**PROJECT DOCUMENTATION**

**NAME: SHRUSHTI SAMANT**

**TOPIC: COURIER DATABASE MANAGEMENT SYSTEM.**

**INTRODUCTION**

My project is a courier management database system that utilizes SQL and ER modeling to manage customer information, delivery details, package information, dispatch data, logistics operations, and invoice generation. The system allows you to store and manage relevant data and generate reports to analyze data related to shipments, inventory, billing, and payments. The system is designed to be user-friendly, mobile-friendly, and scalable, with strong security measures to protect data from unauthorized access.

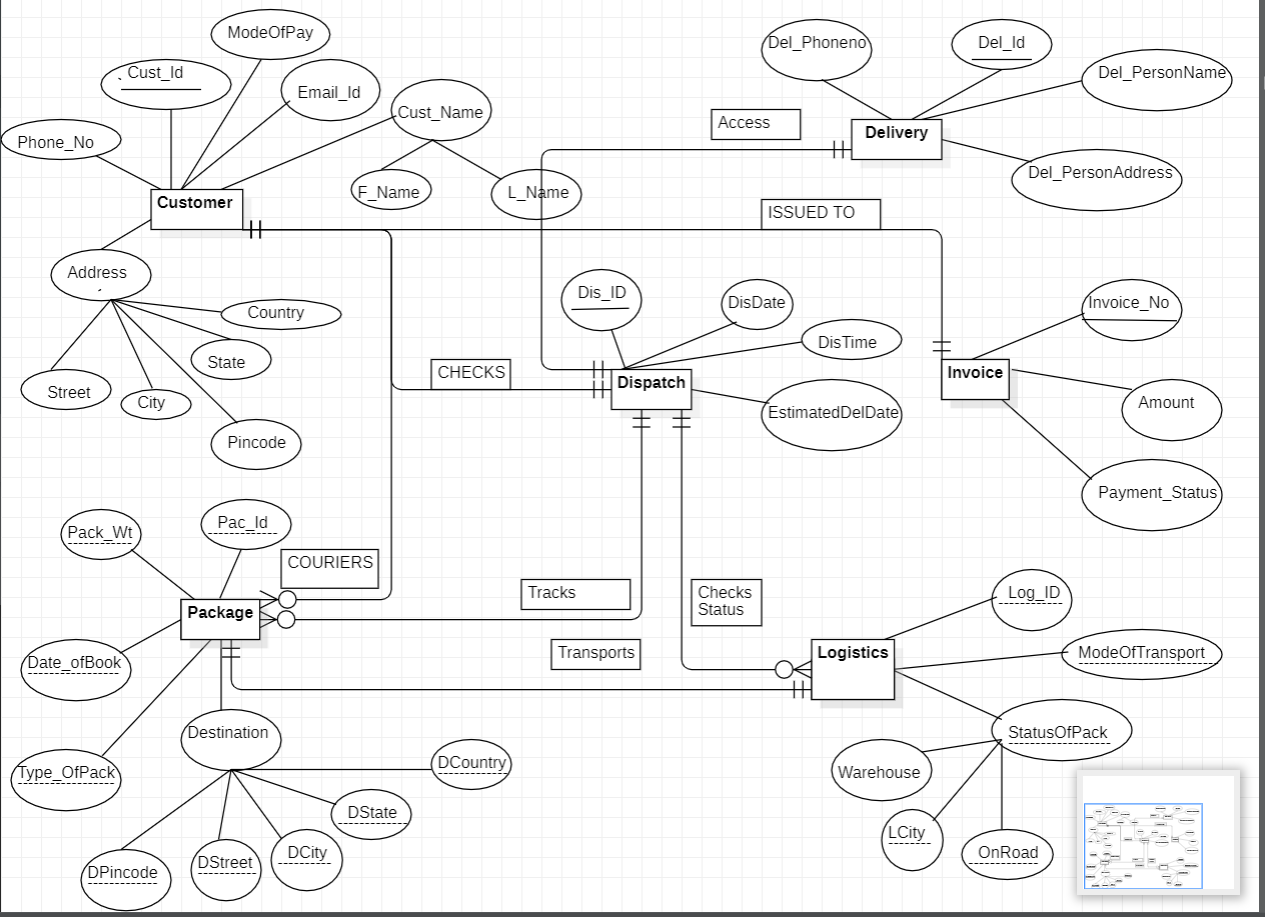
**OBJECTIVES**

* Database design and normalization
* SQL programming language
* Handling and manipulating data in a database
* Data analysis and querying
* Data visualization techniques
* Understanding customer behavior and needs
* Building efficient and effective data management systems
* Collaboration and project management skills.

