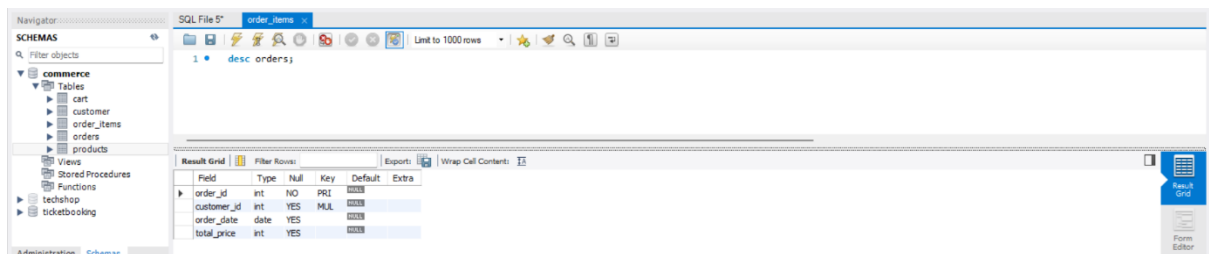


NAME: R. SURYA PRAKASH

CODING CHALLENGE 6_ECOMMERCE

Schema of Tables:

1.Orders

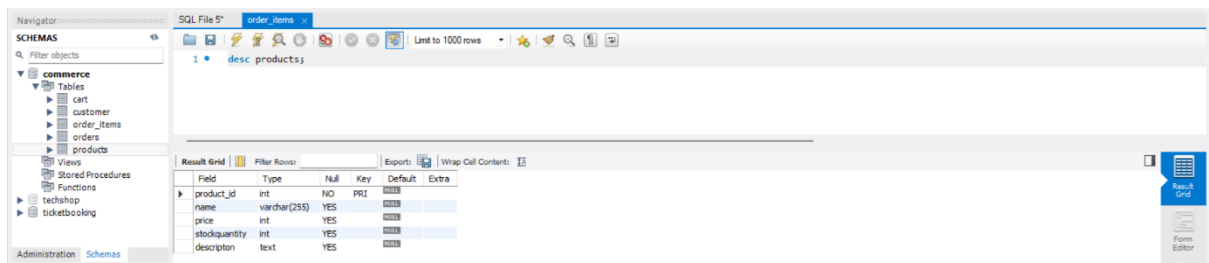


SQL File 5* order_items

1 • desc orders;

Field	Type	Null	Key	Default	Extra
order_id	int	NO	PK		
customer_id	int	YES	FK		
order_date	date	YES			
total_price	int	YES			

2.products

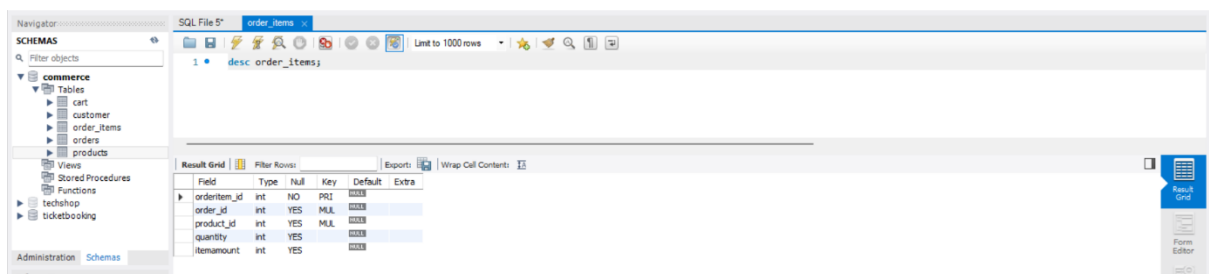


SQL File 5* order_items

1 • desc products;

Field	Type	Null	Key	Default	Extra
product_id	int	NO	PK		
name	varchar(255)	YES			
price	int	YES			
stockquantity	int	YES			
description	text	YES			

