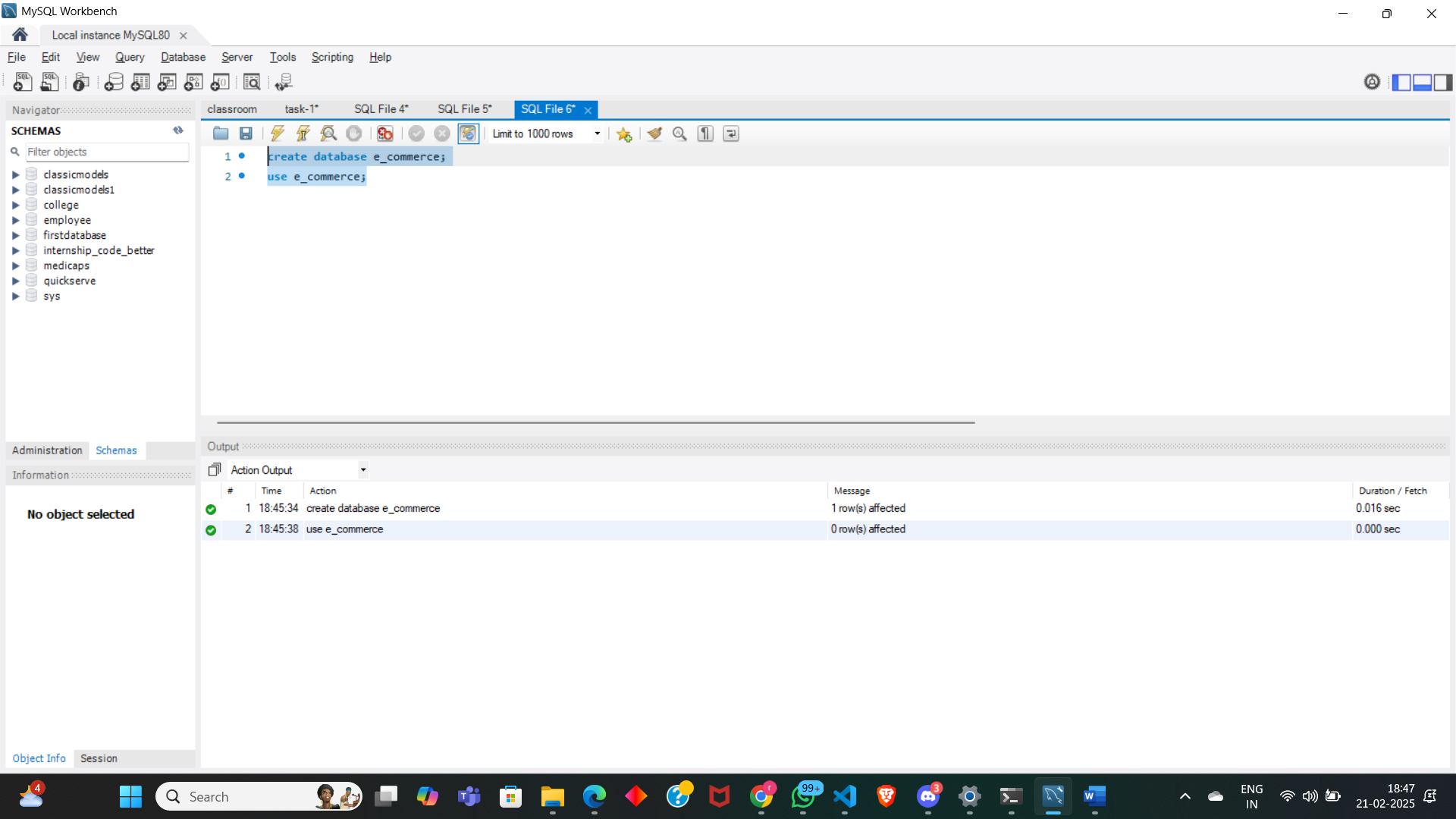
**Basic SQL Training Questions**

**1. Create Database e\_commerce.**

Query:

create database e\_commerce;

use e\_commerce;



**2. Create following Tables:**

Customers:

a. customer\_id - int auto-increment primary key

b. name - varchar(50)

c. email - varchar(50)

d. mobile - varchar(15)