**APPLICATION SPECIFICATIONS**

1. Customer management
2. Order tracking
3. Order management
4. Payment and invoice management
5. Delivery and Logistics management

**Entity-Relationship Model**



Following are the entities and their attributes

1. Customer -This entity represents the people or companies who are sending or receiving packages through your courier service. It may include fields such as name, address, phone number, and email.
2. Delivery - This entity represents the specific delivery of a package from one location to another. It may include fields such as the date/time of the delivery, the sender and recipient's information, and the package ID.
3. Dispatch - This entity represents the dispatch of a package from one location to another. It may include fields such as the date/time of dispatch, the location of dispatch, and the package ID.
4. Package - This entity represents the package being sent or received through your courier service. It may include fields such as the package ID, weight, dimensions, and contents.
5. Logistics - This entity represents the logistics or operations involved in managing your courier service. It may include fields such as employee information, vehicle information, and delivery routes.
6. Invoice - This entity represents the billing and payment information for each package delivery. It may include fields such as the package ID, customer information, delivery information, and payment information.

**ER to RELATIONAL MODEL**

1. Customer( **Cust\_id** , F\_name , L\_name , Phone no. , emailid ,street ,city ,state, country,mode of payment)
2. Dispatch(**Disp\_id** , dis\_date,dis\_time, estimateddeldate,*Cust\_id1*)
3. Package(Pack\_id,*Cust\_id1*, *Disp\_id1*, pack wt, DOB,Pac type , st, ct,street1,country1)
4. Logistics(Log\_id, modeoftransport,warehouse,ct, onroad*,Disp\_id* 1)
5. Delivery(**Del\_id**,delpersonname, delpersonadd, delpersonphoneno,*Disp\_id1*)
6. Invoice(**Invoice no**, Amount ,Payment status ,*Cust\_id1*)

