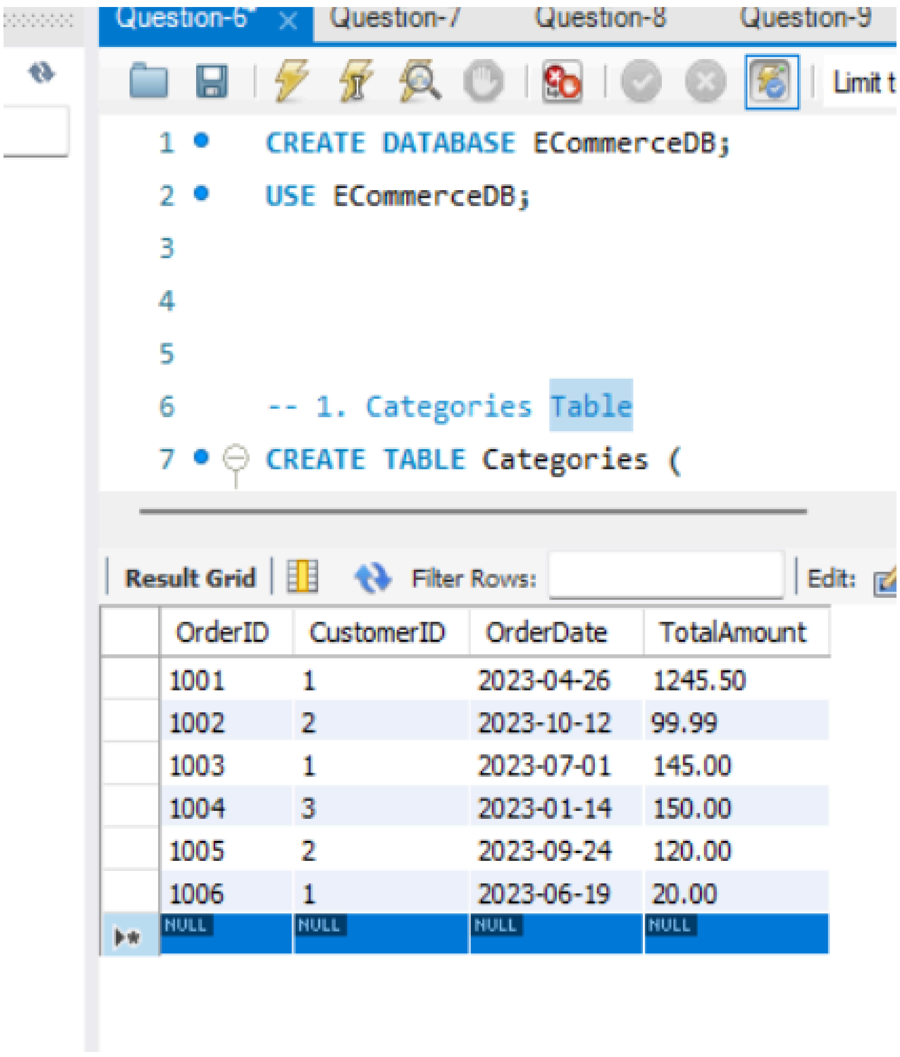
QUESTION-6

```
CREATE DATABASE ECommerceDB;
USE ECommerceDB;



-- 1. Categories Table
CREATE TABLE Categories (
    CategoryID INT PRIMARY KEY,
    CategoryName VARCHAR(50) NOT NULL UNIQUE
);

-- 2. Products Table
CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100) NOT NULL UNIQUE,
    CategoryID INT,
    Price DECIMAL(10,2) NOT NULL,
    StockQuantity INT,
    FOREIGN KEY (CategoryID) REFERENCES
Categories(CategoryID)
);

-- 3. Customers Table
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    CustomerName VARCHAR(100) NOT NULL,
    Email VARCHAR(100) UNIQUE,
    JoinDate DATE
);

-- 4. Orders Table
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE NOT NULL,
    TotalAmount DECIMAL(10,2),
    FOREIGN KEY (CustomerID) REFERENCES
Customers(CustomerID)
);

INSERT INTO Categories (CategoryID, CategoryName) VALUES
(1, 'Electronics'),
(2, 'Books'),
(3, 'Home Goods'),
(4, 'Apparel');

INSERT INTO Products (ProductID, ProductName, CategoryID,
Price, StockQuantity) VALUES
(101, 'Laptop Pro', 1, 1200.00, 50),
(102, 'SQL Handbook', 2, 45.50, 200),
(103, 'Smart Speaker', 1, 99.99, 150),
(104, 'Coffee Maker', 3, 75.00, 80),
(105, 'Novel : The Great SQL', 2, 25.00, 120),
(106, 'Wireless Earbuds', 1, 150.00, 100),
(107, 'Blender X', 3, 120.00, 60),
(108, 'T-Shirt Casual', 4, 20.00, 300);

INSERT INTO Customers (CustomerID, CustomerName, Email,
JoinDate) VALUES
(1, 'Alice Wonderland', 'alice@example.com', '2023-01-10'),
(2, 'Bob the Builder', 'bob@example.com', '2022-11-25'),
(3, 'Charlie Chaplin', 'charlie@example.com', '2023-03-01'),
(4, 'Diana Prince', 'diana@example.com', '2021-04-26');

INSERT INTO Orders (OrderID, CustomerID, OrderDate,
TotalAmount) VALUES
(1001, 1, '2023-04-26', 1245.50),
(1002, 2, '2023-10-12', 99.99),
(1003, 1, '2023-07-01', 145.00),
(1004, 3, '2023-01-14', 150.00),
(1005, 2, '2023-09-24', 120.00),
(1006, 1, '2023-06-19', 20.00);


SELECT * FROM Categories;
SELECT * FROM Products;
SELECT * FROM Customers;
SELECT * FROM Orders;
```

