**Ecommerce – SQL**

create database ecommerce;

use ecommerce;

**SQL Tables:**

1. customers table:

• customer\_id (Primary Key)

• name

• email

• password

1. products table:

• product\_id (Primary Key)

• name

• price

• description

• stockQuantity

1. cart table:

• cart\_id (Primary Key)

• customer\_id (Foreign Key)

• product\_id (Foreign Key)

• quantity

1. orders table:

• order\_id (Primary Key)

• customer\_id (Foreign Key)

• order\_date

• total\_price

• shipping\_address

1. order\_items table (to store order details):

• order\_item\_id (Primary Key)

• order\_id (Foreign Key)

• product\_id (Foreign Key)

• quantity

**Customer table:**

create table customers(

customer\_id int primary key,

name varchar(50),

email varchar(200),

password varchar(50));

alter table customers

drop column password;

alter table customers

add column address varchar(500);

**Product table:**

create table products(

product\_id int primary key,

name varchar(200),

price decimal(10,2),

description varchar(500),

stockQuantity int);

**Cart table:**

create table cart(

card\_id int primary key,

customer\_id int,

product\_id int,

quantity int,

foreign key (customer\_id) references customers(customer\_id) ,

foreign key (product\_id) references products(product\_id));

**Order table:**

create table orders(

order\_id int primary key,

customer\_id int,

order\_date date,

total\_price decimal(10,2),

shipping\_address varchar(500),

foreign key (customer\_id) references customers(customer\_id));

alter table orders

drop column shipping\_address;

**Order items table:**

create table order\_items(

order\_item\_id int primary key,

order\_id int,

product\_id int,

quantity int,

itemAmount decimal(10,2),

foreign key (order\_id) references orders(order\_id),

foreign key (product\_id) references products(product\_id));

insert into **customers** values

(1, 'john doe', 'johndoe@example.com', '123 main st, city'),

(2, 'jane smith', 'janesmith@example.com', '456 elm st, town'),

(3, 'robert johnson', 'robert@example.com', '789 oak st, village'),

(4, 'sarah brown', 'sarah@example.com', '101 pine st, suburb'),

(5, 'david lee', 'david@example.com', '234 cedar st, district'),

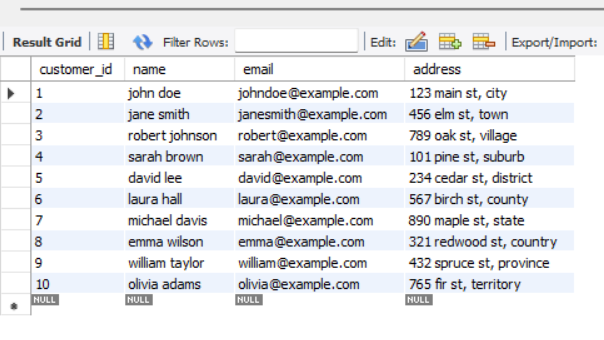
(6, 'laura hall', 'laura@example.com', '567 birch st, county'),

(7, 'michael davis', 'michael@example.com', '890 maple st, state'),

(8, 'emma wilson', 'emma@example.com', '321 redwood st, country'),

(9, 'william taylor', 'william@example.com', '432 spruce st, province'),

(10, 'olivia adams', 'olivia@example.com', '765 fir st, territory');



insert into **products** (product\_id, name, description, price, stockQuantity)values

(1, 'laptop', 'high-performance laptop', 800.00, 10),

(2, 'smartphone', 'latest smartphone', 600.00, 15),

(3, 'tablet', 'portable tablet', 300.00, 20),

(4, 'headphones', 'noise-canceling', 150.00, 30),

(5, 'tv', '4k smart tv', 900.00, 5),

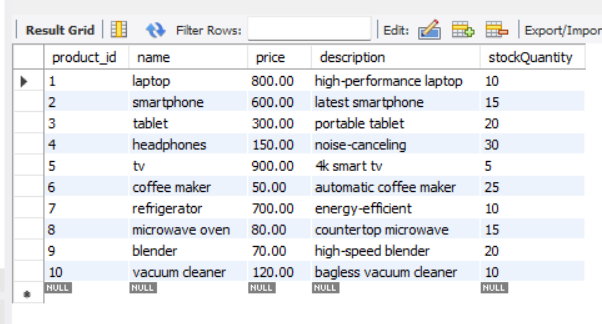
(6, 'coffee maker', 'automatic coffee maker', 50.00, 25),