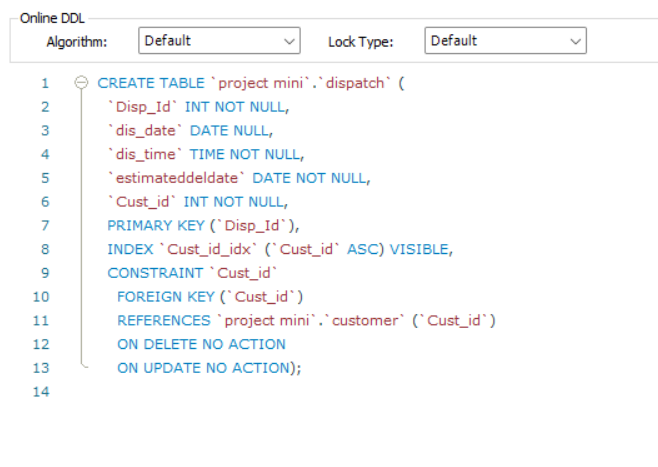
**NORMALIZATION**

DDL commands before normalization

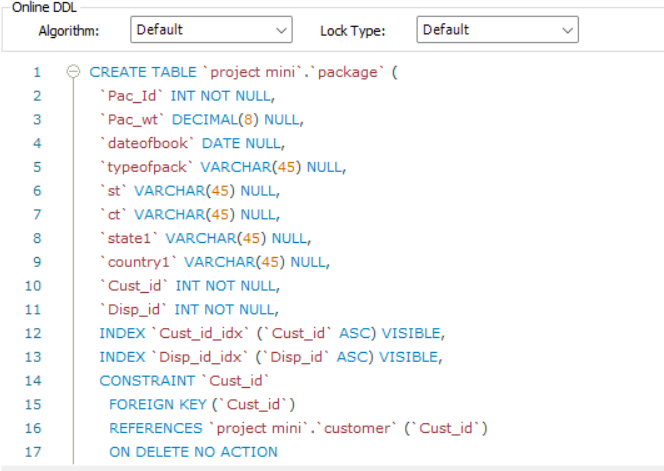
Customer

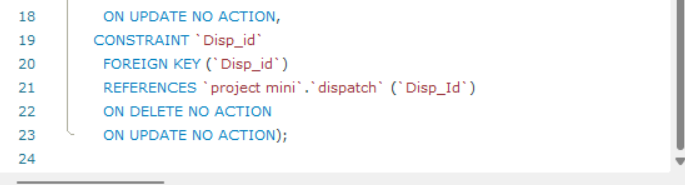


Dispatch



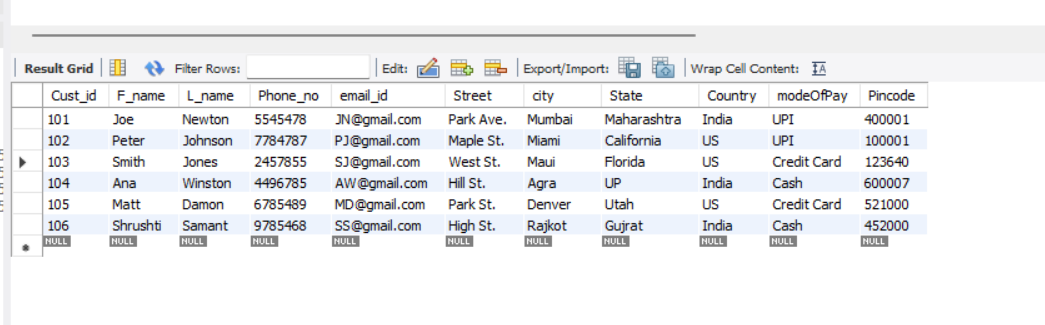
Package



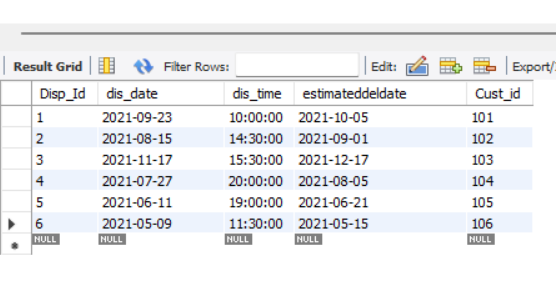


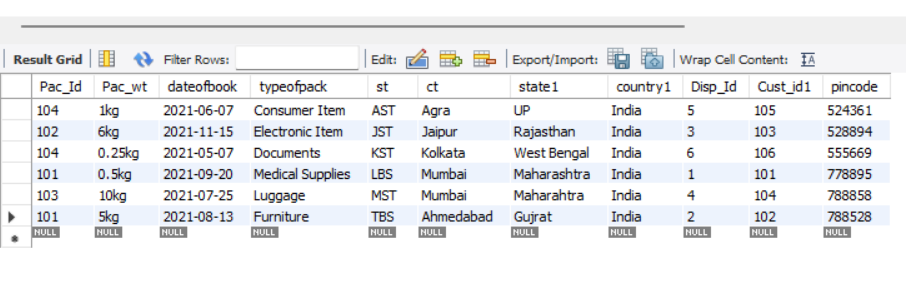
**Tables before normalization**

Customer

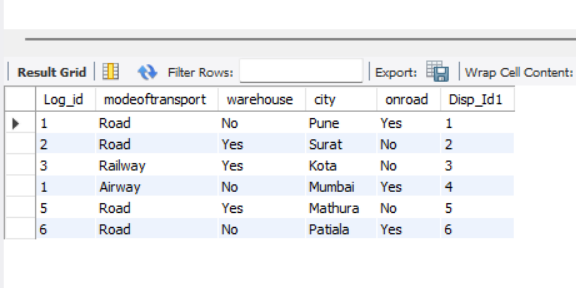


Dispatch

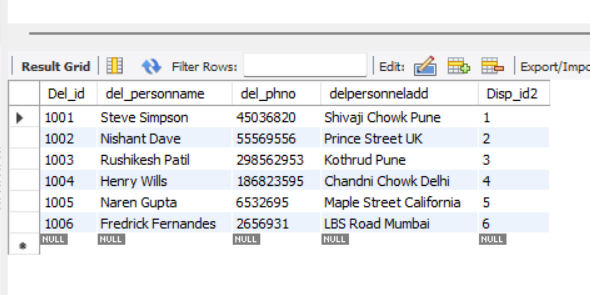


Package

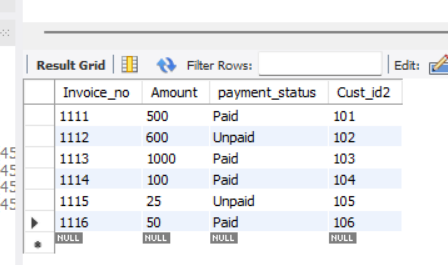
Logistics



Delivery



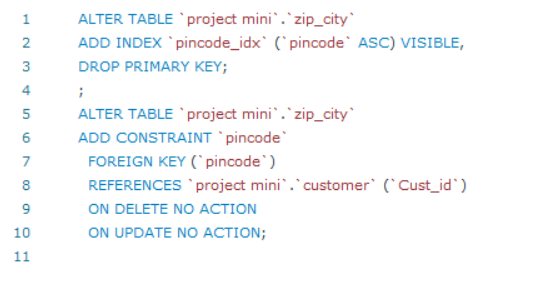
Invoice

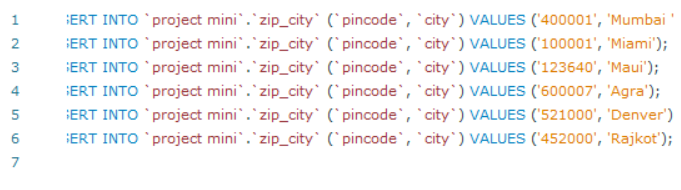


After Normalization

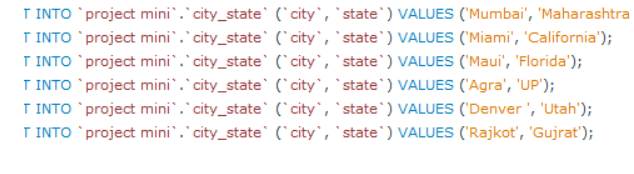
1. Customer

1.1 Pincode\_city

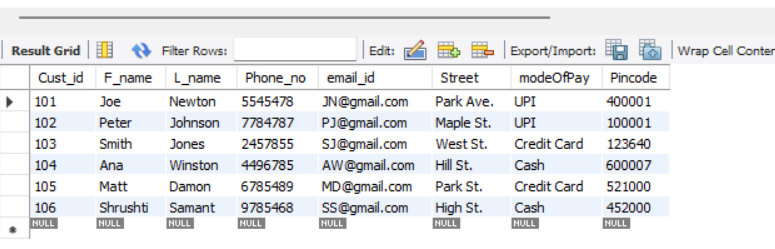




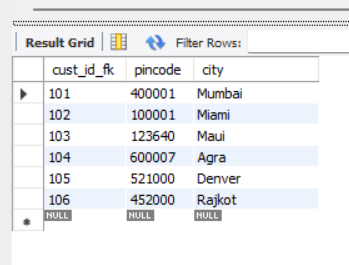
1.2 City State



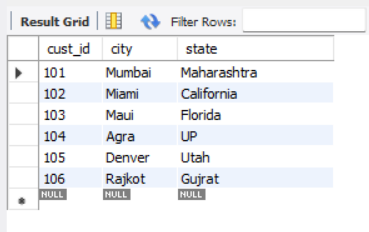
Customer



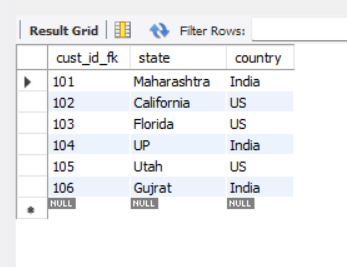
Pincode\_city



City\_state



State\_country



Normalization of Customer Entity

1 NF

The table is already in first normal form (1NF) as it follows the basic rules of a relational database. Each row represents a unique record, and each column contains atomic values.

2NF

The table is already in 2NF. The primary key is Cust\_id, and all the other attributes appear to be fully dependent on it. This means that there are no partial dependencies in the table, and it meets the requirements of 2NF.

3 NF

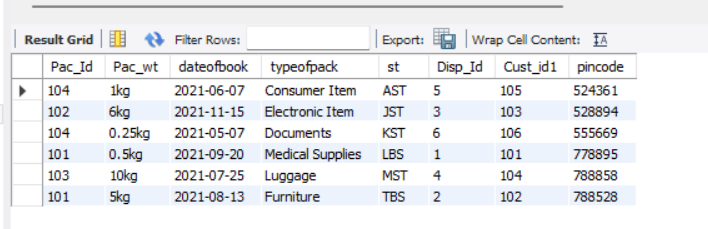
The first table, "Customer", contains the attributes that are directly related to the customer, including the primary key Cust\_id, first name, last name, phone number, email address, street address, mode of payment, and pincode.

