 ;
USE `Couriermanagement_db1` ;

-- ---
-- Table `Couriermanagement_db1`.`user`

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Couriermanagement_db1`.`user` (
  `UserID` INT NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(255) NULL,
  `Email` VARCHAR(255) NULL,
  `Password` VARCHAR(255) NULL,
  `ContactNumber` VARCHAR(20) NULL,
  `Address` TEXT(2555) NULL,
  PRIMARY KEY (`UserID`),
  UNIQUE INDEX `Email_UNIQUE` (`Email` ASC) )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `Couriermanagement_db1`.`courierServices`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Couriermanagement_db1`.`courierServices` (
  `ServiceID` INT NOT NULL AUTO_INCREMENT,
  `ServiceName` VARCHAR(100) NULL,
  `cost` DECIMAL NULL,
  PRIMARY KEY (`ServiceID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `Couriermanagement_db1`.`courier`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Couriermanagement_db1`.`courier` (
  `courierID` INT NOT NULL AUTO_INCREMENT,
  `senderNmae` VARCHAR(255) NULL,
  `SenderAddress` TEXT(2550) NULL,
  `ReceiverName` VARCHAR(255) NULL,
  `ReceiverAddress` TEXT(2550) NULL,
  `weight` DECIMAL NULL,
  `status` VARCHAR(50) NULL,
  `TrackingNumber` VARCHAR(20) NULL,
  `DeliveryDate` DATE NULL,
  `user_UserID` INT NOT NULL,
  `courierServices_ServiceID` INT NOT NULL,
  PRIMARY KEY (`courierID`, `user_UserID`, `courierServices_ServiceID`),
  UNIQUE INDEX `TrackingNumber_UNIQUE` (`TrackingNumber` ASC) ,
  INDEX `fk_courier_user1_idx` (`user_UserID` ASC) ,
  INDEX `fk_courier_courierServices1_idx` (`courierServices_ServiceID` ASC) ,
  CONSTRAINT `fk_courier_user1`
    FOREIGN KEY (`user_UserID`)
    REFERENCES `Couriermanagement_db1`.`user` (`UserID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
```