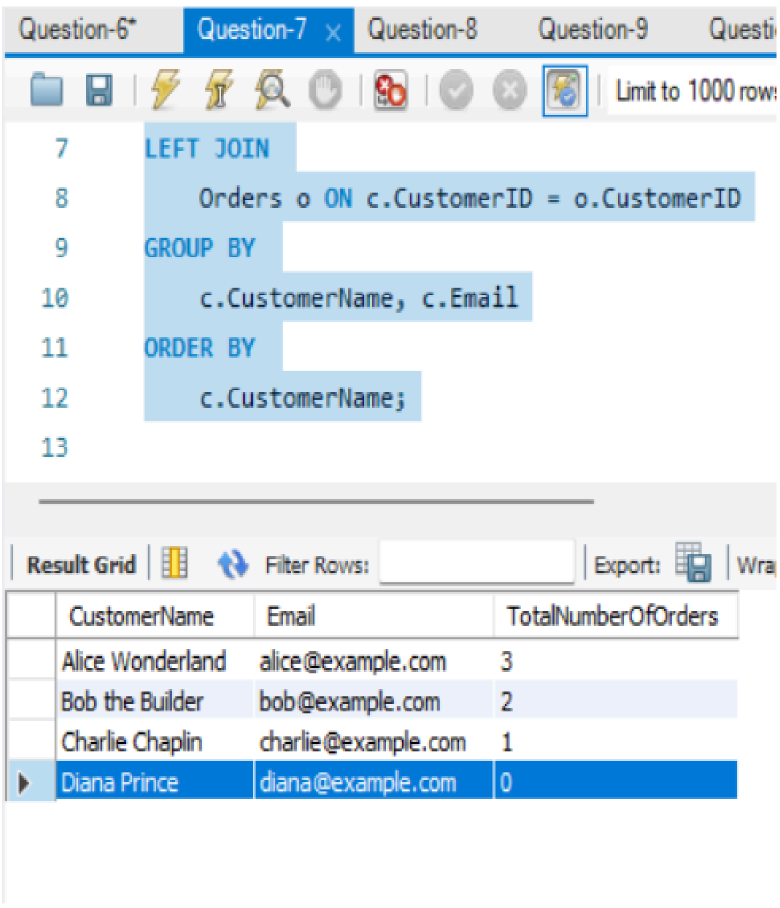


```
CREATE DATABASE ECommerceDB;
USE ECommerceDB;



-- 1. Categories Table
CREATE TABLE Categories (
```