The second table, "ZipCity", contains the attributes pincode and city, which were previously in the "Customer" table but had a transitive dependency. By creating a separate table for pincode and city, we eliminate this dependency.

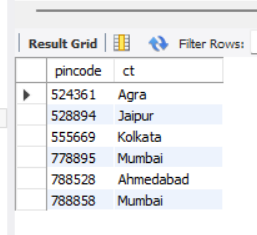
The third table, "CityState", contains the attributes city and state, which also had a transitive dependency in the original "Customer" table.

Finally, the fourth table, "StateCountry", contains the attributes state and country, which were previously in the "Customer" table but have been moved to a separate table to eliminate the transitive dependency.

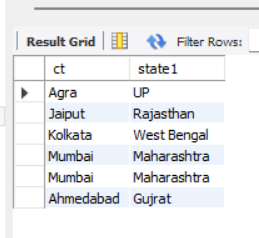
1. Package



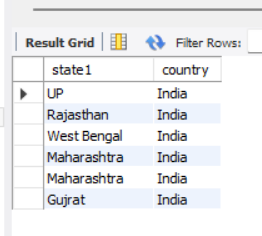
Pin\_city



Ct\_state



St\_country



Normalization of Package Entity

1NF , 2NF

The table is already in 1NF and 2NF because it satisfies the following conditions:

1. Atomic values: Each column of the table contains atomic values, which means that each value in a column is indivisible.
2. Unique column names: Each column in the table has a unique name, and there are no repeating groups.
3. Primary key: The table has a primary key (Pack\_id) that uniquely identifies each row in the table.
4. No partial dependencies: In 2NF, there should not be any partial dependencies where a non-key column is dependent on only a part of the primary key. This table has a single-column primary key, so it cannot have partial dependencies.

3NF

The first table, Package, includes the primary key (Pac\_id) along with all attributes that depend solely on the Pac\_id.

The second table, PinCity, contains the attribute Pincode along with its dependent attribute City.

The third table, CityState, includes the attribute City along with its dependent attribute State.