(7, 'refrigerator', 'energy-efficient', 700.00, 10),

(8, 'microwave oven', 'countertop microwave', 80.00, 15),

(9, 'blender', 'high-speed blender', 70.00, 20),

(10, 'vacuum cleaner', 'bagless vacuum cleaner', 120.00, 10);



insert into **cart** (card\_id, customer\_id, product\_id, quantity) values

(1, 1, 1, 2),

(2, 1, 3, 1),

(3, 2, 2, 3),

(4, 3, 4, 4),

(5, 3, 5, 2),

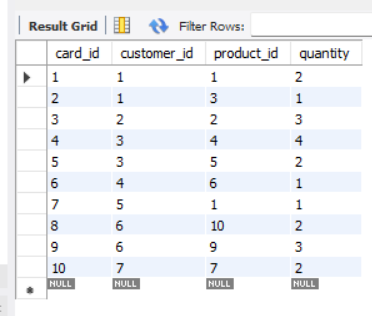
(6, 4, 6, 1),

(7, 5, 1, 1),

(8, 6, 10, 2),

(9, 6, 9, 3),

(10, 7, 7, 2);



insert into **orders** values

(1, 1, '2023-01-05', 1200.00),

(2, 2, '2023-02-10', 900.00),

(3, 3, '2023-03-15', 300.00),

(4, 4, '2023-04-20', 150.00),

(5, 5, '2023-05-25', 1800.00),

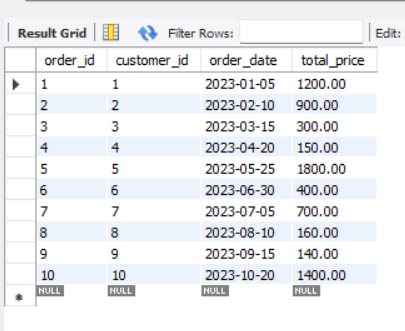
(6, 6, '2023-06-30', 400.00),

(7, 7, '2023-07-05', 700.00),

(8, 8, '2023-08-10', 160.00),

(9, 9, '2023-09-15', 140.00),

(10, 10, '2023-10-20', 1400.00);



insert into **order\_items** values

(1, 1, 1, 2, 1600.00),

(2, 1, 3, 1, 300.00),

(3, 2, 2, 3, 1800.00),

(4, 3, 5, 2, 1800.00),

(5, 4, 4, 4, 600.00),

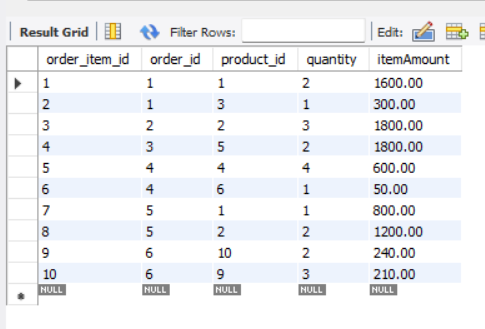
(6, 4, 6, 1, 50.00),

(7, 5, 1, 1, 800.00),

(8, 5, 2, 2, 1200.00),

(9, 6, 10, 2, 240.00),

(10, 6, 9, 3, 210.00);



select \* from customers;

select \* from products;

select \* from cart;

select \* from orders;

select \* from order\_items;

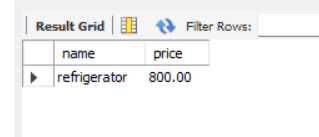
**1. Update refrigerator product price to 800.**

update products

set price = 800.00

where product\_id=7;