3.Orderitems

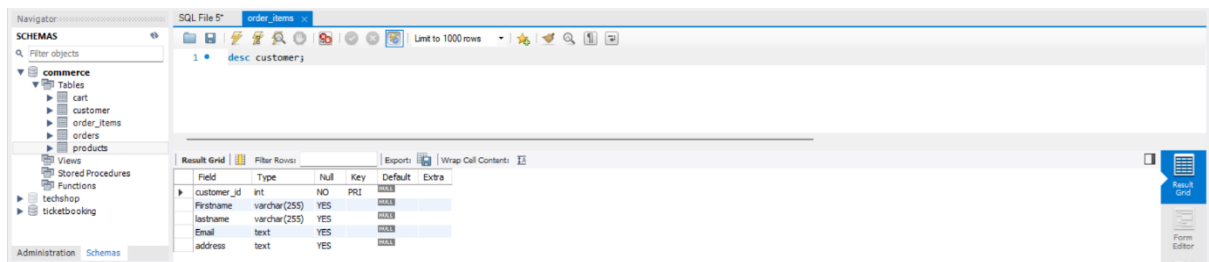


SQL File 5* order_items

1 • desc order_items;

Field	Type	Null	Key	Default	Extra
orderitem_id	int	NO	PK		
order_id	int	YES	FK		
product_id	int	YES	FK		
quantity	int	YES			
itemamount	int	YES			

4.customer



5.cart

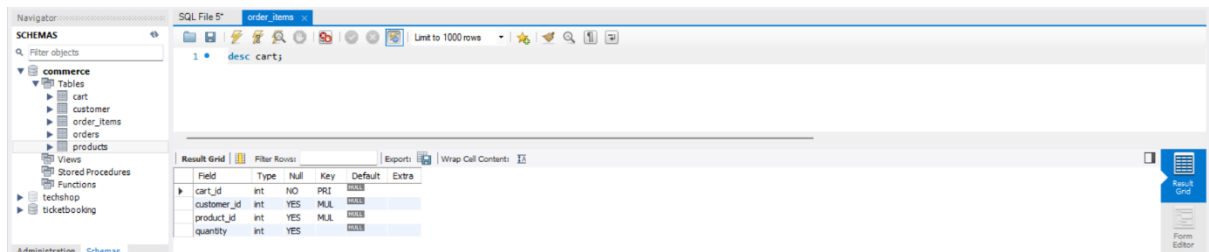
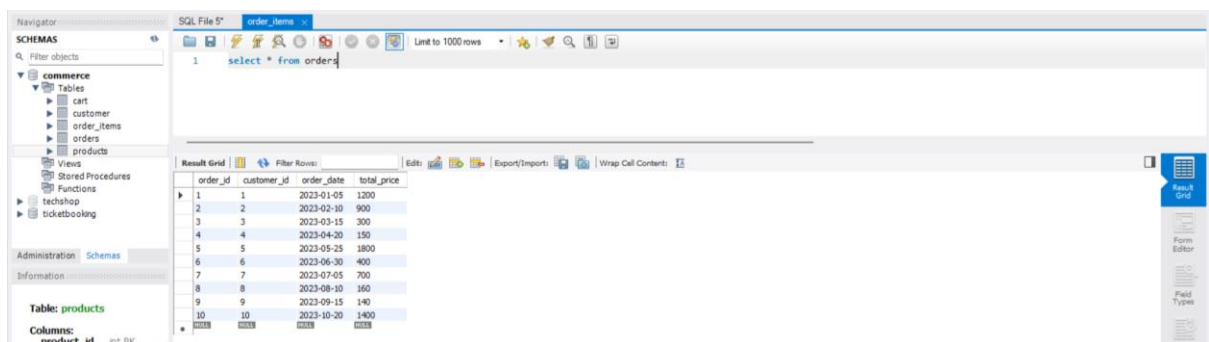


Table values:

1.Orders



2.products

SQL File 5* order_items

1 • select * from products;

product_id	name	price	stockquantity	description
1	Laptop	800	10	High-pe High-performance laptop
2	Latest smartphone	600	15	Smartphone
3	Tablet	300	20	Portable tablet
4	Headphones	150	30	Noise-canceling
5	TV	900	5	Smart TV
6	Coffee Maker	50	25	Automatic coffee maker
7	Refrigerator	700	10	Energy-efficient
8	Microwave Oven	80	15	Countertop microwave
9	Blender	70	20	High-speed blender
10	Vacuum Cleaner	120	10	Bagless vacuum cleaner