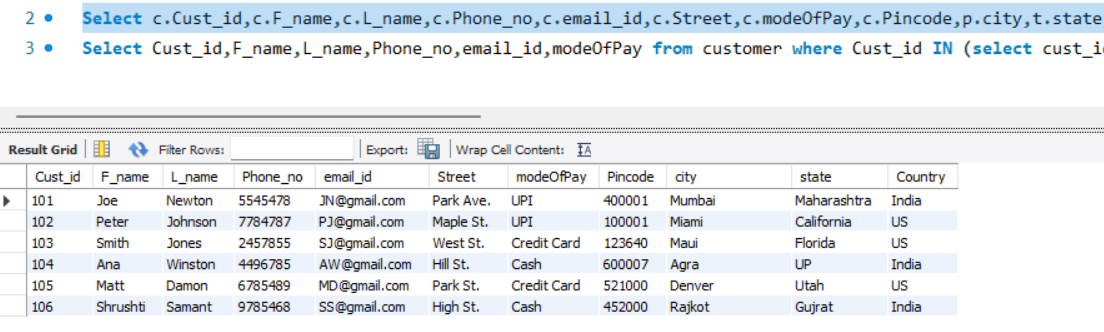
Finally, the fourth table, StateCountry, includes the attribute State along with its dependent attribute Country.

By splitting the table into these four tables, we have eliminated the issue of attributes being dependent on non-key attributes, and the table is now in third normal form (3NF).

Queries:

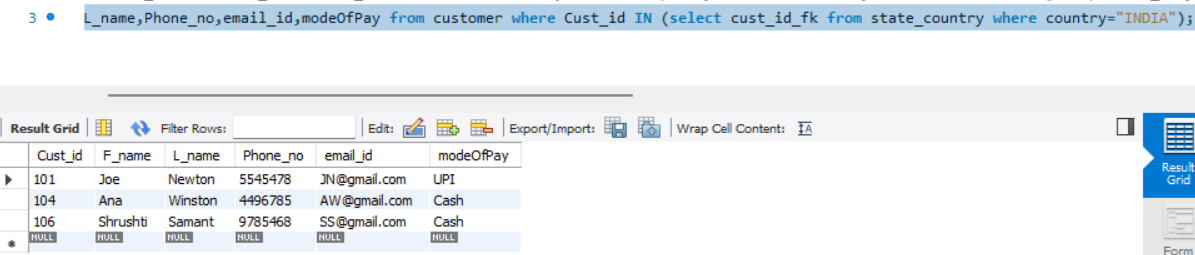
1. Write a query for extracting all details of the customer.

Select c.Cust\_id,c.F\_name,c.L\_name,c.Phone\_no,c.email\_id,c.Street,c.modeOfPay,c.Pincode,p.city,t.state,t.Country from customer as c join pincode\_city as p on (c.Cust\_id= p.cust\_id\_fk) join city\_state as s on(p.cust\_id\_fk = s.cust\_id) join state\_country as t on (s.cust\_id=t.cust\_id\_fk);



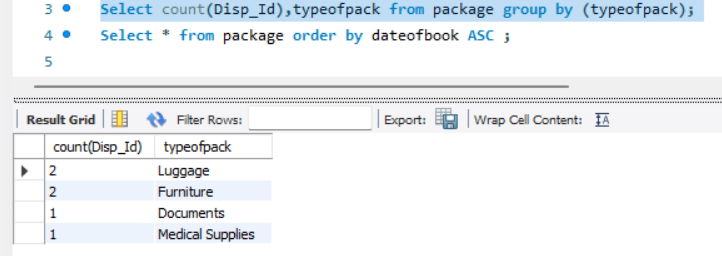
1. Write a query to extract customers from India.

Select Cust\_id,F\_name,L\_name,Phone\_no,email\_id,modeOfPay from customer where Cust\_id IN (select cust\_id\_fk from state\_country where country="INDIA");



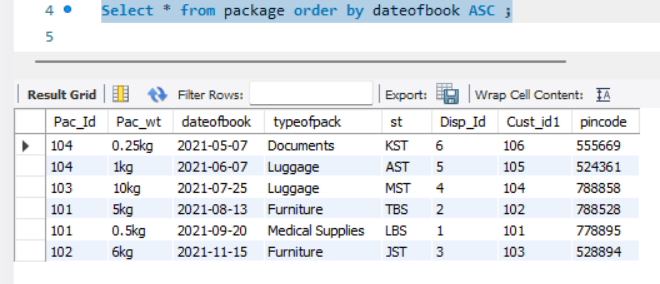
1. Write a query to count and identify the type of packages.

Select count(Disp\_Id),typeofpack from package group by (typeofpack);



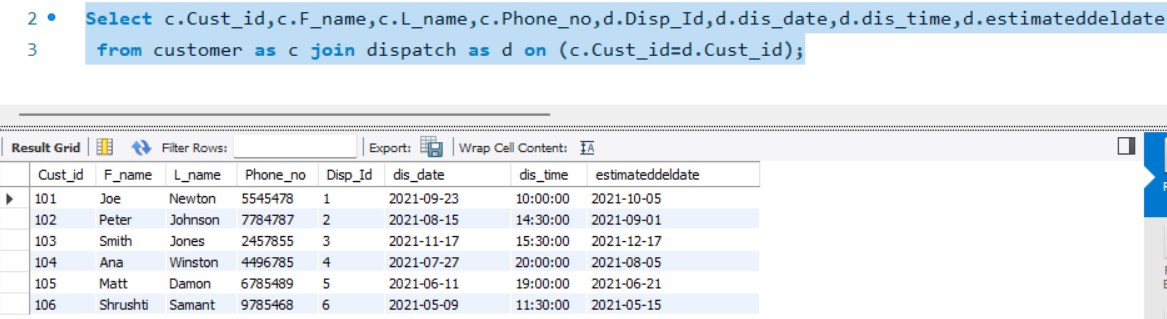
1. Write a query to sort the packages on the basis of Date of booking.