OrderID	CustomerID	OrderDate	TotalAmount
1001	1	2023-04-26	1245.50
1002	2	2023-10-12	99.99
1003	1	2023-07-01	145.00
1004	3	2023-01-14	150.00
1005	2	2023-09-24	120.00
1006	1	2023-06-19	20.00
NULL	NULL	NULL	NULL

QUESTION-7

```
SELECT
    c.CustomerName,
    c.Email,
    COUNT(o.OrderID) AS TotalNumberOfOrders
FROM
    Customers c
LEFT JOIN
    Orders o ON c.CustomerID = o.CustomerID
GROUP BY
    c.CustomerName, c.Email
ORDER BY
    c.CustomerName;
```

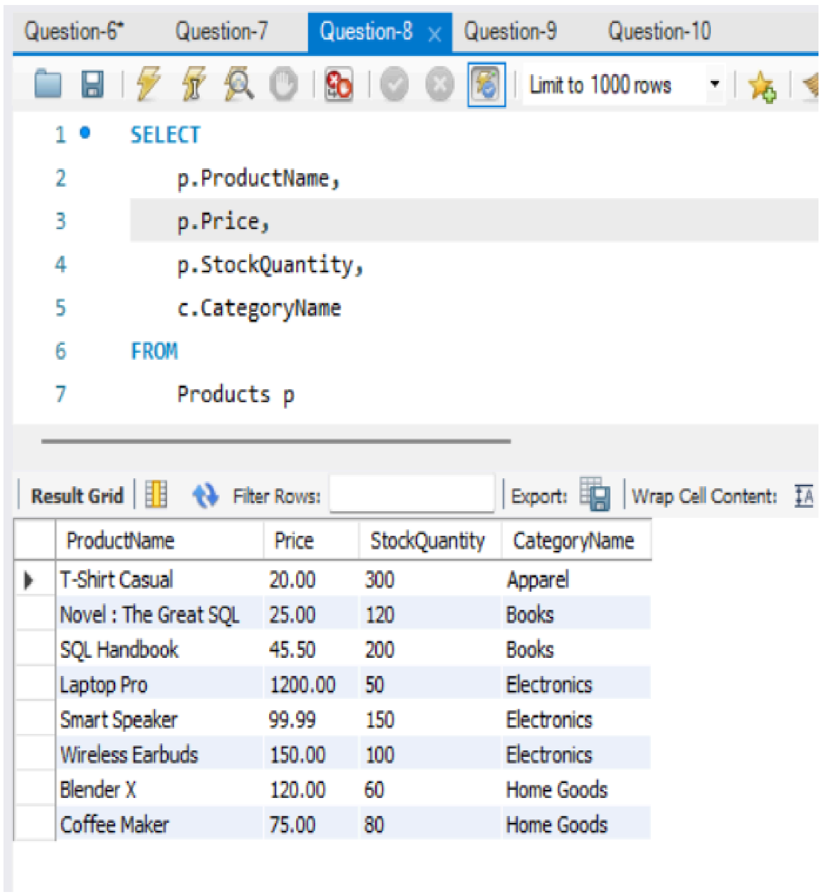


```
LEFT JOIN
    Orders o ON c.CustomerID = o.CustomerID
GROUP BY
    c.CustomerName, c.Email
ORDER BY
    c.CustomerName;
```

CustomerName	Email	TotalNumberOfOrders
Alice Wonderland	alice@example.com	3
Bob the Builder	bob@example.com	2
Charlie Chaplin	charlie@example.com	1
Diana Prince	diana@example.com	0

QUESTION-8

```
SELECT
    p.ProductName,
    p.Price,
    p.StockQuantity,
    c.CategoryName
FROM
    Products p
JOIN
    Categories c ON p.CategoryID = c.CategoryID
ORDER BY
    c.CategoryName,
    p.ProductName;
```