3.orderitems

SQL File 5* order_items

1 • select * from order_items;

orderitem_id	order_id	product_id	quantity	itemamount
1	1	1	2	1600
2	1	3	1	300
3	2	2	3	1800
4	3	5	2	1800
5	4	4	4	600
6	4	6	1	50
7	5	1	1	800
8	5	2	2	1200
9	6	10	2	240
10	6	9	3	210

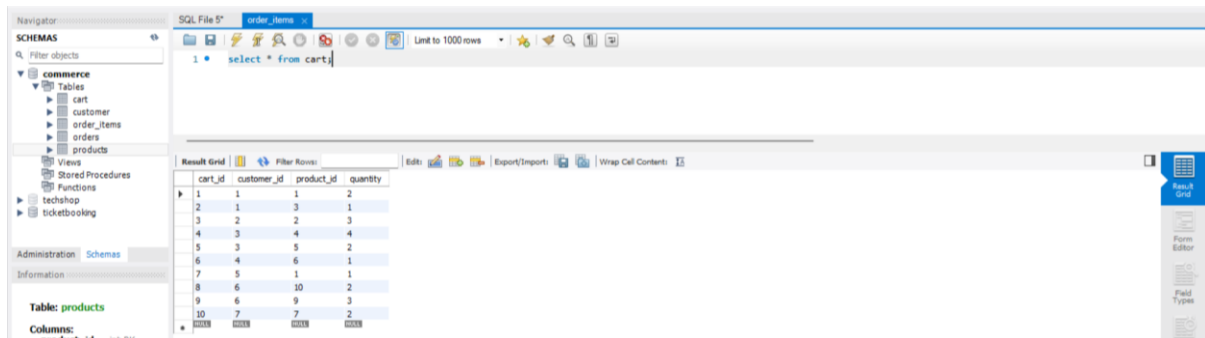
4.Customer

SQL File 5* order_items

1 • select * from customer;

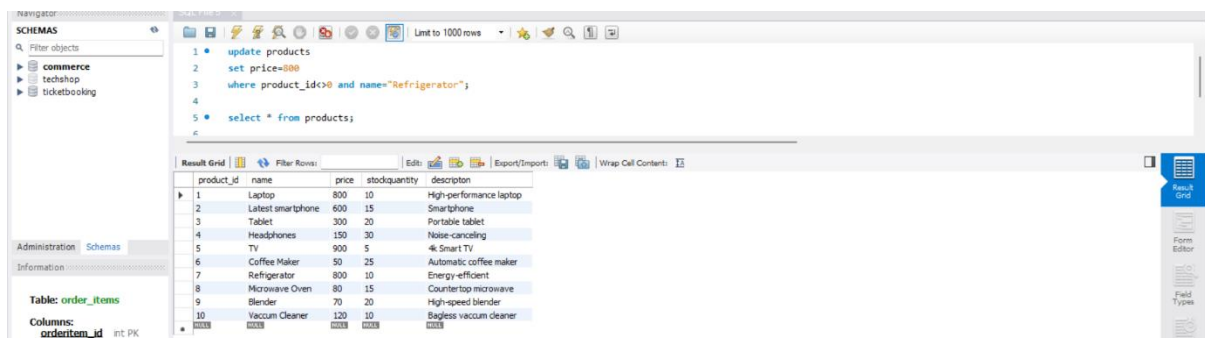
customer_id	Firstname	lastname	Email	address
1	John	Doe	john.doe@example.com	123 Main St, City
2	Jane	Smith	jane.smith@example.com	456 Elm St Town
3	Robert	Johnson	robert@example.com	789 Oak St Village
4	Sarah	Brown	sarah@example.com	101 Pine St Suburb
5	David	Lee	david@example.com	234 Cedar St District
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

5.Cart

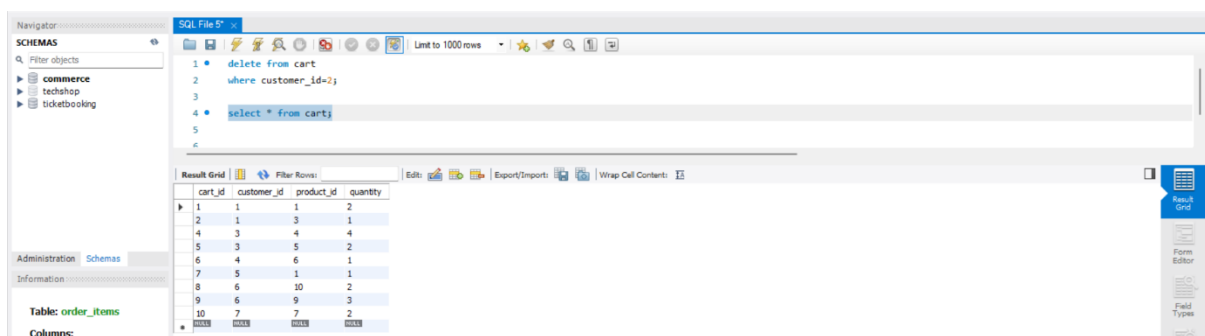


Queries:

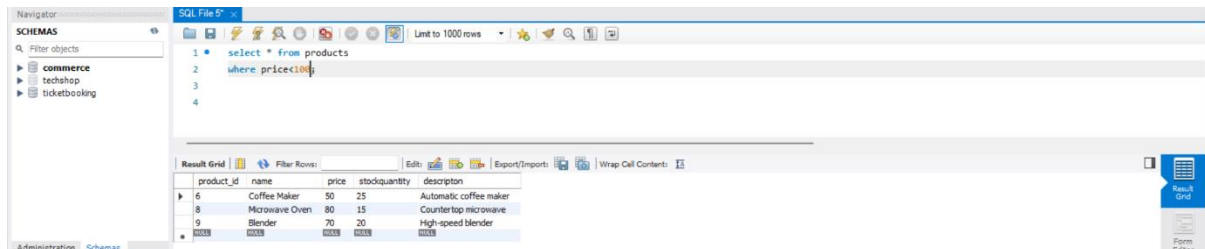
1. Update refrigerator product price to 800.



2. Remove all cart items for a specific customer.



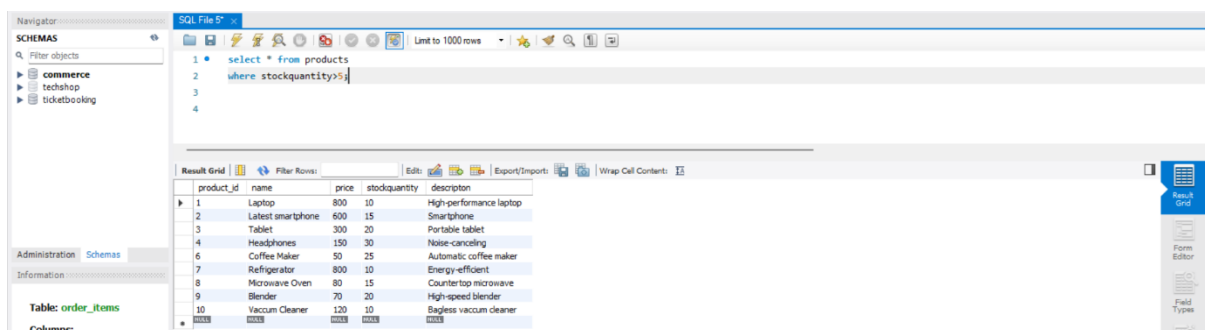
3. Retrieve Products Priced Below \$100.



The screenshot shows a SQL IDE interface. The SQL editor contains the query: `select * from products where price < 100`. The result grid displays the following data:

product_id	name	price	stockquantity	description
6	Coffee Maker	50	25	Automatic coffee maker
8	Microwave Oven	80	15	Countertop microwave
9	Blender	70	20	High-speed blender

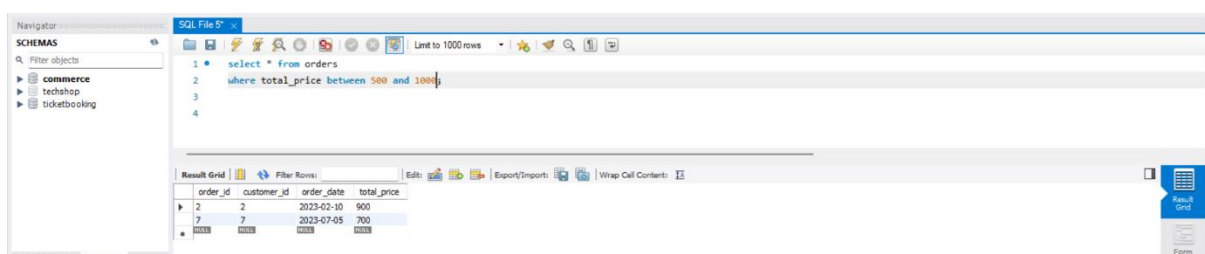
4. Find Products with Stock Quantity Greater Than 5.