Select \* from package order by dateofbook ASC ;



1. Write a query to fetch the dispatch details of the customer's package.

Select c.Cust\_id,c.F\_name,c.L\_name,c.Phone\_no,d.Disp\_Id,d.dis\_date,d.dis\_time,d.estimateddeldate from customer as c join dispatch as d on (c.Cust\_id=d.Cust\_id);

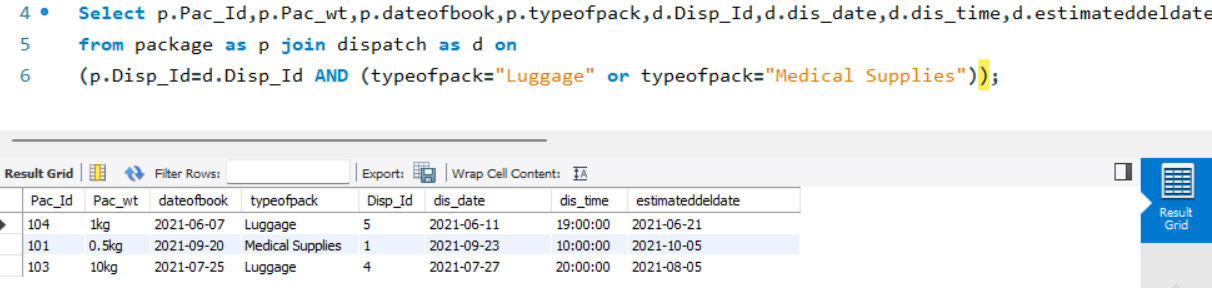


1. Write a query to fetch the dispatch details of Luggage and Medical Supplies packages.

Select p.Pac\_Id,p.Pac\_wt,p.dateofbook,p.typeofpack,d.Disp\_Id,d.dis\_date,d.dis\_time,

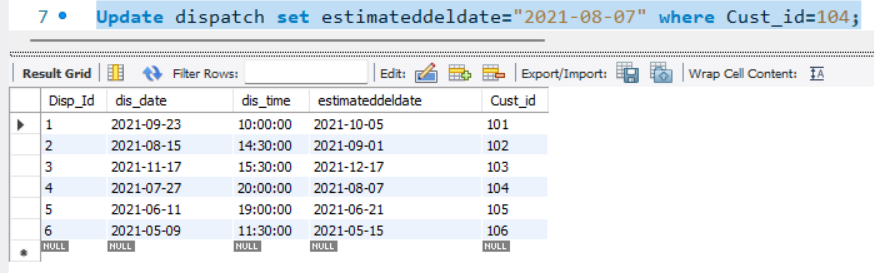
d.estimateddeldate from package as p join dispatch as d on

(p.Disp\_Id=d.Disp\_Id AND (typeofpack="Luggage" or typeofpack="Medical Supplies"));



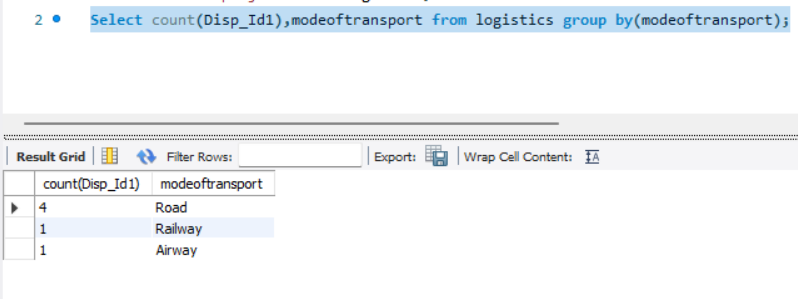
1. There is a delay of two days for delivering the package of the customer with id 104 so update the database for the same by writing an appropriate query.

Update dispatch set estimateddeldate="2021-08-07" where Cust\_id=104;



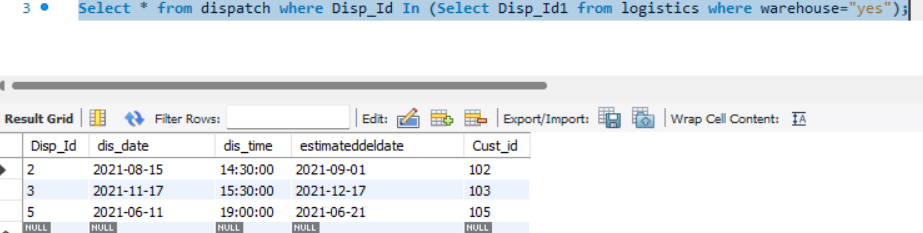
1. Write a query to separate the packages based on their mode of transport.