```
  CONSTRAINT `fk_courier_courierServices1`
    FOREIGN KEY (`courierServices_ServiceID`)
    REFERENCES `Couriermanagement_db1`.`courierServices` (`ServiceID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `Couriermanagement_db1`.`location`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Couriermanagement_db1`.`location` (
  `locationID` INT NOT NULL AUTO_INCREMENT,
  `LocationName` VARCHAR(100) NULL,
  `Address` TEXT NULL,
  PRIMARY KEY (`locationID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `Couriermanagement_db1`.`Payment`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Couriermanagement_db1`.`Payment` (
  `PaymentID` INT NOT NULL AUTO_INCREMENT,
  `Amount` DECIMAL NULL,
  `PaymentDate` DATE NULL,
  `location_locationID` INT NOT NULL,
  `courier_courierID` INT NOT NULL,
  PRIMARY KEY (`PaymentID`, `location_locationID`, `courier_courierID`),
  INDEX `fk_Payment_location1_idx` (`location_locationID` ASC) ,
  INDEX `fk_Payment_courier1_idx` (`courier_courierID` ASC) ,
  CONSTRAINT `fk_Payment_location1`
    FOREIGN KEY (`location_locationID`)
    REFERENCES `Couriermanagement_db1`.`location` (`locationID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Payment_courier1`
    FOREIGN KEY (`courier_courierID`)
    REFERENCES `Couriermanagement_db1`.`courier` (`courierID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `Couriermanagement_db1`.`Employee`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Couriermanagement_db1`.`Employee` (
```

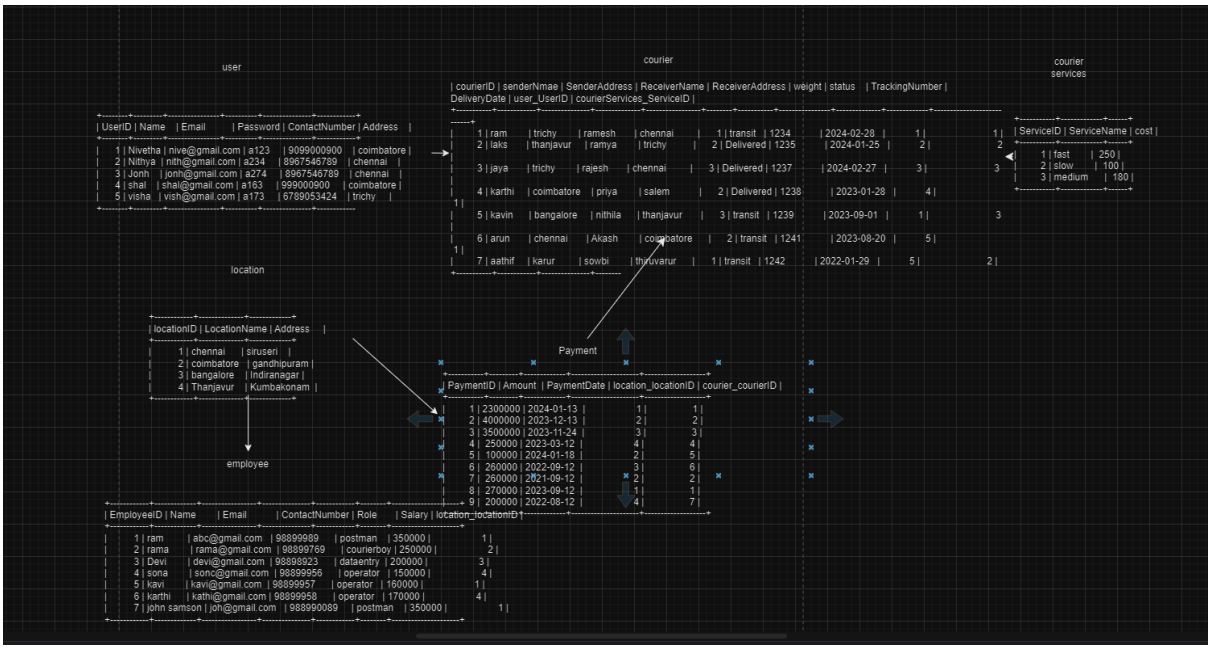
```
  `EmployeeID` INT NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(255) NULL,
  `Email` VARCHAR(255) NULL,
  `ContactNumber` VARCHAR(20) NULL,
  `Role` VARCHAR(50) NULL,
  `Salary` DECIMAL NULL,
  `location_locationID` INT NOT NULL,
  PRIMARY KEY (`EmployeeID`, `location_locationID`),
  UNIQUE INDEX `Email_UNIQUE` (`Email` ASC) ,
  INDEX `fk_Employee_location1_idx` (`location_locationID` ASC) ,
  CONSTRAINT `fk_Employee_location1`
    FOREIGN KEY (`location_locationID`)
    REFERENCES `Couriermanagement_db1`.`location` (`locationID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