Products:

a. id - int

b. name - varchar(50) not null

c. description - varchar(200)

d. price - decimal(10, 2) not null

e. category - varchar(50)

Query:

create table Customers(

customer\_id int auto\_increment primary key,

name varchar(50),

email varchar(50),

mobile varchar(15) );

create table Products(

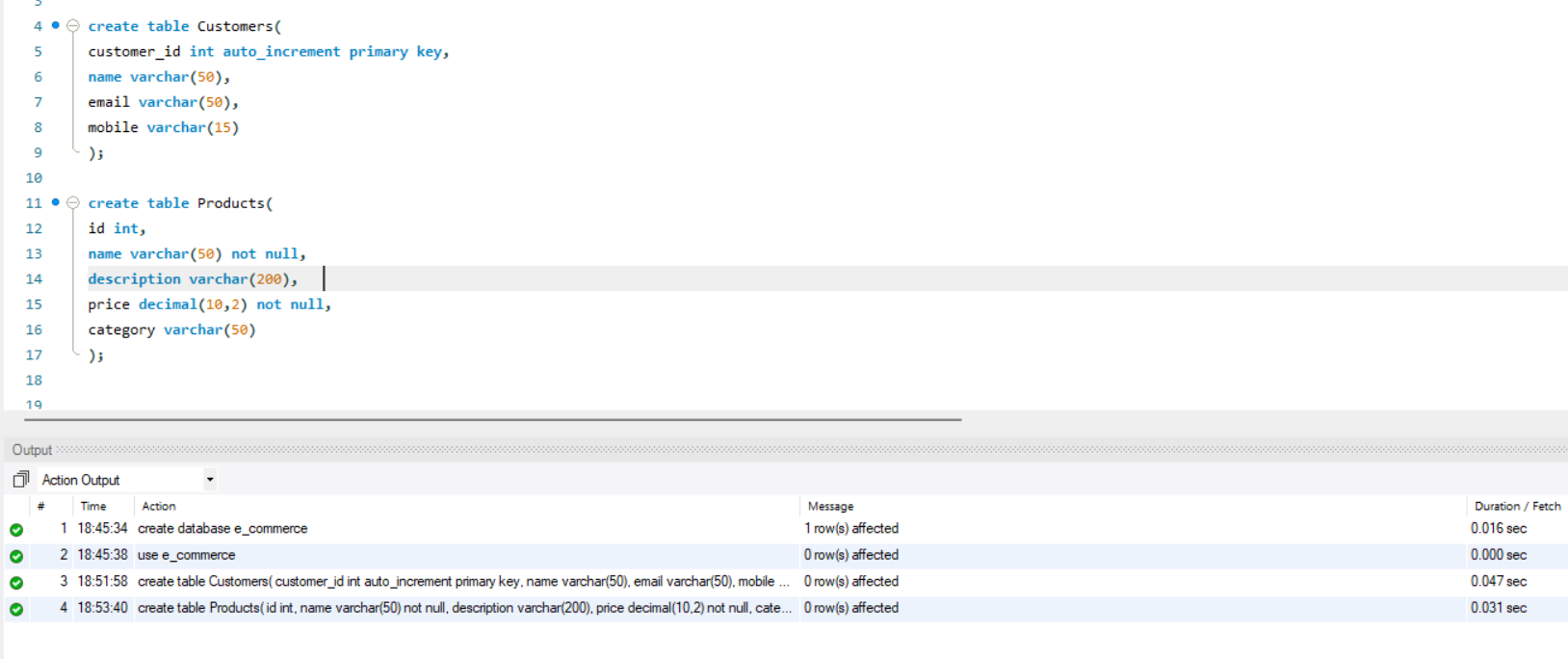
id int,

name varchar(50) not null,

description varchar(200),

price decimal(10,2) not null,

category varchar(50) );



**3. Modify Tables(using Alter keyword):**

a. Add not null on name and email in the Customers table

b. Add unique key on email in the Customers table

c. Add column age in the Customers table

d. Change column name from id to product\_id in the Products table;

e. Add primary key and auto increment on product\_id in the Products table

f. Change datatype of description from varchar to text in the Products table

Query:

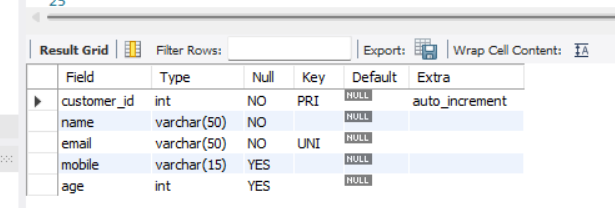
alter table Customers

modify name varchar(50) not null,

modify email varchar(50) not null,

add constraint email unique(email),

add column age int;



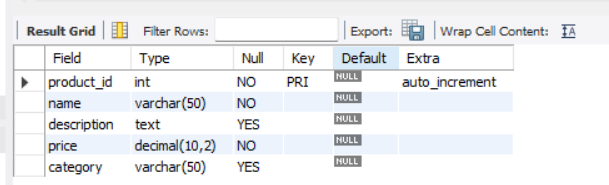
alter table Products

change column id product\_id int;

alter table Products

modify product\_id int auto\_increment primary key,

modify description text;



**4. Create table Order:**

a. order\_id - int auto-increment primary key

b. customer\_id - int -foreign key

c. product\_id - int

d. quantity - int not null,

e. order\_date - date not null,

f. status - enum(Pending, Success, Cancel),

g. payment\_method - enum(Credit, Debit, UPI),

h. total\_amount - decimal(10, 2) not null

Query:

create table `Order`(

order\_id int auto\_increment primary key,

customer\_id int,

product\_id int,

quantity int not null,

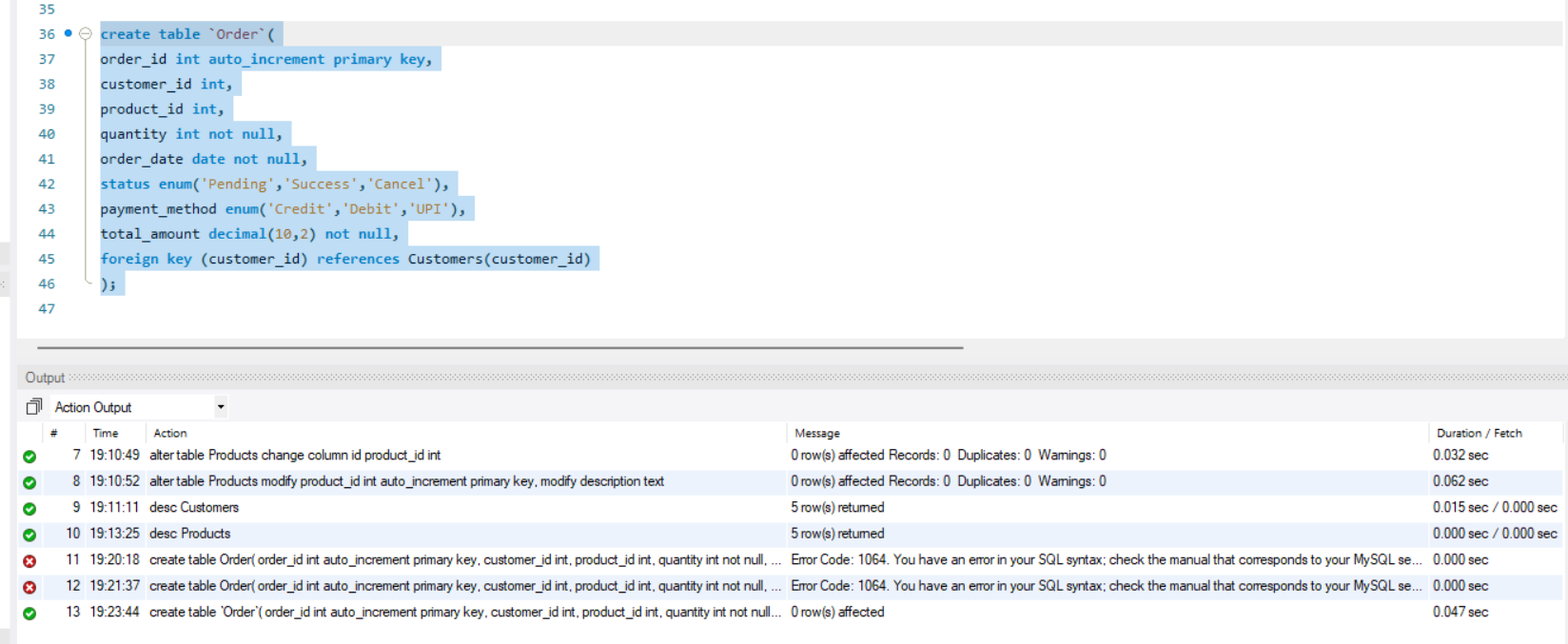
order\_date date not null,

status enum('Pending','Success','Cancel'),

payment\_method enum('Credit','Debit','UPI'),

total\_amount decimal(10,2) not null,

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



**5. Modify Orders Table(using Alter keyword):**

a. Change table name Order -> Orders

b. Set default value pending in status.

c. Modify payment\_method ENUM to add one more value: 'COD'

d. Make product id as foreign key

Query:

alter table Orders

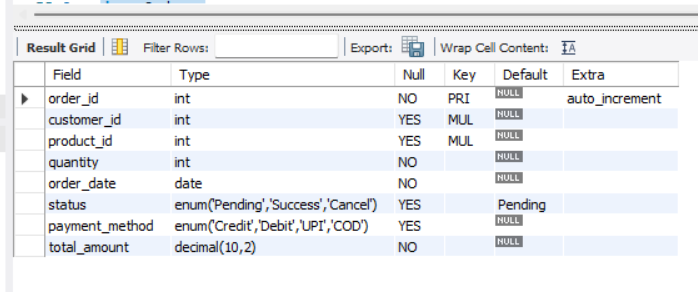
modify status enum('Pending','Success','Cancel') default 'Pending',

modify payment\_method enum('Credit','Debit','UPI','COD');

alter table Orders

add constraint fk\_orders\_product

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



**6. Insert 20 sample records in all the tables.**

Query:

insert into Customers (name, email, mobile, age) values

('Rohit Muchhal', 'rohit.muchhal@example.com', '9876543210', 21),

('Priya Patel', 'priya.patel@example.com', '8765432109', 25),

('Amit Rathore', 'amit.rathore@example.com', '7654321098', 30),

('Neha Gupta', 'neha.gupta@example.com', '6543210987', 22),

('Vikram Yadav', 'vikram.yadav@example.com', '5432109876', 35),

('Anjali Desai', 'anjali.desai@example.com', '4321098765', 27),

('Rajesh Kumar', 'rajesh.kumar@example.com', '3210987654', 40),

('Rudransh Reddy', 'Rudransh.reddy@example.com', '2109876543', 29),

('Arun Jain', 'arun.jain@example.com', '1098765432', 33),

('Pooja Choudhary', 'pooja.choudhary@example.com', '9988776655', 26),

('Sanjay Lela', 'sanjay.lela@example.com', '8877665544', 31),

('Kavita Joshi', 'kavita.joshi@example.com', '7766554433', 24),

('Akshat Tiwari', 'akshat.tiwari@example.com', '6655443322', 38),

('Sunita Mehta', 'sunita.mehta@example.com', '5544332211', 36),

('Deepak Rao', 'deepak.rao@example.com', '4433221100', 32),

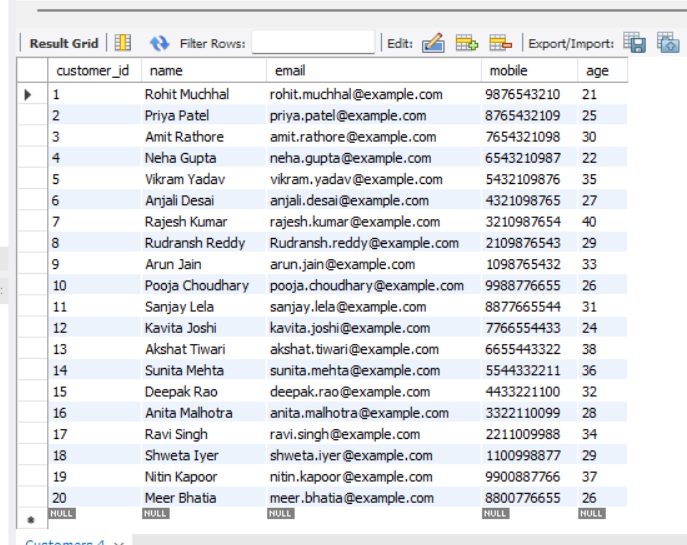
('Anita Malhotra', 'anita.malhotra@example.com', '3322110099', 28),