The screenshot shows a SQL IDE interface. The SQL editor contains the query: `select * from products where stockquantity > 5`. The result grid displays the following data:

product_id	name	price	stockquantity	description
1	Laptop	800	10	High-performance laptop
2	Latest smartphone	600	15	Smartphone
3	Tablet	300	20	Portable tablet
4	Headphones	150	30	Noise-canceling
6	Coffee Maker	50	25	Automatic coffee maker
7	Refrigerator	800	10	Energy-efficient
8	Microwave Oven	80	15	Countertop microwave
9	Blender	70	20	High-speed blender
10	Vacuum Cleaner	120	10	Bagless vacuum cleaner

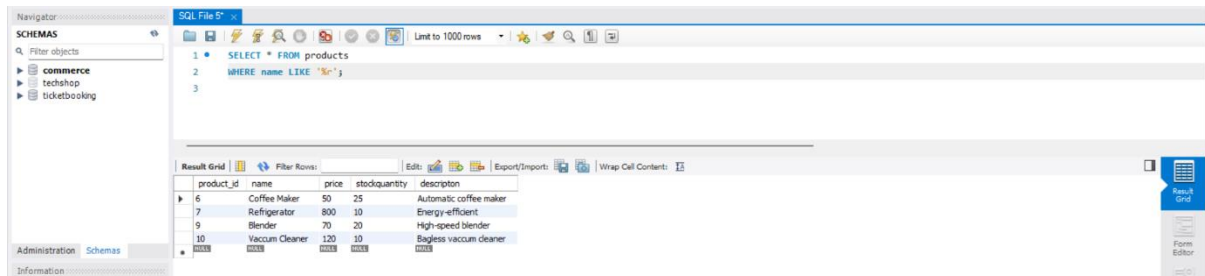
5. Retrieve Orders with Total Amount Between \$500 and \$1000.



The screenshot shows a SQL IDE interface. The SQL editor contains the query: `select * from orders where total_price between 500 and 1000`. The result grid displays the following data:

order_id	customer_id	order_date	total_price
2	2	2023-02-10	600
7	7	2023-07-05	700

6. Find Products which name end with letter 'r'



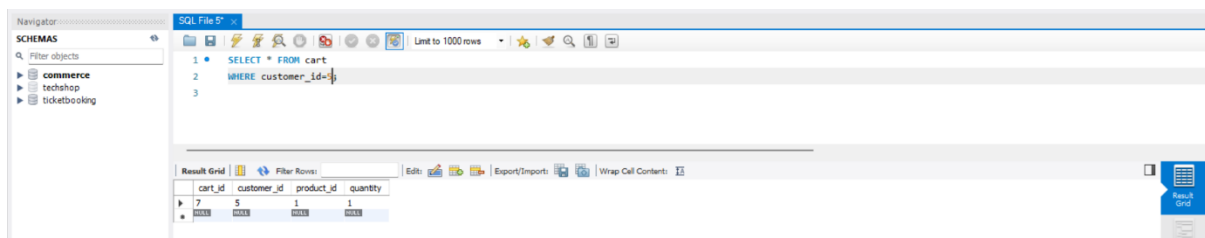
SQL File 5

```
1 SELECT * FROM products
2 WHERE name LIKE '%r';
3
```

Result Grid

product_id	name	price	stockquantity	description
6	Coffee Maker	50	25	Automatic coffee maker
7	Refrigerator	800	10	Energy-efficient
9	Blender	70	20	High-speed blender
10	Vacuum Cleaner	120	10	Bagless vacuum cleaner

