

Basic SQL Training Questions

1. Create Database e_commerce.

Query:

```
create database e_commerce;
```

```
use e_commerce;
```

The screenshot displays a SQL IDE interface. On the left, the 'Navigator' pane shows a tree view of 'SCHEMAS' with 'e_commerce' expanded, listing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. Below this, the 'Administration' and 'Schemas' tabs are visible, with 'Schemas' selected. The 'Information' pane shows 'No object selected'. The 'Query 1' pane on the right contains two SQL statements: 'CREATE DATABASE e_commerce;' and 'USE e_commerce;'. The 'Output' pane at the bottom right shows the execution results in a table format.

#	Time	Action	Message
1	23:32:38	CREATE DATABASE e_commerce	1 row(s) affected
2	23:32:38	USE e_commerce	0 row(s) affected

Query Completed

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)

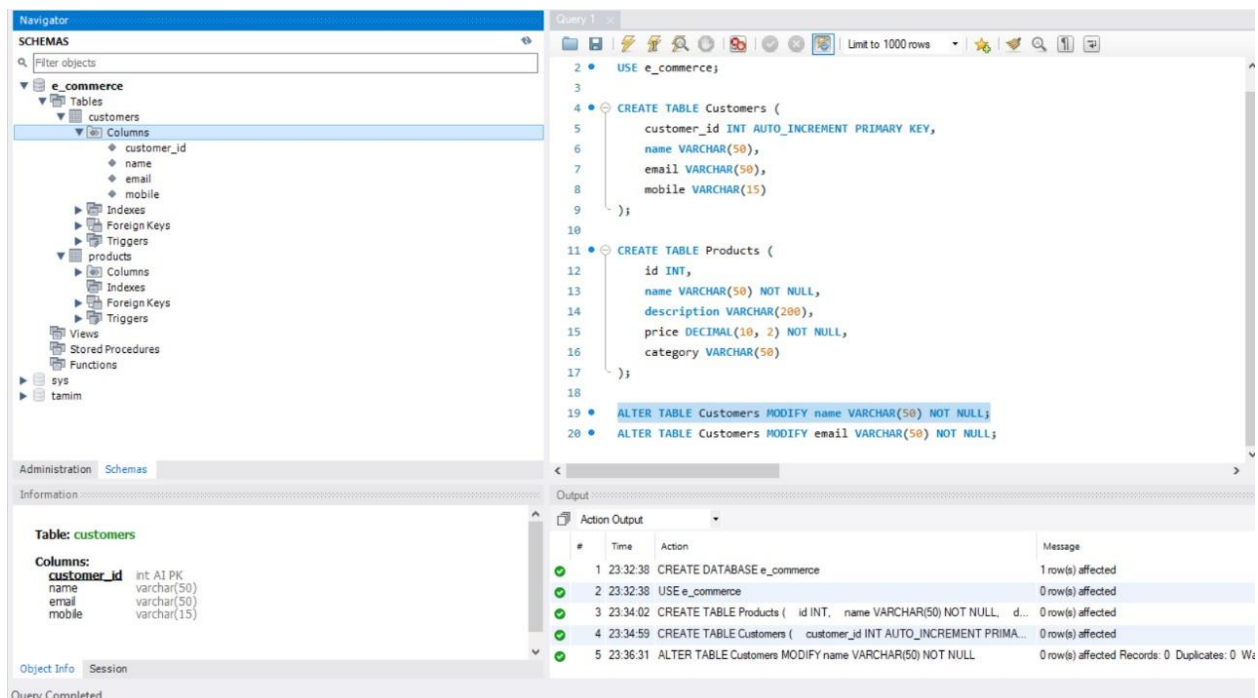
Products:

- 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):

- Add not null on name and email in the Customers table
- Add unique key on email in the Customers table
- Add column age in the Customers table
- Change column name from id to product_id in the Products table;
- Add primary key and auto increment on product_id in the Products table
- Change datatype of description from varchar to text in the Products table

Query:

- Add NOT NULL on name and email in Customers
 ALTER TABLE Customers
 MODIFY name VARCHAR(50) NOT NULL; ALTER TABLE Customers
 MODIFY email VARCHAR(50) NOT NULL;
- Add UNIQUE key on email in Customers
 ALTER TABLE Customers ADD
 CONSTRAINT unique_email UNIQUE (email);

c. Add column age in Customers ALTER TABLE Customers ADD COLUMN age INT;

d. Change column name from id to product_id in Products ALTER TABLE Products CHANGE COLUMN id product_id INT;

e. Add PRIMARY KEY and AUTO_INCREMENT on product_id in Products ALTER TABLE Products MODIFY product_id INT AUTO_INCREMENT PRIMARY KEY;

f. Change datatype of description from VARCHAR to TEXT in Products ALTER TABLE Products MODIFY description TEXT;