('Ravi Singh', 'ravi.singh@example.com', '2211009988', 34),

('Shweta Iyer', 'shweta.iyer@example.com', '1100998877', 29),

('Nitin Kapoor', 'nitin.kapoor@example.com', '9900887766', 37),

('Meer Bhatia', 'meer.bhatia@example.com', '8800776655', 26);



insert into Products (name, description, price, category) values

('Laptop', 'High-performance laptop with 16GB RAM and 512GB SSD', 903.61, 'Electronics'),

('Smartphone', 'Latest smartphone with 128GB storage and 48MP camera', 301.20, 'Electronics'),

('Tablet', '10-inch tablet with 64GB storage and 8MP camera', 216.87, 'Electronics'),

('Desktop Computer', 'Powerful desktop with 8GB RAM and 1TB HDD', 542.17, 'Electronics'),

('Smartwatch', 'Fitness tracker with heart rate monitor and GPS', 144.58, 'Electronics'),

('Headphones', 'Noise-cancelling over-ear headphones', 60.24, 'Electronics'),

('Printer', 'Wireless all-in-one printer', 108.43, 'Electronics'),

('Camera', 'DSLR camera with 24MP sensor and 4K video', 662.65, 'Electronics'),

('Sofa', '3-seater leather sofa', 301.20, 'Furniture'),

('Dining Table', '6-seater wooden dining table', 216.87, 'Furniture'),

('Bed', 'Queen-size bed with storage', 361.45, 'Furniture'),

('Novel', 'Bestseller fiction novel', 3.61, 'Books'),

('Textbook', 'Engineering textbook', 14.46, 'Books'),

('T-Shirt', 'Cotton crew-neck t-shirt', 6.02, 'Clothing'),

('Jeans', 'Slim-fit denim jeans', 18.07, 'Clothing'),

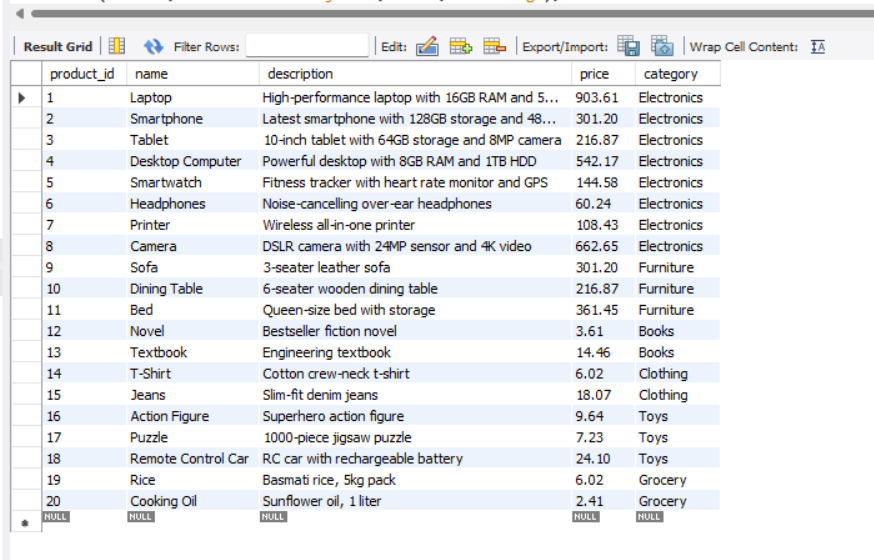
('Action Figure', 'Superhero action figure', 9.64, 'Toys'),

('Puzzle', '1000-piece jigsaw puzzle', 7.23, 'Toys'),

('Remote Control Car', 'RC car with rechargeable battery', 24.10, 'Toys'),

('Rice', 'Basmati rice, 5kg pack', 6.02, 'Grocery'),

('Cooking Oil', 'Sunflower oil, 1 liter', 2.41, 'Grocery');



insert into Orders (customer\_id, product\_id, quantity, order\_date, status, payment\_method, total\_amount) values