Select count(Disp\_Id1),modeoftransport from logistics group by(modeoftransport);



1. Write a query to fetch the dispatch details of those customers whose packages are still in the warehouse.

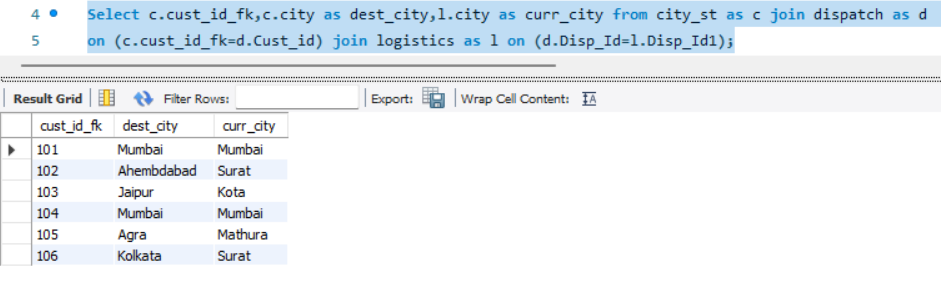
Select \* from dispatch where Disp\_Id In (Select Disp\_Id1 from logistics where warehouse="yes");



1. Write a query to fetch the status of a package if it has reached its destination city.

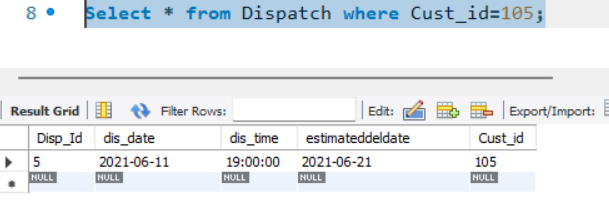
Select c.cust\_id\_fk,c.city as dest\_city,l.city as curr\_city from city\_st as c join dispatch as d

on (c.cust\_id\_fk=d.Cust\_id) join logistics as l on (d.Disp\_Id=l.Disp\_Id1);



1. Write a query to fetch dispatch details of customer with id 105.

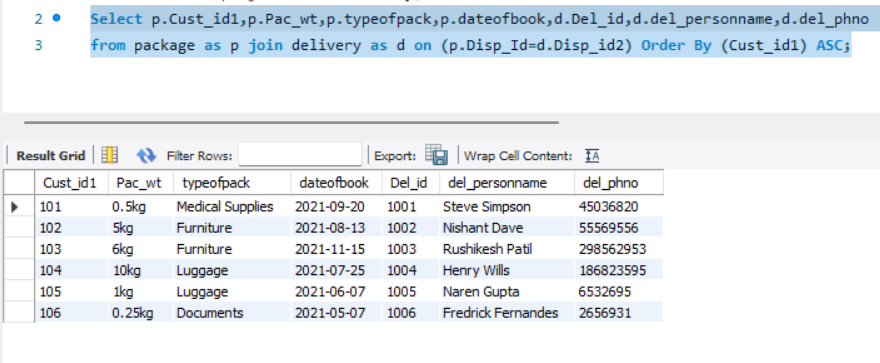
Select \* from Dispatch where Cust\_id=105;



1. Write a query to fetch delivery personnel details for each customer's package.

Selectp.Cust\_id1,p.Pac\_wt,p.typeofpack,p.dateofbook,d.Del\_id,d.del\_personname,

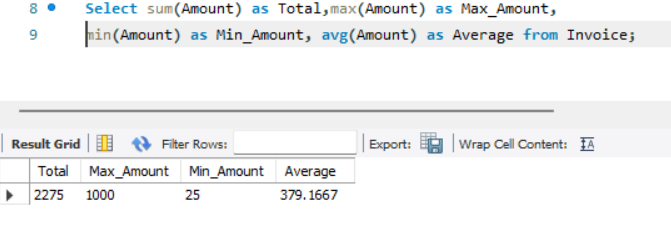
d.del\_phno from package as p join delivery as d on (p.Disp\_Id=d.Disp\_id2) Order By (Cust\_id1) ASC;



1. Write a query to obtain maximum, minimum,total and average cost earned by our courier service.

Select sum(Amount) as Total,max(Amount) as Max\_Amount,

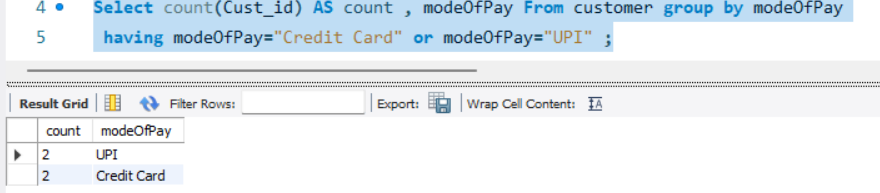
min(Amount) as Min\_Amount, avg(Amount) as Average from Invoice;



1. Write a query that separates customers based on the mode of payment they choose, displaying only when the mode of payment is not cash.