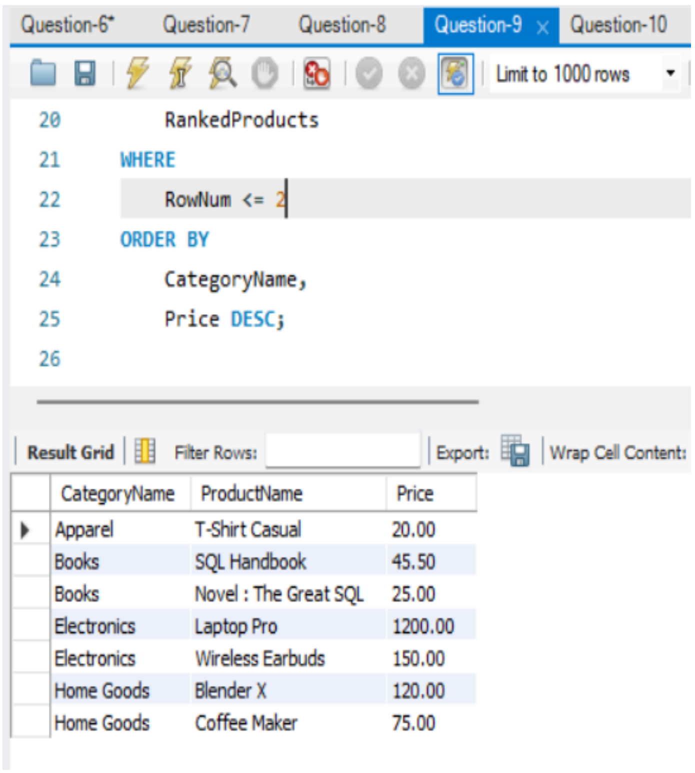


```
SELECT
    p.ProductName,
    p.Price,
    p.StockQuantity,
    c.CategoryName
FROM
    Products p
```

ProductName	Price	StockQuantity	CategoryName
T-Shirt Casual	20.00	300	Apparel
Novel : The Great SQL	25.00	120	Books
SQL Handbook	45.50	200	Books
Laptop Pro	1200.00	50	Electronics
Smart Speaker	99.99	150	Electronics
Wireless Earbuds	150.00	100	Electronics
Blender X	120.00	60	Home Goods
Coffee Maker	75.00	80	Home Goods

QUESTION-9

```
.WITH RankedProducts AS (
    SELECT
        c.CategoryName,
        p.ProductName,
        p.Price,
        ROW_NUMBER() OVER (
            PARTITION BY c.CategoryName
            ORDER BY p.Price DESC
        ) AS RowNum
    FROM
        Products p
    JOIN
        Categories c ON p.CategoryID = c.CategoryID
)
SELECT
    CategoryName,
    ProductName,
    Price
FROM
    RankedProducts
WHERE
    RowNum <= 2
ORDER BY
    CategoryName,
    Price DESC;
```



```
        RankedProducts
    WHERE
        RowNum <= 2
    ORDER BY
        CategoryName,
        Price DESC;
```