The screenshot displays a database management interface with a Navigator pane on the left and a Query Editor on the right. The Navigator shows a schema named 'e_commerce' with tables 'customers' and 'products'. The Query Editor contains the following SQL statements:

```
12 id INT,  
13 name VARCHAR(50) NOT NULL,  
14 description VARCHAR(200),  
15 price DECIMAL(10, 2) NOT NULL,  
16 category VARCHAR(50)  
17 },  
18  
19 ALTER TABLE Customers MODIFY name VARCHAR(50) NOT NULL;  
20 ALTER TABLE Customers MODIFY email VARCHAR(50) NOT NULL;  
21  
22 ALTER TABLE Customers ADD CONSTRAINT unique_email UNIQUE (email);  
23  
24 ALTER TABLE Customers ADD COLUMN age INT;  
25  
26 ALTER TABLE Products CHANGE COLUMN id product_id INT;  
27  
28 ALTER TABLE Products MODIFY product_id INT AUTO_INCREMENT PRIMARY KEY;  
29  
30 ALTER TABLE Products MODIFY description TEXT;
```

The Output pane at the bottom shows the execution results of these queries:

#	Time	Action	Message
9	23:42:26	ALTER TABLE Customers ADD CONSTRAINT unique_email UNIQUE (email)	0 row(s) affected Records: 0 Duplicates: 0 Warnin
10	23:42:33	ALTER TABLE Customers ADD COLUMN age INT	0 row(s) affected Records: 0 Duplicates: 0 Warnin
11	23:42:38	ALTER TABLE Products CHANGE COLUMN id product_id INT	0 row(s) affected Records: 0 Duplicates: 0 Warnin
12	23:42:43	ALTER TABLE Products MODIFY product_id INT AUTO_INCREMENT PRIM...	0 row(s) affected Records: 0 Duplicates: 0 Warnin
13	23:42:58	ALTER TABLE Products MODIFY description TEXT	0 row(s) affected Records: 0 Duplicates: 0 Warnin

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)

);

The screenshot displays a database management interface with a 'Navigator' pane on the left showing a schema named 'e_commerce' containing tables 'customers', 'order', and 'products'. The main area shows a 'Query 1' editor with the following SQL code:

```

24 ALTER TABLE Customers ADD COLUMN age INT;
25
26 ALTER TABLE Products CHANGE COLUMN id product_id INT;
27
28 ALTER TABLE Products MODIFY product_id INT AUTO_INCREMENT PRIMARY KEY;
29
30 ALTER TABLE Products MODIFY description TEXT;
31
32 CREATE TABLE `Order` (
33   order_id INT AUTO_INCREMENT PRIMARY KEY,
34   customer_id INT,
35   product_id INT,
36   quantity INT NOT NULL,
37   order_date DATE NOT NULL,
38   status ENUM('Pending', 'Success', 'Cancel'),
39   payment_method ENUM('Credit', 'Debit', 'UPI'),
40   total_amount DECIMAL(10, 2) NOT NULL,
41   FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
42 );

```

The 'Output' pane at the bottom shows the execution results of four queries:

#	Time	Action	Message
10	23:42:33	ALTER TABLE Customers ADD COLUMN age INT	0 row(s) affected Records: 0 Duplicates: 0 Wa
11	23:42:38	ALTER TABLE Products CHANGE COLUMN id product_id INT	0 row(s) affected Records: 0 Duplicates: 0 Wa
12	23:42:43	ALTER TABLE Products MODIFY product_id INT AUTO_INCREMENT PRIM...	0 row(s) affected Records: 0 Duplicates: 0 Wa
13	23:42:58	ALTER TABLE Products MODIFY description TEXT	0 row(s) affected Records: 0 Duplicates: 0 Wa
14	23:46:12	CREATE TABLE `Order` (order_id INT AUTO_INCREMENT PRIMARY KE...	0 row(s) affected

The status bar at the bottom indicates 'Query Completed'.