(1, 1, 1, '2025-10-01', 'Success', 'Credit', 903.61),

(2, 2, 2, '2025-10-02', 'Pending', 'Debit', 602.40),

(3, 3, 1, '2025-10-03', 'Success', 'UPI', 216.87),

(3, 6, 1, '2025-10-06', 'Pending', 'UPI', 60.24),

(4, 7, 1, '2025-10-07', 'Success', 'Credit', 108.43),

(5, 10, 1, '2025-10-09', 'Pending', 'UPI', 301.20),

(5, 10, 1, '2025-10-10', 'Success', 'Credit', 216.87),

(6, 4, 1, '2025-10-04', 'Cancel', 'Credit', 542.17),

(6, 12, 1, '2025-10-11', 'Success', 'Debit', 361.45),

(9, 12, 2, '2025-10-12', 'Success', 'UPI', 28.92),

(9, 13, 2, '2025-10-13', 'Success', 'Debit', 7.22),

(10, 13, 3, '2025-10-14', 'Success', 'UPI', 18.06),

(11, 15, 1, '2025-10-15', 'Cancel', 'COD', 18.07),

(12, 16, 1, '2025-10-16', 'Pending', 'COD', 9.64),

(12, 17, 1, '2025-10-17', 'Success', 'Credit', 7.23),

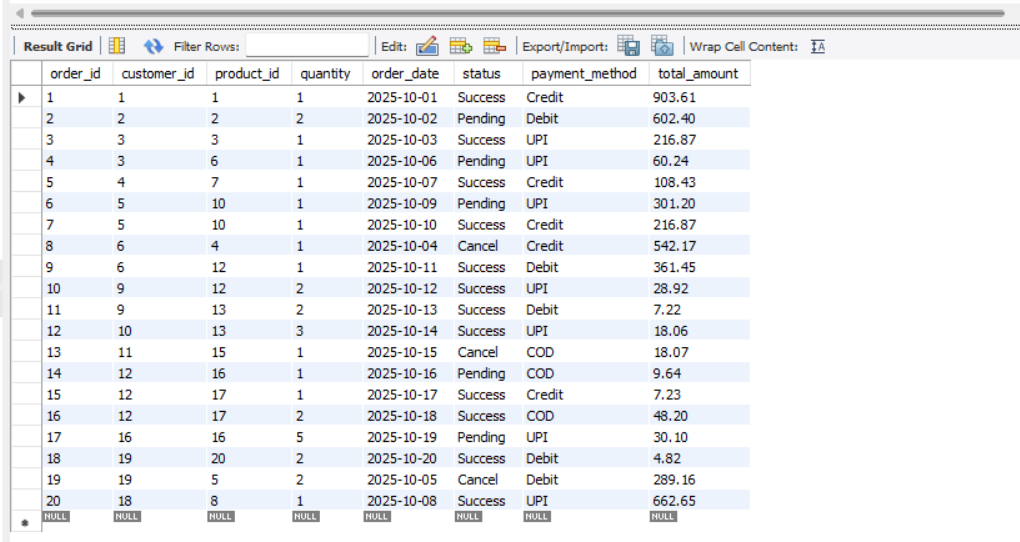
(12, 17, 2, '2025-10-18', 'Success', 'COD', 48.20),

(16, 16, 5, '2025-10-19', 'Pending', 'UPI', 30.10),

(19, 20, 2, '2025-10-20', 'Success', 'Debit', 4.82),

(19, 5, 2, '2025-10-05', 'Cancel', 'Debit', 289.16),

(18, 8, 1, '2025-10-08', 'Success', 'UPI', 662.65);



**7. Perform following queries:**

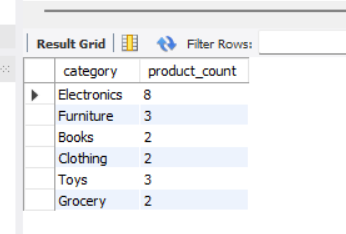
a. Count the number of products as product\_count in each category.

Query:

select category, count(\*) as product\_count

from Products

group by category;



b. Retrieve all products that belong to the 'Electronics' category, have a

price between $50 and $500, and whose name contains the letter 'a'.