#insertions and query

```
use Couriermanagement_db1;
#insertions
insert into user(Name,Email,Password,ContactNumber,Address)
values
('Nivetha', 'nive@gmail.com', 'a123' ,'9099000900','coimbatore'),
('Nithya','nith@gmail.com' , 'a234' , '8967546789' , 'chennai'),
('Jonh','jonh@gmail.com' , 'a274' , '8967546789' , 'chennai'),
('shal', 'shal@gmail.com', 'a163' ,'999000900','coimbatore'),
('visha' , 'vish@gmail.com','a173','6789053424','trichy');

insert into courierServices( ServiceName,cost)
values
('fast','250'),
('slow','100'),
('medium','180') ;
insert into courier(senderNmae,  senderAddress, ReceiverName , Receiveraddress ,weight ,
status , trackingnumber , DeliveryDate,user_userID ,courierservices_serviceID)
values
('ram','trichy','ramesh','chennai','1','transit', '1234', '2024-02-28', 1,1),
('laks', 'thanjavur','ramya','trichy','2','Delivered' ,'1235','2024-01-25',2,2),
('jaya','trichy','rajesh','chennai','3','Delivered', '1237', '2024-02-27',3,3),
('karthi','coimbatore','priya','salem','2','Delivered', '1238', '2023-01-28',4,1),
('kavin','bangalore','nithila','thanjavur','3','transit', '1239', '2023-09-01',1,3),
('arun','chennai','Akash','coimbatore','2','transit', '1241', '2023-08-20',5,1),
('aathif','karur','sowbi','thiruvarur','1','transit', '1242', '2022-01-29',5,2);
insert into location ( LocationName , address)
values
```

```
('chennai','siruseri'),
('coimbatore','gandhipuram'),
('bangalore','Indiranagar'),
('Thanjavur','Kumbakonam');

insert into payment(Amount,PaymentDate,location_locationID,courier_courierID) values
(2300000,'2024-01-13',1,1),
(4000000, '2023-12-13',2,2),
(3500000,'2023-11-24',3,3),
('250000','2023-03-12',4,4);
insert into payment(Amount,PaymentDate,location_locationID,courier_courierID) values
('100000','2024-01-18',2,5),
('260000','2022-09-12',3,6),
('260000','2021-09-12',2,2),
('270000','2023-09-12',1,1),
('200000','2022-08-12',4,7);
insert into Employee(name, Email, ContactNumber,Role,Salary,location_locationID)
values
('ram','abc@gmail.com','98899989', 'postman','350000',1),
('rama','rama@gmail.com','98899769', 'courierboy','250000',2),
('Devi','devi@gmail.com','98898923', 'dataentry','200000',3),
('sona','sonc@gmail.com','98899956', 'operator','150000',4),
('kavi','kavi@gmail.com','98899957', 'operator','160000',1),
('karthi','kathi@gmail.com','98899958', 'operator','170000',4);
insert into Employee(name, Email, ContactNumber,Role,Salary,location_locationID)
values ('john samson','joh@gmail.com','988990089', 'postman','350000',1);


select * from courierServices;
select * from employee;
select * from payment;
select * from location;
```

Reference Image



TASK 2:

-- 1. List all customers:
select* from user;
-- 2. List all orders for a specific customer:
select * from courier
where senderNmae='ram';
-- 3. List all couriers:
select * from courier;
-- 4. List all packages for a specific order:
select * from courier
where courierID=4;
-- 5. List all deliveries for a specific courier:
select * from courier
where status='Delivered';
-- 6.. List all undelivered packages:alter
select * from courier
where status!='Delivered';
-- 7. List all packages that are scheduled for delivery today:
select * from courier
where DeliveryDate=CURDATE();
-- 8. List all packages with a specific status:
select * from courier where status = 'transit';
-- 9. Calculate the total number of packages for each courier.
select courierID ,count(courierID) as total_courier from courier
group by courierID;
-- 10. Find the average delivery time for each courier
select c.courierID , avg(abs(c.DeliveryDate-p.paymentDate)) as average_time

```
from courier c , payment p
where c.courierID=p.courier_courierID
group by c.courierID;
-- 11. List all packages with a specific weight range:
select * from courier
where weight between 1 and 2;
-- 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 c.senderNmae,  c.senderAddress, c.ReceiverName , c.Receiveraddress ,c.weight ,
c.status , c.trackingnumber , c.DeliveryDate
from courier c , payment p
where c.courierID= p.courier_courierID AND amount >50;
```