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:

- a. Change table name Order -> Orders

```
ALTER TABLE Order RENAME TO Orders;
```

- b. Set default value 'Pending' in status

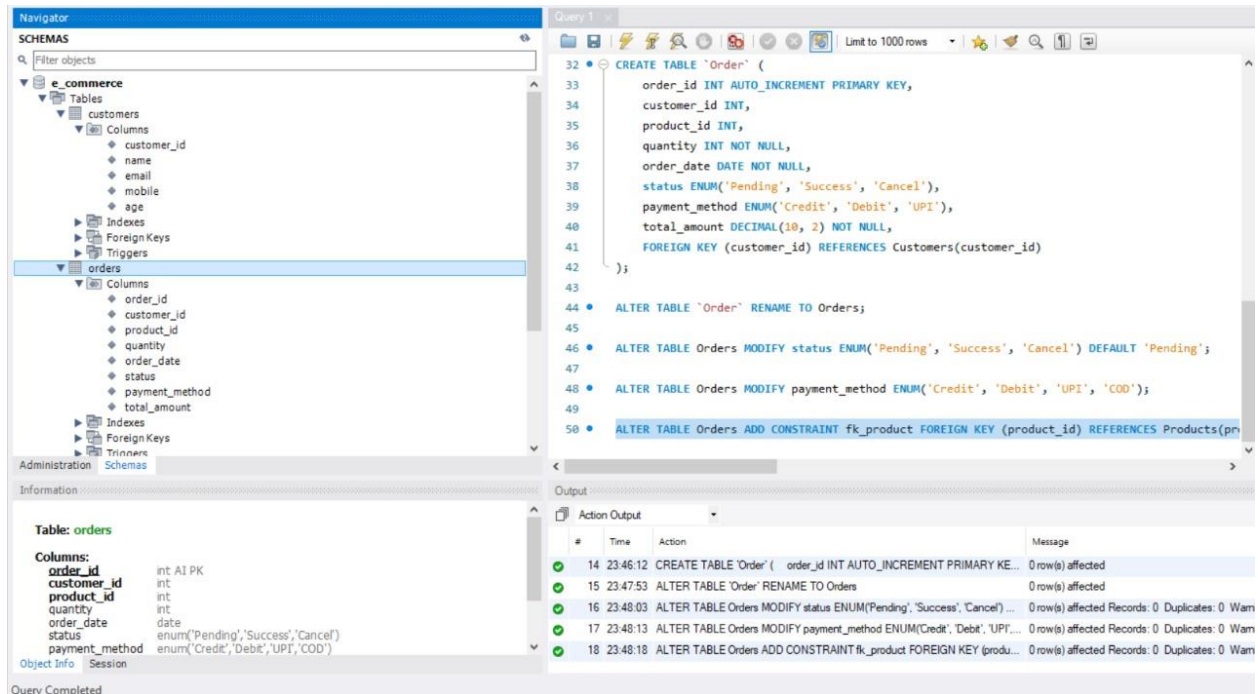
```
ALTER TABLE Orders MODIFY status ENUM('Pending', 'Success', 'Cancel')  
DEFAULT 'Pending';
```

- c. Modify payment_method

```
ENUM to add 'COD' ALTER TABLE Orders MODIFY payment_method  
ENUM('Credit', 'Debit', 'UPI', 'COD');
```