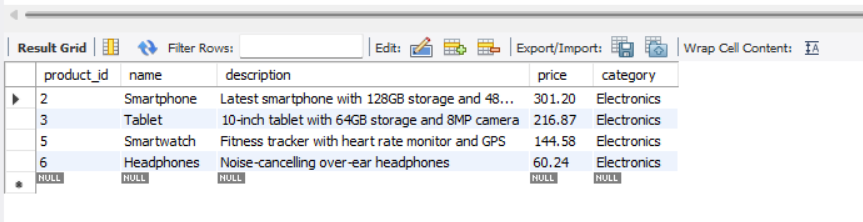
Query:

select \* from Products

where category = 'Electronics'

and price between 50 and 500

and name like '%a%';



c. Get the top 5 most expensive products in the 'Electronics' category,

skipping the first 2.

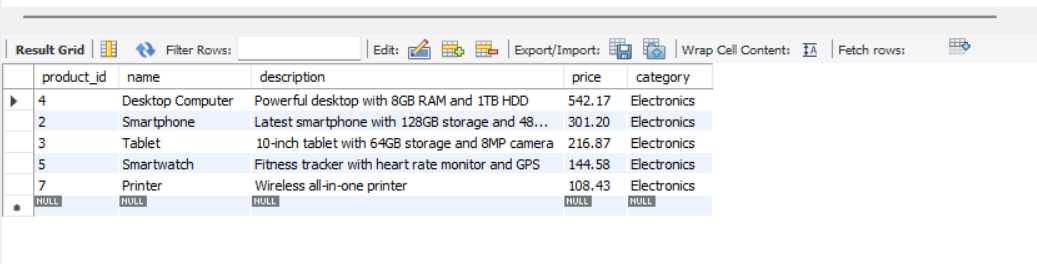
Query:

select \* from Products

where category = 'Electronics'

order by price desc

limit 5 offset 2;

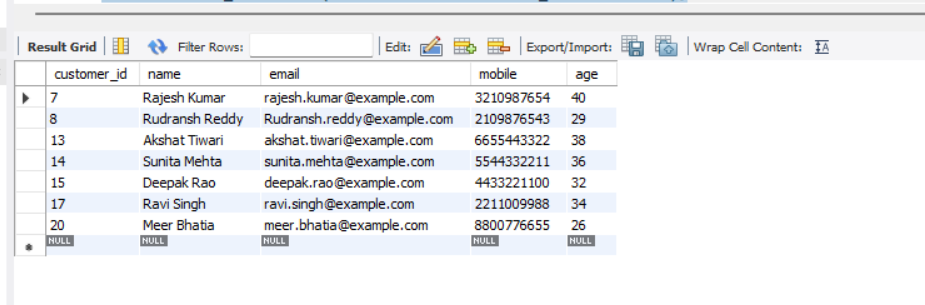


d. Retrieve customers who have not placed any orders.

Query:

select \* from Customers

where customer\_id not in (select distinct customer\_id from Orders);



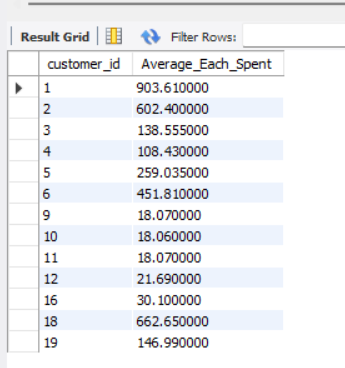
e. Find the average total amount spent by each customer.

Query:

select customer\_id, AVG(total\_amount) as Average\_Each\_Spent

from Orders

group by customer\_id;