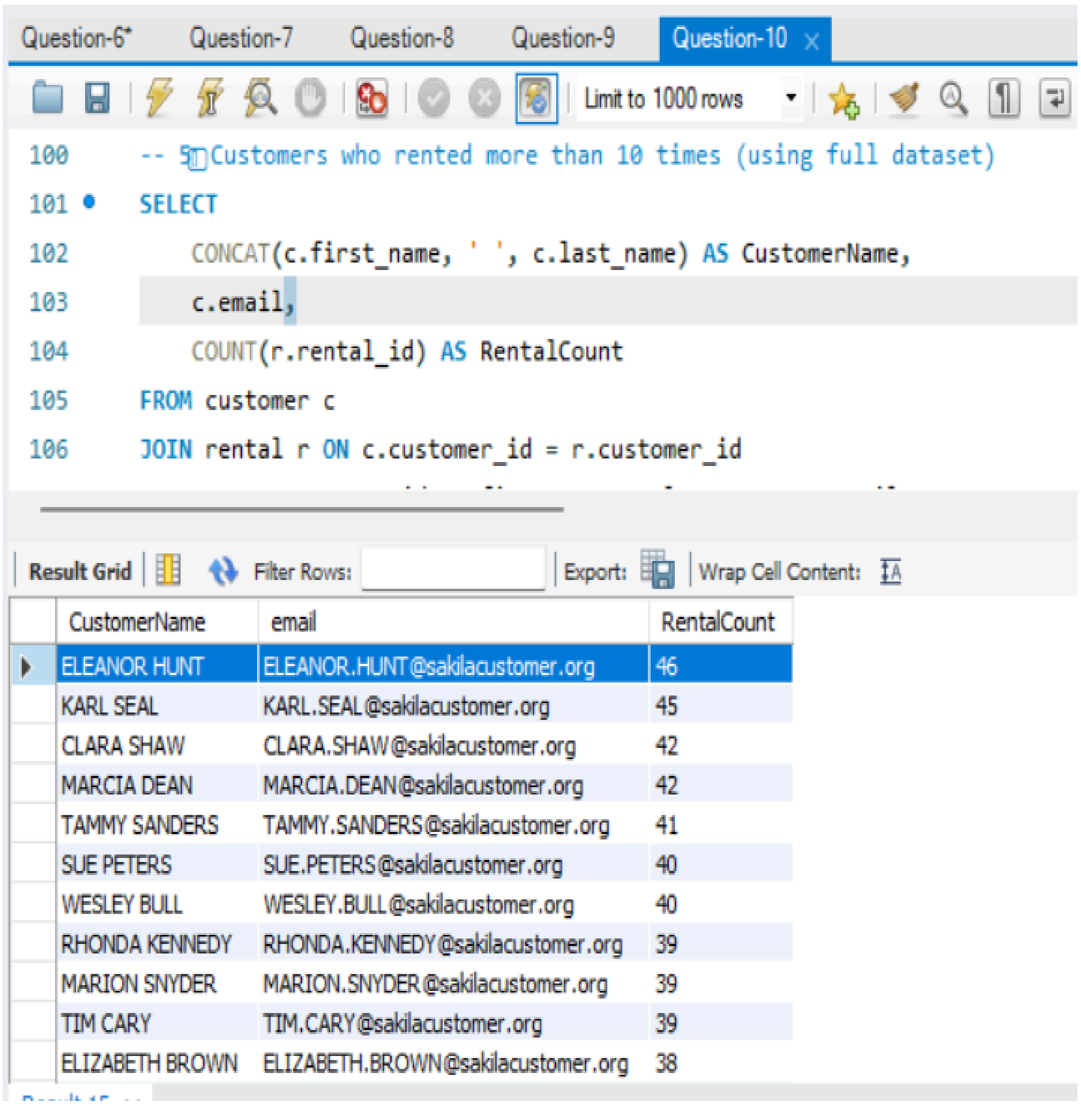
CategoryName	ProductName	Price
Apparel	T-Shirt Casual	20.00
Books	SQL Handbook	45.50
Books	Novel : The Great SQL	25.00
Electronics	Laptop Pro	1200.00
Electronics	Wireless Earbuds	150.00
Home Goods	Blender X	120.00
Home Goods	Coffee Maker	75.00