7. Retrieve Cart Items for Customer 5.



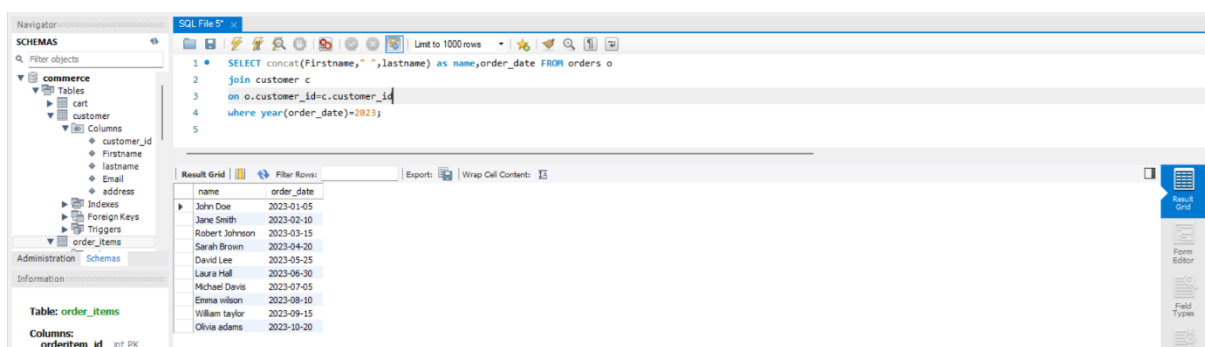
SQL File 5

```
1 SELECT * FROM cart
2 WHERE customer_id=5;
3
```

Result Grid

cart_id	customer_id	product_id	quantity
7	5	1	1

8. Find Customers Who Placed Orders in 2023.



SQL File 5

```
1 SELECT concat(Firstname,"",Lastname) as name,order_date FROM orders o
2 join customer c
3 on o.customer_id=c.customer_id
4 where year(order_date)=2023;
5
```

Result Grid

name	order_date
John Doe	2023-01-05
Jane Smith	2023-02-10
Robert Johnson	2023-03-15
Sarah Brown	2023-04-20
David Lee	2023-05-25
Laura Hall	2023-06-30
Michael Davis	2023-07-05
Emma Wilson	2023-08-10
William Taylor	2023-09-15
Olivia Adams	2023-10-20

Table: order_items
Columns: order_id PK

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

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree with 'products' expanded. The central pane shows the following SQL query:

```
1 SELECT Product_id, Name as product_name, MIN(StockQuantity) AS MinStockQuantity
2 FROM products
3 GROUP BY Product_name;
```

The right pane shows the 'Result Grid' with the following data:

Product_id	product_name	MinStockQuantity
1	Laptop	10
2	Latest smartphone	15
3	Tablet	20
4	Headphones	30
5	TV	5
6	Coffee Maker	25
7	Refrigerator	10
8	Microwave Oven	15
9	Blender	20
10	Vacuum Cleaner	10

10. Calculate the Total Amount Spent by Each Customer

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree with 'order_items' expanded. The central pane shows the following SQL query:

```
1 SELECT c.customer_id, concat(Firstname, " ", lastname) as name, o.total_price FROM orders o
2 join customer c
3 on o.customer_id=c.customer_id;
```

The right pane shows the 'Result Grid' with the following data:

customer_id	name	total_price
1	John Doe	1200
2	Jane Smith	900
3	Robert Johnson	300
4	Sarah Brown	150
5	David Lee	1800
6	Laura Hall	400
7	Michael Davis	700
8	Emma Wilson	350
9	William Taylor	140
10	Olivia Adams	1400

11. Find the Average Order Amount for Each Customer

SQL File 5

```

1 select customer_id, avg(total_price) as average_order_amount
2 from orders
3 group by customer_id

```

customer_id	average_order_amount
1	1200.0000
2	900.0000
3	300.0000
4	150.0000
5	1800.0000
6	400.0000
7	700.0000
8	160.0000
9	140.0000
10	1400.0000

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

SQL File 5

```

1 SELECT customer_id, COUNT(orders_id) AS NumberOfOrders
2 FROM orders
3 GROUP BY customer_id;
4

```

customer_id	NumberOfOrders
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

13. Find the Maximum Order Amount for Each Customer.

SQL File 5

```

1 select customer_id, max(total_price) as maximum_mount
2 from orders
3 group by customer_id

```

customer_id	maximum_mount
1	1200
2	900
3	300
4	150
5	1800
6	400
7	700
8	160
9	140
10	1400

14. Get Customers Who Placed Orders Totalling Over \$1000.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'orders' and 'order_items' tables expanded. The central pane contains the following SQL query:

```

1 select o.customer_id,concat(Firstname," ",lastname) as fullname,sum(total_price) as tot_price
2 from orders o
3 join customer c
4 on o.customer_id=c.customer_id
5 group by customer_id
6 having tot_price>1000;
7