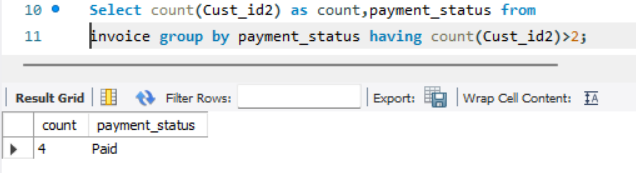
Select count(Cust\_id) AS count , modeOfPay From customer group by modeOfPay

having modeOfPay="Credit Card" or modeOfPay="UPI" ;



1. Write a query that separates customers based on payment status and displays only if the number of customers is greater than two.

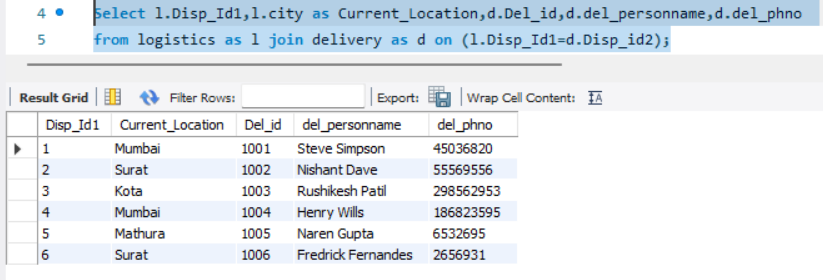
Select count(Cust\_id2) as count,payment\_status from invoice group by payment\_status having count(Cust\_id2)>2;



1. Write a query to determine the current location of each package and the personnel responsible for its delivery.

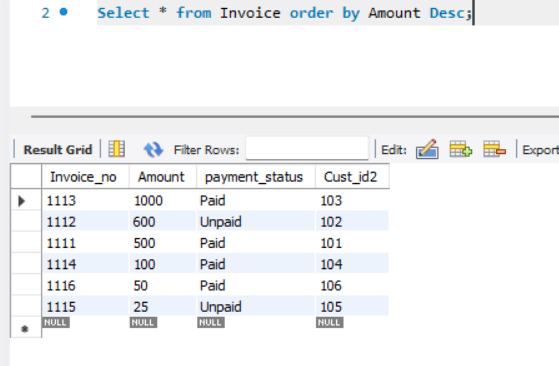
Select l.Disp\_Id1,l.city as Current\_Location,d.Del\_id,d.del\_personname,d.del\_phno

from logistics as l join delivery as d on (l.Disp\_Id1=d.Disp\_id2);



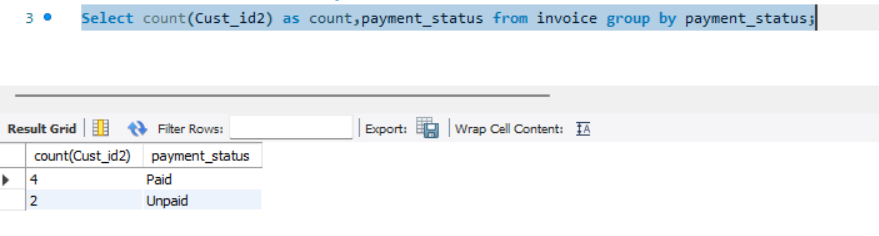
1. Write a query to sort the table from largest to smallest amount.

Select \* from Invoice order by Amount Desc;



1. Write a query to separate customers based on their payment status.

Select count(Cust\_id2) as count,payment\_status from invoice group by payment\_status;



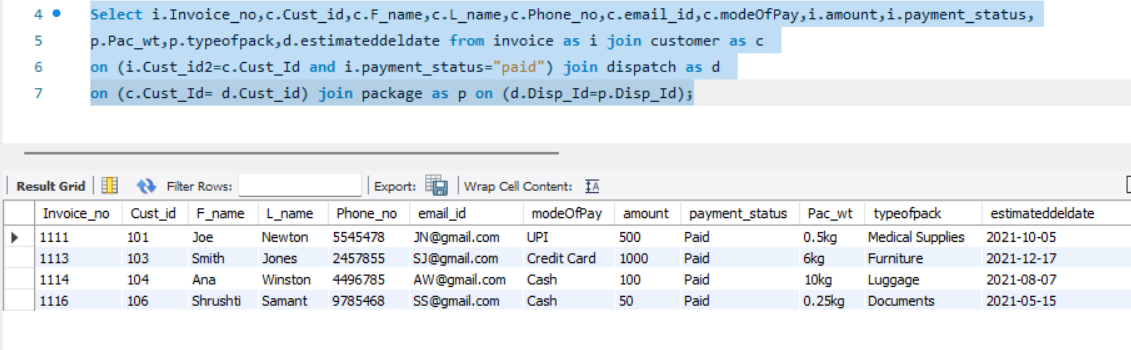
1. Write a query to generate an Invoice receipt to the customer.

Select

i.Invoice\_no,c.Cust\_id,c.F\_name,c.L\_name,c.Phone\_no,c.email\_id,c.modeOfPay,i.amount,i.payment\_status,p.Pac\_wt,p.typeofpack,d.estimateddeldate from invoice as i join customer as c

on (i.Cust\_id2=c.Cust\_Id and i.payment\_status="paid") join dispatch as d

on (c.Cust\_Id= d.Cust\_id) join package as p on (d.Disp\_Id=p.Disp\_Id);

****