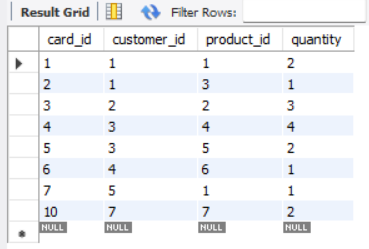
select name, price from products where product\_id=7;



**2. Remove all cart items for a specific customer**

delete from cart

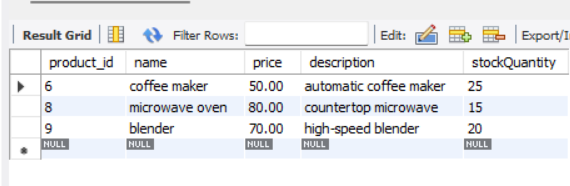
where customer\_id = 6;



**3. Retrieve Products Priced Below $100.**

select \* from products

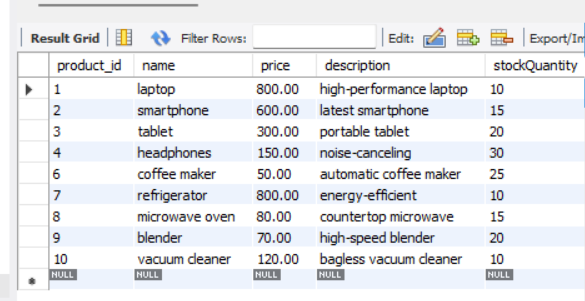
where price<100;



**4. Find Products with Stock Quantity Greater Than 5**

select \* from products

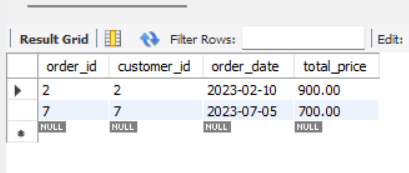
where stockQuantity>5;



**5. Retrieve Orders with Total Amount Between $500 and $1000**.

select \* from orders

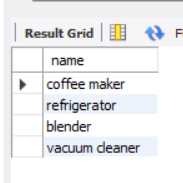
where total\_price between 500 and 1000;



**6. Find Products which name end with letter ‘r’.**

select name from products

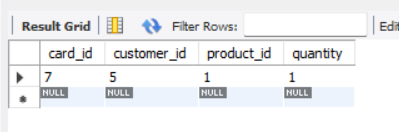
where name like "%r";



**7. Retrieve Cart Items for Customer 5.**

select \* from cart

where customer\_id = 5;

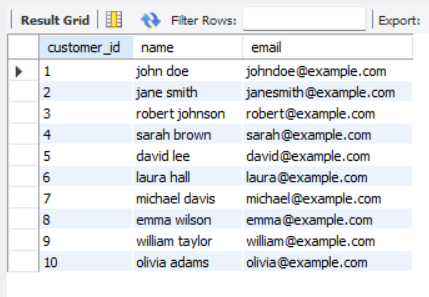


**8. Find Customers Who Placed Orders in 2023.**

select c.customer\_id,c.name,c.email from customers c

join orders o on c.customer\_id = o.customer\_id

where year(o.order\_date)=2023;



**9. Determine the Minimum Stock Quantity for Each Product Category**

alter table products

add column category varchar(100);

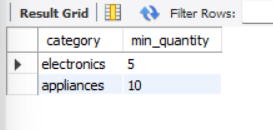
update products set category = 'electronics' where product\_id in (1, 2, 3, 4, 5);

update products set category = 'appliances' where product\_id in (6, 7, 8, 9, 10);

select category, min(stockQuantity) as min\_quantity

from products

group by category;

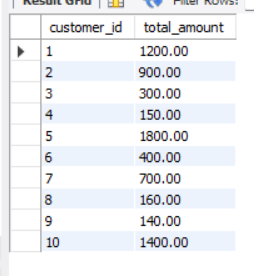


**10. Calculate the Total Amount Spent by Each Customer.**

select customer\_id, sum(total\_price) as total\_amount

from orders

group by customer\_id;