- d. Make product_id as foreign key

```
ALTER TABLE Orders ADD CONSTRAINT fk_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

('Tamim kothari', 'tamimkothari@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);

The screenshot displays a database management interface. On the left, a 'Navigator' pane shows the 'e_commerce' schema with tables 'customers' and 'orders'. The 'customers' table has columns: customer_id, name, email, mobile, and age. The 'orders' table has columns: order_id, customer_id, product_id, quantity, order_date, status, payment_method, and total_amount. The main pane shows a SQL query (Query 1) with 20 rows of data, each representing a customer with their name, email, mobile number, and age. The query execution log at the bottom shows the following actions:

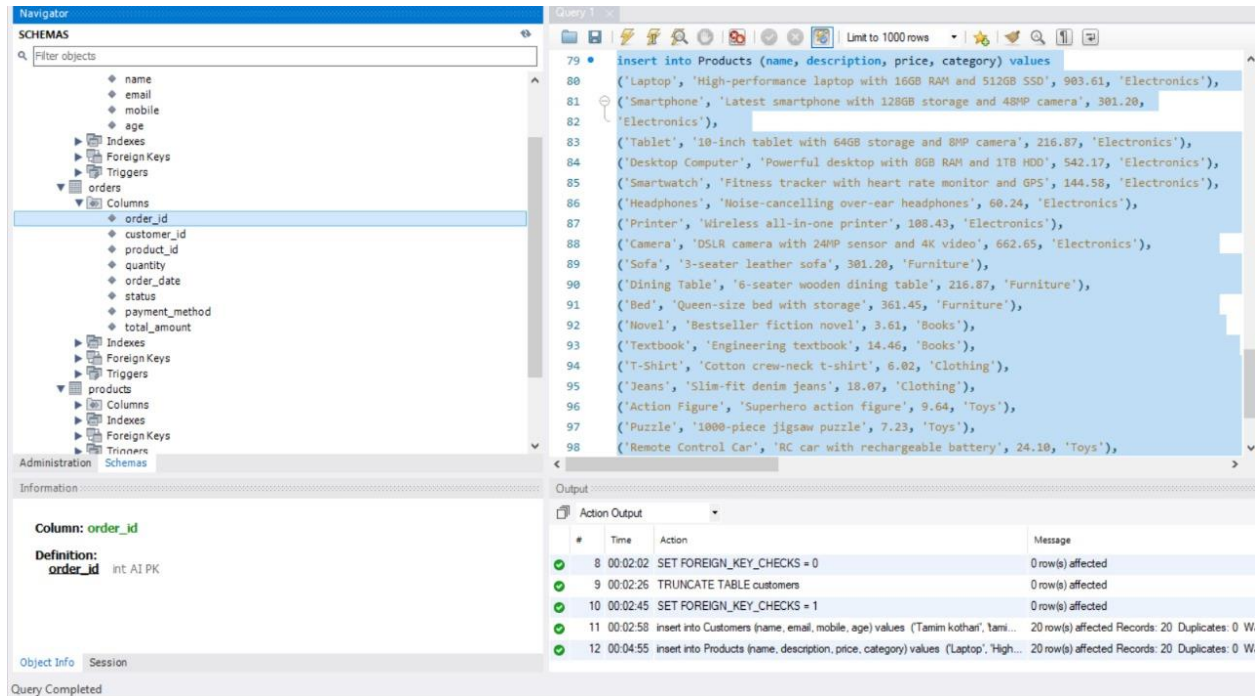
#	Time	Action	Message
7	00:00:32	TRUNCATE TABLE customers	Error Code: 1701. Cannot truncate a table referen
8	00:02:02	SET FOREIGN_KEY_CHECKS = 0	0 row(s) affected
9	00:02:26	TRUNCATE TABLE customers	0 row(s) affected
10	00:02:45	SET FOREIGN_KEY_CHECKS = 1	0 row(s) affected
11	00:02:58	insert into Customers (name, email, mobile, age) values ('Tamim kothari', 'tami...	20 row(s) affected Records: 20 Duplicates: 0 W

Query:

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);

Result Grid		Filter Rows:		Edit:		Export/Import:		Wrap Cell Content:	
	order_id	customer_id	product_id	quantity	order_date	status	payment_method	total_amount	
	1	1	1	1	2025-10-01	Success	Credit	903.61	
	2	2	2	2	2025-10-02	Pending	Debit	602.40	
	3	3	3	1	2025-10-03	Success	UPI	216.87	
	4	3	6	1	2025-10-06	Pending	UPI	60.24	
	5	4	7	1	2025-10-07	Success	Credit	108.43	
	6	5	10	1	2025-10-09	Pending	UPI	301.20	
	7	5	10	1	2025-10-10	Success	Credit	216.87	
	8	6	4	1	2025-10-04	Cancel	Credit	542.17	
	9	6	12	1	2025-10-11	Success	Debit	361.45	
	10	9	12	2	2025-10-12	Success	UPI	28.92	
	11	9	13	2	2025-10-13	Success	Debit	7.22	
	12	10	13	3	2025-10-14	Success	UPI	18.06	
	13	11	15	1	2025-10-15	Cancel	COD	18.07	
	14	12	16	1	2025-10-16	Pending	COD	9.64	
	15	12	17	1	2025-10-17	Success	Credit	7.23	
	16	12	17	2	2025-10-18	Success	COD	48.20	
	17	16	16	5	2025-10-19	Pending	UPI	30.10	
	18	19	20	2	2025-10-20	Success	Debit	4.82	
	19	19	5	2	2025-10-05	Cancel	Debit	289.16	
	20	18	8	1	2025-10-08	Success	UPI	662.65	
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	

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;

The screenshot shows a database management tool interface. On the left, a 'SCHEMAS' pane displays a tree view of database objects, including 'orders' and 'products'. The 'products' table is selected, showing its columns: order_id, customer_id, product_id, quantity, order_date, status, payment_method, and total_amount. The main query editor displays the following SQL query:

```
SELECT category, COUNT(*) AS product_count FROM Products GROUP BY category;
```

The 'Result Grid' shows the query results:

category	product_count
Electronics	8
Furniture	3
Books	2
Clothing	2
Toys	3
Grocery	2

The 'Output' pane shows the query execution details:

#	Time	Action	Message
1	00:08:15	SELECT category, COUNT(*) AS product_count FROM Products GROUP ...	6 row(s) returned

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%';

Query 1

```

106 • SELECT * FROM Products
107 WHERE category = 'Electronics'
108 AND price BETWEEN 50 AND 500
109 AND name LIKE '%a%';
110
111
112 • SELECT * FROM Products

```

Result Grid

product_id	name	description	price	category
2	Smartphone	Latest smartphone with 128GB storage and 48...	301.20	Electronics
3	Tablet	10-inch tablet with 64GB storage and 8MP camera	216.87	Electronics
5	Smartwatch	Fitness tracker with heart rate monitor and GPS	144.58	Electronics
6	Headphones	Noise-cancelling over-ear headphones	60.24	Electronics

Products 2

Output

#	Time	Action	Message
1	00:08:15	SELECT category, COUNT(*) AS product_count FROM Products GROUP ...	6 row(s) returned
2	00:08:55	SELECT * FROM Products WHERE category = 'Electronics' AND price B...	4 row(s) returned

Query Completed

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;

Navigator

SCHEMAS

Filter objects

- name
- email
- mobile
- age
- Indexes
- Foreign Keys
- Triggers
- orders
 - Columns
 - order_id
 - customer_id
 - product_id
 - quantity
 - order_date
 - status
 - payment_method
 - total_amount
 - Indexes
 - Foreign Keys
 - Triggers
- products
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers

Administration Schemas

Information

Column: **order_id**

Definition:

order_id int AI PK

Object Info Session

Query Completed

Query 1

Execute the statement under the keyboard cursor

```

112 SELECT * FROM Products
113 WHERE category = 'Electronics'
114 ORDER BY price DESC
115 LIMIT 5 OFFSET 2;
116

```

Result Grid

product_id	name	description	price	category
4	Desktop Computer	Powerful desktop with 8GB RAM and 1TB HDD	542.17	Electronics
2	Smartphone	Latest smartphone with 128GB storage and 48...	301.20	Electronics
3	Tablet	10-inch tablet with 64GB storage and 8MP camera	216.87	Electronics
5	Smartwatch	Fitness tracker with heart rate monitor and GPS	144.58	Electronics
7	Printer	Wireless all-in-one printer	108.43	Electronics

Products 3 x

Output

Action Output

#	Time	Action	Message
1	00:08:15	SELECT category, COUNT(*) AS product_count FROM Products GROUP ...	6 row(s) returned
2	00:08:55	SELECT * FROM Products WHERE category = 'Electronics' AND price B...	4 row(s) returned
3	00:09:47	SELECT * FROM Products WHERE category = 'Electronics' ORDER BY p...	5 row(s) returned

d. Retrieve customers who have not placed any orders.

Query:

select * from Customers

where customer_id not in (select distinct customer_id from Orders);

Navigator

SCHEMAS

Filter objects

- name
- email
- mobile
- age
- Indexes
- Foreign Keys
- Triggers
- orders
 - Columns
 - order_id
 - customer_id
 - product_id
 - quantity
 - order_date
 - status
 - payment_method
 - total_amount
 - Indexes
 - Foreign Keys
 - Triggers
- products
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers

Administration Schemas

Information

Column: **order_id**

Definition:

order_id int AI PK

Object Info Session

Query Completed

Query 1