TASK 3:

```
-- Task 3: GroupBy, Aggregate Functions, Having, Order By, where
-- 14. Find the total number of couriers handled by each employee.
select e.name , e.employeeId ,count(c.courierID)
from courier c , employee e, payment p , location l
where c.courierID=p.courier_courierID AND
l.locationID=p.location_locationID AND
l.locationID=e.location_locationID
group by e.name;
-- 15. Calculate the total revenue generated by each location
select LocationName ,sum(Amount) as total_revenue
from Location l, payment p
where l.locationID=p.location_locationID
group by LocationName;
-- 16. Find the total number of couriers delivered to each location.
select l.LocationName ,count(c.courierID) as No_of_couriers
from location l , courier c , payment p
where c.courierID =p.courier_courierID AND
l.locationID =p.location_locationID
group by l.LocationName;
-- 17. Find the courier with the highest average delivery time:
select c.courierID , avg(abs(c.DeliveryDate-p.paymentDate) )as average_time
from courier c , payment p
where c.courierID=p.courier_courierID
group by c.courierID
order by courierID asc
limit 0, 1;
-- 18. Find Locations with Total Payments Less Than a Certain Amount
select l.locationName , sum(p.amount) as total_payments
from location l , payment p
where l.locationID = p.location_locationID
```

```
group by LocationName
HAving total_payments<1000000;
```
-- 19. Calculate Total Payments per Location
```
select l.locationName , sum(p.amount) as total_payments
from location l , payment p
where l.locationID = p.location_locationID
group by LocationName;
```
-- 20. Retrieve couriers who have received payments totaling more than $1000 in a specific location (LocationID = X):
```
select c.courierID, c.sendernmae, sum(p.amount) as total_pay
from  courier c , location l , payment p
where l.locationID = p.location_locationID AND LocationID =1
AND  c.courierID = p.courier_courierID
group by 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, c.sendernmae, sum(p.amount) as total_pay
from  courier c , location l , payment p
where l.locationID = p.location_locationID AND p.paymentdate > '2023-01-01'
AND  c.courierID = p.courier_courierID
group by 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 c.courierID, c.sendernmae, sum(p.amount) as total_pay
from  courier c , location l , payment p
where l.locationID = p.location_locationID AND p.paymentdate > '2023-01-01'
AND  c.courierID = p.courier_courierID
group by courierID
HAVING sum(p.amount)>5000;
```

TASK 4:

-- Task 4: Inner Join,Full Outer Join, Cross Join, Left Outer Join,Right Outer Join
-- 23. Retrieve Payments with Courier Information
```
 select * from
 payment p left join courier c on p.courier_courierID = c.courierId;
```

-- 24. Retrieve Payments with Location Information
```
select * from
payment p join location l on p.location_locationID= l.locationID;
```

-- 25. Retrieve Payments with Courier and Location Information
```
select *
from payment p join  courier c on p.courier_courierID = c.courierId join  location l on
p.location_locationID= l.locationID;
```

-- 26. List all payments with courier details

```
select *
from payment p left join courier c on  p.courier_courierID = c.courierId;
```

-- 27. Total payments received for each courier

```
select c.courierID, sum(p.amount) as Total_Payment
from payment p left join courier c on  p.courier_courierID = c.courierId
group by c.courierid;
```

-- 28. List payments made on a specific date

```
select * from payment
where paymentdate ='2023-03-12';
```

-- 29. Get Courier Information for Each Payment

```
select p.paymentid , c.courierID , c.senderAddress , c.Receivername , c.weight , c.status ,
c.trackingnumber , c.deliverydate
from courier c join payment p on  p.courier_courierID = c.courierId
group by paymentid;
```

-- 30. Get Payment Details with Location

```
select p.paymentid, p.amount , p.paymentdate ,l.locationname from
payment p left join location l on p.location_locationID= l.locationID;
```

-- 31. Calculating Total Payments for Each Courier

```
select c.courierID, sum(p.amount) as Total_Payment
from payment p left join courier c on  p.courier_courierID = c.courierId
group by c.courierid;
```

-- 32. List Payments Within a Date Range

```
select paymentid , amount ,paymentdate from payment
where paymentdate between '2023-03-12' AND '2024-01-03';
```