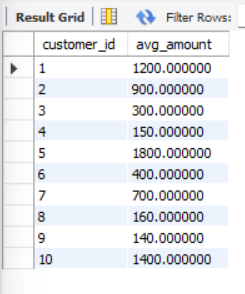


**11. Find the Average Order Amount for Each Customer.**

select customer\_id, avg(total\_price) as avg\_amount

from orders

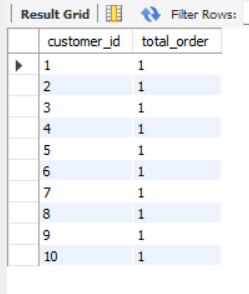
group by customer\_id;



**12. Count the Number of Orders Placed by Each Customer**

select customer\_id, count(order\_id) as total\_order from orders

group by customer\_id;

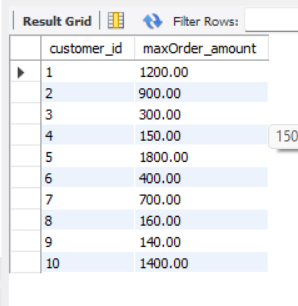


**13. Find the Maximum Order Amount for Each Customer**

select customer\_id, max(total\_price) as maxOrder\_amount

from orders

group by customer\_id;



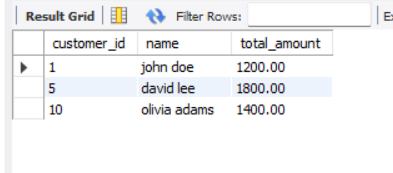
**14. Get Customers Who Placed Orders Totaling Over $1000.**

select c.customer\_id,c.name,sum(total\_price) as total\_amount from customers c

join orders o on c.customer\_id = o.customer\_id

group by o.customer\_id

having sum(total\_price)>1000;

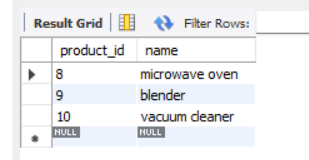


**15. Subquery to Find Products Not in the Cart.**

select product\_id,name from products

where product\_id not in

(select product\_id from cart);

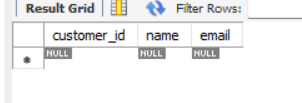


**16. Subquery to Find Customers Who Haven't Placed Orders.**

select customer\_id, name,email from customers

where customer\_id not in

(select customer\_id from orders);



**17. Subquery to Calculate the Percentage of Total Revenue for a Product.**

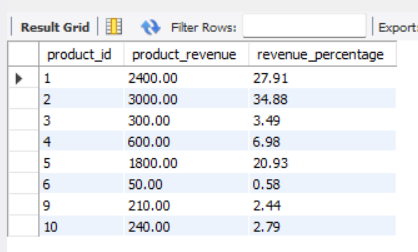
select product\_id, sum(itemAmount) as product\_revenue,round(

(sum(itemAmount) \* 100.0) /

(select sum(itemAmount) from order\_items), 2) as revenue\_percentage

from order\_items

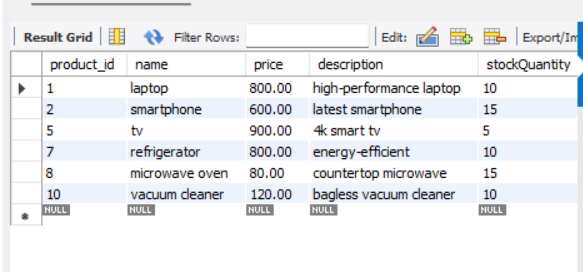
group by product\_id;



**18. Subquery to Find Products with Low Stock.**

select \* from products

where stockQuantity < (select avg(stockQuantity) from products);



**19. Subquery to Find Customers Who Placed High-Value Orders.**

select \*

from customers

where customer\_id in

(select customer\_id from orders where total\_price > 1000);