```

115 LIMIT 5 OFFSET 2;
116
117
118 SELECT * FROM Customers
119 WHERE customer_id NOT IN (SELECT DISTINCT customer_id FROM Orders);
120
121

```

Result Grid

customer_id	name	email	mobile	age
1	Tamim kothari	tamimkothari@example.com	9876543210	21
2	Priya Patel	priya.patel@example.com	8765432109	25
3	Amit Rathore	amit.rathore@example.com	7654321098	30
4	Neha Gupta	neha.gupta@example.com	6543210987	22
5	Vikram Yadav	vikram.yadav@example.com	5432109876	35
6	Anjali Desai	anjali.desai@example.com	4321098765	27
7	Rajesh Kumar	rajesh.kumar@example.com	3210987654	40
8	Rudransh Reddy	rudransh.reddy@example.com	2109876543	29
9	Arun Jain	arun.jain@example.com	1098765432	33
10	Pooja Choudhary	pooja.choudhary@example.com	9988776655	26
11	Sanjay Lela	sanjay.lela@example.com	8877665544	31

Customers 6 x

Output

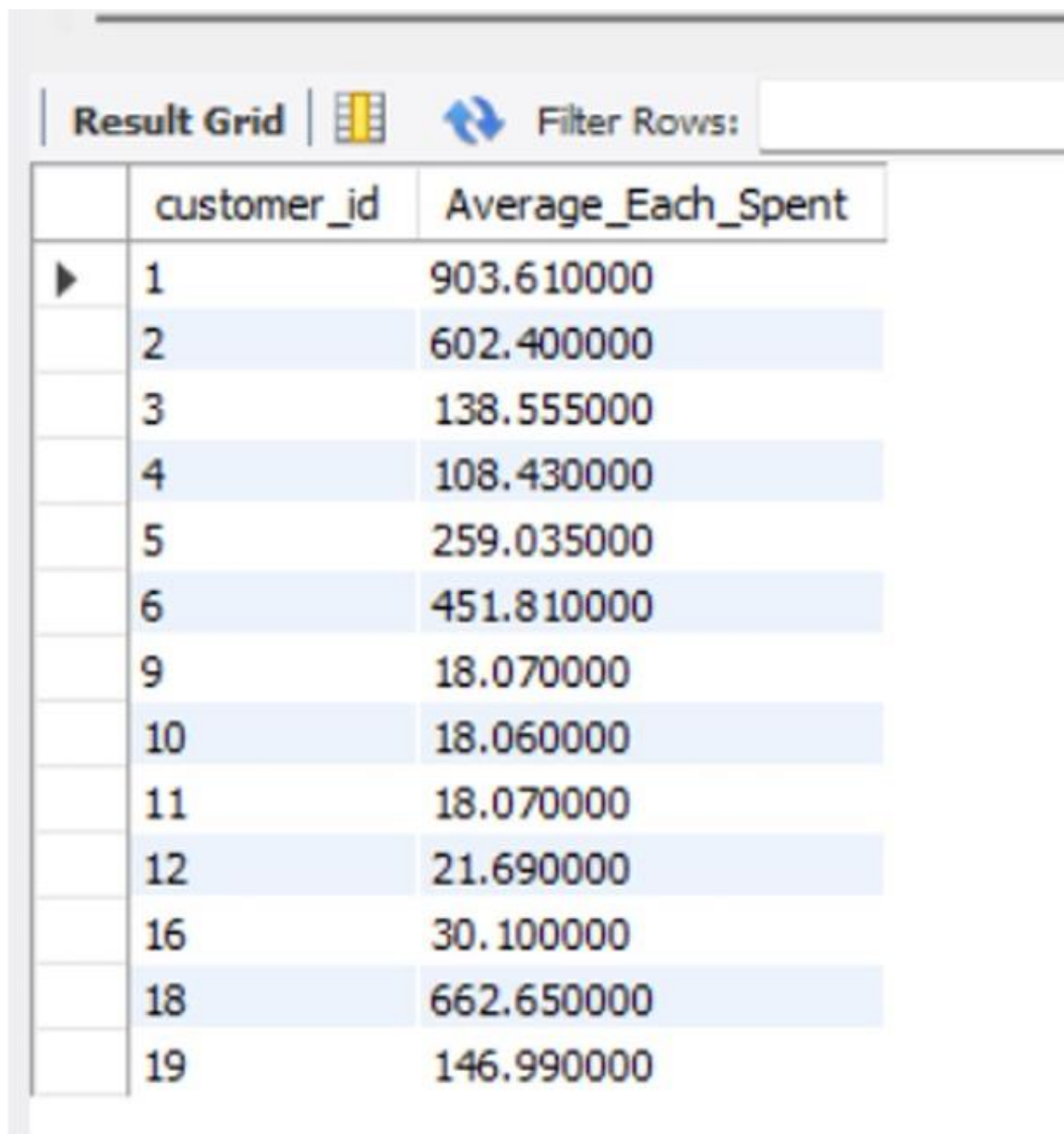
Action Output

#	Time	Action	Message
1	00:08:15	SELECT category, COUNT(*) AS product_count FROM Products GROUP ...	6 row(s) returned
2	00:08:55	SELECT * FROM Products WHERE category = 'Electronics' AND price B...	4 row(s) returned
3	00:09:47	SELECT * FROM Products WHERE category = 'Electronics' ORDER BY p...	5 row(s) returned
4	00:11:46	SELECT * FROM Products WHERE product_id NOT IN (SELECT DISTIN...	20 row(s) returned
5	00:13:03	SELECT customer_id, AVG(total_amount) AS avg_spent FROM Orders G...	0 row(s) returned
6	00:14:33	SELECT * FROM Customers WHERE customer_id NOT IN (SELECT DIST...	20 row(s) returned

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;
```