-- 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 left join courier c on u.userid = c.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 c left join courierservices cs on cs.serviceid = c.courierServices_serviceid;
```

-- 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  left join payment p on e.employeeID =p.paymentid;
```

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

```
select *
from user , courier;
```

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

```
select *
from employee , location;
```

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

```
select courierID , sendernmae, senderAddress
from courier;
```
-- 39. Retrieve a list of couriers and their corresponding receiver information (if available):
```
select courierID , Receivername ,receiverAddress
from courier;
```
-- 40. Retrieve a list of couriers along with the courier service details (if available):
```
select c.courierID ,cs.serviceID, cs.servicename, cs.cost
from courier c left join courierservices cs on cs.serviceID=c.courierservices_serviceID;
```
-- 41. Retrieve a list of employees and the number of couriers assigned to each employee:
```
select e.employeeID , e.name , e.email ,e.contactNUmber, e.salary
,  c.courierID , c.senderAddress , c.Receivername , c.weight , c.status , c.trackingnumber ,
c.deliverydate
from employee e left join location l on l.locationid = e. location_locationID
join payment p on l.locationID = p.location_locationID join courier c on c.courierid =
p.courier_courierID;
```
-- 42. Retrieve a list of locations and the total payment amount received at each location:
```
select l.locationID , l.locationname , sum(p.amount) as total_payment
from location l join payment p on l.locationid = p.location_locationid
group by l.locationid;
```
-- 43. Retrieve all couriers sent by the same sender (based on SenderName).
```
select courierID ,sendernmae , senderaddress , receivername , receiveraddress ,weight ,
status , trackingnumber
from courier
where sendernmae ='arun';
```
-- 44. List all employees who share the same role.-- subquery
```
select employeeid , name ,role , email , contactnumber , salary
from employee where role  in(
select role
from employee
group by role
having count(employeeid) > 1);
```

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

```
select p.paymentID , p.amount,p.paymentdate , l.locationname
from payment p join location l on p.location_locationId = l.locationID where locationid IN (
select l.locationid
from payment p join location l on p.location_locationId = l.locationID
group by l.locationID
having count(l.locationID)>1);
```
-- 46. Retrieve all couriers sent from the same location (based on SenderAddress).
```
select courierID , senderAddress , Receivername ,Receiveraddress, weight , status ,
trackingnumber , deliverydate
from courier where senderaddress IN(
select senderaddress
```

```
from courier
group by senderaddress
having count(senderaddress)>1);
```
-- 47. List employees and the number of couriers they have delivered:
```
select e.employeeID , e.name , e.email ,count(c.status) as Number_of_couriers_Delivered
from employee e left join location l on l.locationid = e. location_locationID
join payment p on l.locationID = p.location_locationID join courier c on c.courierid =
p.courier_courierID
group by c.status ;
```
-- 48. Find couriers that were paid an amount greater than the cost of their respective courier services
```
select c.courierID , c.senderAddress , c.Receivername ,c.Receiveraddress, c.weight ,
c.status , c.trackingnumber , c.deliverydate , p.amount as payment , cs.cost as service_cost
from courierservices cs join courier c  on cs.serviceid = c.courierservices_serviceid
join  payment p ON p.courier_courierid = c.courierID
 where p.amount >cs.cost;
```

TASK 5:

-- 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 courierid , weight
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  serviceID ,cost
from courierservices where cost < (select max(cost) from courierServices);
```
-- 52. Find all couriers that have been paid for
```
select courierID , sendernmae from courier
where courierID in ( select courier_courierID from payment) ;
```
-- 53. Find the locations where the maximum payment amount was made
```
select locationid
from location where locationid in ( select location_locationID from payment where amount in
(select max(amount) from payment));
```
-- 54. Find all couriers whose weight is greater than the weight of all couriers sent by a specific sender
-- (e.g., 'SenderName'):
```
select courierid , sendernmae from
courier where weight > ( select sum(weight) from courier where sendernmae = 'ram');
```