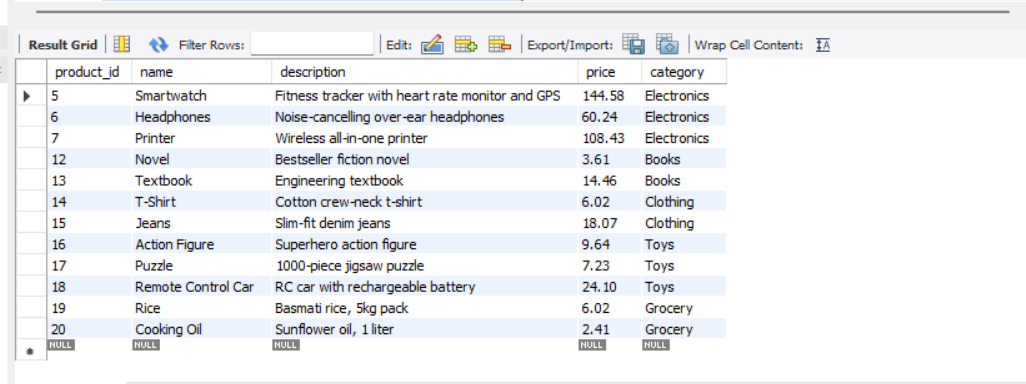
f. Get the products that have a price less than the average price of all

products.

Query:

select \* from Products

where price < (select AVG(price) from Products);



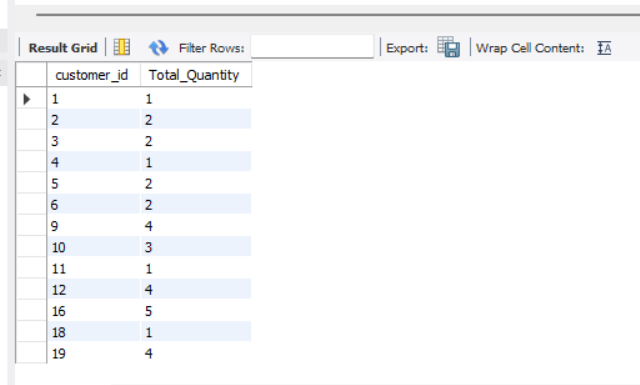
g. Calculate the total quantity of products ordered by each customer:

Query:

select customer\_id, SUM(quantity) as Total\_Quantity

from Orders

group by customer\_id;



h. List all orders along with customer name and product name.

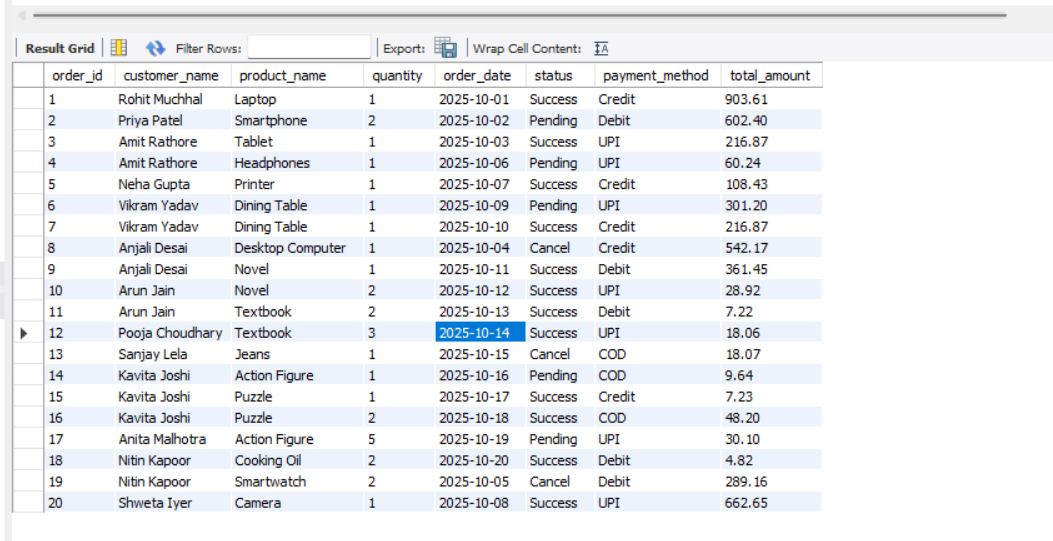
Query:

select O.order\_id, C.name AS customer\_name, P.name AS product\_name, O.quantity, O.order\_date, O.status, O.payment\_method, O.total\_amount

from Orders O

JOIN Customers C on O.customer\_id = C.customer\_id

JOIN Products P on O.product\_id = P.product\_id;



i. Find products that have never been ordered.

Query:

select \* from Products

where product\_id not in (select distinct product\_id from Orders);