```

The right pane shows the 'Result Grid' with the following data:

customer_id	fullname	tot_price
1	John Doe	1200
5	David Lee	1800
10	Olivia Adams	1400

15. Subquery to Find Products Not in the Cart.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'orders' and 'order_items' tables expanded. The central pane contains the following SQL query:

```

1 SELECT *
2 FROM Products
3 WHERE Product_id NOT IN (SELECT Product_id FROM cart);
4

```

The right pane shows the 'Result Grid' with the following data:

product_id	name	price	stockquantity	description
2	Latest smartphone	600	15	Smartphone
8	Microwave Oven	80	15	Countertop microwave

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

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'orders' and 'order_items' tables expanded. The central pane contains the following SQL query:

```

1 SELECT *
2 FROM Customer
3 WHERE customer_id NOT IN (SELECT Customer_id FROM Orders);
4

```

The right pane shows the 'Result Grid' with the following data:

customer_id	Firstname	lastname	Email	address
NAVA	NAVA	NAVA	NAVA	NAVA

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

Navigator: SCHEMAS

Filter objects

- quantity
- itemamount
- Indexes
- Foreign Keys
- Triggers
- orders
- products
 - Columns
 - product_id
 - name
 - price
 - stockquantity
 - description
 - Indexes

Administration Schemas

Information

Table: products

Columns:

- product_id int PK
- name varchar(255)
- price int
- stockquantity int
- description text

SQL File 5: order_items

```

1 SELECT p.Product_id,p.name,p.price,
2 (
3   SELECT SUM(o1.Quantity * o1.itemamount)
4   FROM order_items o1
5   WHERE o1.Product_id = p.product_id
6 ) / (
7   SELECT SUM(total_price)
8   FROM orders
9 ) * 100 AS PercentageOfTotalRevenue
10 FROM
11 Products p;
12

```

Result Grid

Product_id	name	price	PercentageOfTotalRevenue
1	Laptop	800	55.9441
2	Latest smartphone	600	109.0909
3	Tablet	300	4.1958
4	Headphones	150	33.5664
5	TV	900	50.5497
6	Coffee Maker	50	0.6993
7	Refrigerator	800	100.0000
8	Microwave Oven	80	100.0000
9	Blender	70	8.8112
10	Vacuum Cleaner	120	6.7133

18. Subquery to Find Products with Low Stock.

Navigator: SCHEMAS

Filter objects

- quantity
- itemamount
- Indexes
- Foreign Keys
- Triggers
- orders
- products
 - Columns
 - product_id
 - name
 - price
 - stockquantity
 - description
 - Indexes

Administration Schemas

Information

Table: products

Columns:

- product_id int PK
- name varchar(255)
- price int
- stockquantity int
- description text

SQL File 5: order_items

```

1 SELECT *
2 FROM Products
3 WHERE Stockquantity < (SELECT AVG(stockquantity) FROM Products);
4

```

Result Grid

product_id	name	price	stockquantity	description
1	Laptop	800	10	High-performance laptop
2	Latest smartphone	600	15	Smartphone
5	TV	900	5	4k Smart TV
7	Refrigerator	800	10	Energy-efficient
8	Microwave Oven	80	15	Countertop microwave
10	Vacuum Cleaner	120	10	Bagless vacuum cleaner

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

Navigator: SCHEMAS

Filter objects

- quantity
- itemamount
- Indexes
- Foreign Keys
- Triggers
- orders
- products
 - Columns
 - product_id
 - name
 - price
 - stockquantity
 - description
 - Indexes

Administration Schemas

Information

Table: products

Columns:

- product_id int PK
- name varchar(255)
- price int

SQL File 5: order_items

```

1 SELECT *
2 FROM customer
3 WHERE
4   Customer_id IN (
5     SELECT customer_id
6     FROM orders
7     WHERE total_price > (SELECT AVG(total_price) FROM Orders)
8   );
9

```

Result Grid

customer_id	Firstname	lastname	Email	address
1	John	Doe	john.doe@example.com	123 Main St, City
2	Jane	Smith	jane.smith@example.com	456 Elm St Town
5	David	Lee	david@example.com	234 Cedar St District
10	Olivia	adams	olivia@example.com	765 Fir St Territory