QUESTION-10

```
USE sakila;

-- Check first few rows of each key table
SELECT * FROM customer LIMIT 5;
SELECT * FROM payment LIMIT 5;
SELECT * FROM rental LIMIT 5;
SELECT * FROM category LIMIT 5;
SELECT * FROM film LIMIT 5;
SELECT * FROM inventory LIMIT 5;
SELECT * FROM store LIMIT 5;



SELECT
    'customer' AS table_name, COUNT(*) AS row_count FROM
customer
UNION ALL
SELECT 'payment', COUNT(*) FROM payment
UNION ALL
SELECT 'rental', COUNT(*) FROM rental
UNION ALL
SELECT 'film', COUNT(*) FROM film
UNION ALL
SELECT 'inventory', COUNT(*) FROM inventory
UNION ALL
SELECT 'category', COUNT(*) FROM category
UNION ALL
SELECT 'store', COUNT(*) FROM store;



USE sakila;
SELECT COUNT(*) FROM customer;
SELECT COUNT(*) FROM payment;
SELECT COUNT(*) FROM rental;


SELECT
   CONCAT(c.first_name, ' ', c.last_name) AS CustomerName,
   c.email,
   ROUND(SUM(p.amount), 2) AS TotalAmountSpent
FROM
   customer c
JOIN
   payment p ON c.customer_id = p.customer_id
GROUP BY
   c.customer_id
ORDER BY
   TotalAmountSpent DESC
LIMIT 5;




USE sakila;

-- Top 5 customers by total amount spent
SELECT
   CONCAT(c.first_name, ' ', c.last_name) AS CustomerName,
   c.email,
   ROUND(SUM(p.amount), 2) AS TotalAmountSpent
FROM customer c
JOIN payment p ON c.customer_id = p.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name, c.email
ORDER BY TotalAmountSpent DESC
LIMIT 5;
```

```
-- Top 3 movie categories by rental counts
SELECT
   cat.name AS CategoryName,
   COUNT(r.rental_id) AS RentalCount
FROM category cat
JOIN film_category fc ON cat.category_id = fc.category_id
JOIN inventory i ON fc.film_id = i.film_id
JOIN rental r ON i.inventory_id = r.inventory_id
GROUP BY cat.category_id, cat.name
ORDER BY RentalCount DESC
LIMIT 3;

-- Films available at each store and how many have never been
rented
SELECT
   s.store_id,
   COUNT(DISTINCT i.inventory_id) AS TotalFilmsAvailable,
   SUM(CASE WHEN r.rental_id IS NULL THEN 1 ELSE 0 END)
AS NeverRentedFilms
FROM store s
JOIN inventory i ON s.store_id = i.store_id
LEFT JOIN rental r ON i.inventory_id = r.inventory_id
GROUP BY s.store_id
ORDER BY s.store_id;

-- Total revenue per month (using actual years in dataset)
-- Sakila data is from 2005-2006
SELECT
   DATE_FORMAT(p.payment_date, '%Y-%m') AS Month,
   ROUND(SUM(p.amount), 2) AS TotalRevenue
FROM payment p
GROUP BY YEAR(p.payment_date), MONTH(p.payment_date)
ORDER BY Month;

-- Customers who rented more than 10 times (using full dataset)
```

```
SELECT
    CONCAT(c.first_name, ' ', c.last_name) AS CustomerName,
    c.email,
    COUNT(r.rental_id) AS RentalCount
FROM customer c
JOIN rental r ON c.customer_id = r.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name, c.email
HAVING COUNT(r.rental_id) > 10
ORDER BY RentalCount DESC;
```



```
-- 5 Customers who rented more than 10 times (using full dataset)
SELECT
    CONCAT(c.first_name, ' ', c.last_name) AS CustomerName,
    c.email,
    COUNT(r.rental_id) AS RentalCount
FROM customer c
JOIN rental r ON c.customer_id = r.customer_id
```

CustomerName	email	RentalCount
ELEANOR HUNT	ELEANOR.HUNT@sakilacustomer.org	46
KARL SEAL	KARL.SEAL@sakilacustomer.org	45
CLARA SHAW	CLARA.SHAW@sakilacustomer.org	42
MARCIA DEAN	MARCIA.DEAN@sakilacustomer.org	42
TAMMY SANDERS	TAMMY.SANDERS@sakilacustomer.org	41
SUE PETERS	SUE.PETERS@sakilacustomer.org	40
WESLEY BULL	WESLEY.BULL@sakilacustomer.org	40
RHONDA KENNEDY	RHONDA.KENNEDY@sakilacustomer.org	39
MARION SNYDER	MARION.SNYDER@sakilacustomer.org	39
TIM CARY	TIM.CARY@sakilacustomer.org	39
ELIZABETH BROWN	ELIZABETH.BROWN@sakilacustomer.org	38