The screenshot shows a database query result grid. At the top, there is a toolbar with a 'Result Grid' tab, a grid icon, a refresh icon, and a 'Filter Rows:' input field. Below the toolbar is a table with two columns: 'customer_id' and 'Average_Each_Spent'. The table contains 15 rows of data, with alternating light blue and white background colors for the rows. The first row has a small black triangle icon in the first column.

	customer_id	Average_Each_Spent
▶	1	903.610000
	2	602.400000
	3	138.555000
	4	108.430000
	5	259.035000
	6	451.810000
	9	18.070000
	10	18.060000
	11	18.070000
	12	21.690000
	16	30.100000
	18	662.650000
	19	146.990000

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);
```

The screenshot shows a database management tool interface. On the left is a 'Navigator' pane showing a tree view of database objects including 'SCHEMAS', 'orders', and 'products'. The 'products' table is selected. In the center is a 'Query' editor with the following SQL query:

```
121
122 • SELECT customer_id, AVG(total_amount) AS avg_spent
123 FROM Orders
124 GROUP BY customer_id;
125
126
127 • SELECT * FROM Products
```

Below the query editor is a 'Result Grid' showing a list of products with columns: product_id, name, description, price, and category. The data is as follows:

product_id	name	description	price	category
5	Smartwatch	Fitness tracker with heart rate monitor and GPS	144.58	Electronics
6	Headphones	Noise-cancelling over-ear headphones	60.24	Electronics
7	Printer	Wireless all-in-one printer	108.43	Electronics
12	Novel	Bestseller fiction novel	3.61	Books
13	Textbook	Engineering textbook	14.46	Books
14	T-Shirt	Cotton crew-neck t-shirt	6.02	Clothing
15	Jeans	Slim-fit denim jeans	18.07	Clothing
16	Action Figure	Superhero action figure	9.64	Toys
17	Puzzle	1000-piece jigsaw puzzle	7.23	Toys
18	Remote Control Car	RC car with rechargeable battery	24.10	Toys
19	Rice	Basmati rice, 5kg pack	6.02	Grocery

At the bottom, an 'Output' pane shows a log of database actions with timestamps and messages, including the execution of the query shown in the editor.

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

Query:

```
select customer_id,
```

```
SUM(quantity) as Total_Quantity
```

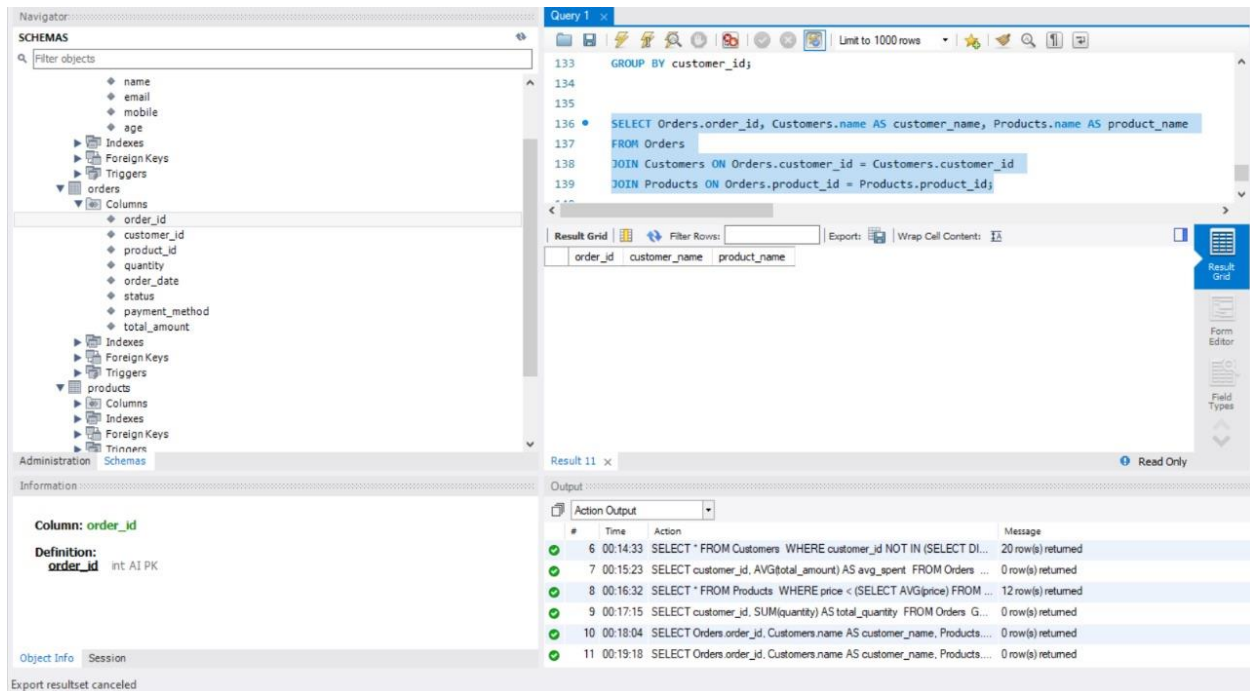
```
from Orders
```

```
group by customer_id;
```


Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	customer_id	Total_Quantity			
▶	1	1			
	2	2			
	3	2			
	4	1			
	5	2			
	6	2			
	9	4			
	10	3			
	11	1			
	12	4			
	16	5			
	18	1			
	19	4			

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);

SCHEMAS

Filter objects

- name
- email
- mobile
- age
- Indexes
- Foreign Keys
- Triggers
- orders
 - Columns
 - order_id
 - customer_id
 - product_id
 - quantity
 - order_date
 - status
 - payment_method
 - total_amount
 - Indexes
 - Foreign Keys
 - Triggers
- products
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers

Administration

Schemas

Column: **order_id**

Definition:
order_id int AI PK

Object Info

Session

Query Completed

Query Editor

Limit to 1000 rows

```
138 JOIN Customers ON Orders.customer_id = Customers.customer_id
139 JOIN Products ON Orders.product_id = Products.product_id;
140
141
142 SELECT * FROM Products
143 WHERE product_id NOT IN (SELECT DISTINCT product_id FROM Orders);
144
```

Result Grid

Filter Rows:

Export/Imports

Wrap Cell Contents

product_id	name	description	price	category
1	Laptop	High-performance laptop with 16GB RAM and 5...	903.61	Electronics
2	Smartphone	Latest smartphone with 128GB storage and 48...	301.20	Electronics
3	Tablet	10-inch tablet with 64GB storage and 8MP camera	216.87	Electronics
4	Desktop Computer	Powerful desktop with 8GB RAM and 1TB HDD	542.17	Electronics
5	Smartwatch	Fitness tracker with heart rate monitor and GPS	144.58	Electronics
6	Headphones	Noise-cancelling over-ear headphones	60.24	Electronics
7	Printer	Wireless all-in-one printer	108.43	Electronics
8	Camera	DSLR camera with 24MP sensor and 4K video	662.65	Electronics
9	Sofa	3-seater leather sofa	301.20	Furniture
10	Dining Table	6-seater wooden dining table	216.87	Furniture
11	Bed	Queen-size bed with storage	361.45	Furniture

Products 12 x

Apply

Revert

Output

Action Output

#	Time	Action	Message
7	00:15:23	SELECT customer_id, AVG(total_amount) AS avg_spent FROM Orders ...	0 row(s) returned
8	00:16:32	SELECT * FROM Products WHERE price < (SELECT AVG(price) FROM ...	12 row(s) returned
9	00:17:15	SELECT customer_id, SUM(quantity) AS total_quantity FROM Orders G...	0 row(s) returned
10	00:18:04	SELECT Orders.order_id, Customers.name AS customer_name, Products...	0 row(s) returned
11	00:19:18	SELECT Orders.order_id, Customers.name AS customer_name, Products...	0 row(s) returned
12	00:20:15	SELECT * FROM Products WHERE product_id NOT IN (SELECT DISTI...	20 row(s) returned