

MySQL Task

Create Database and Tables

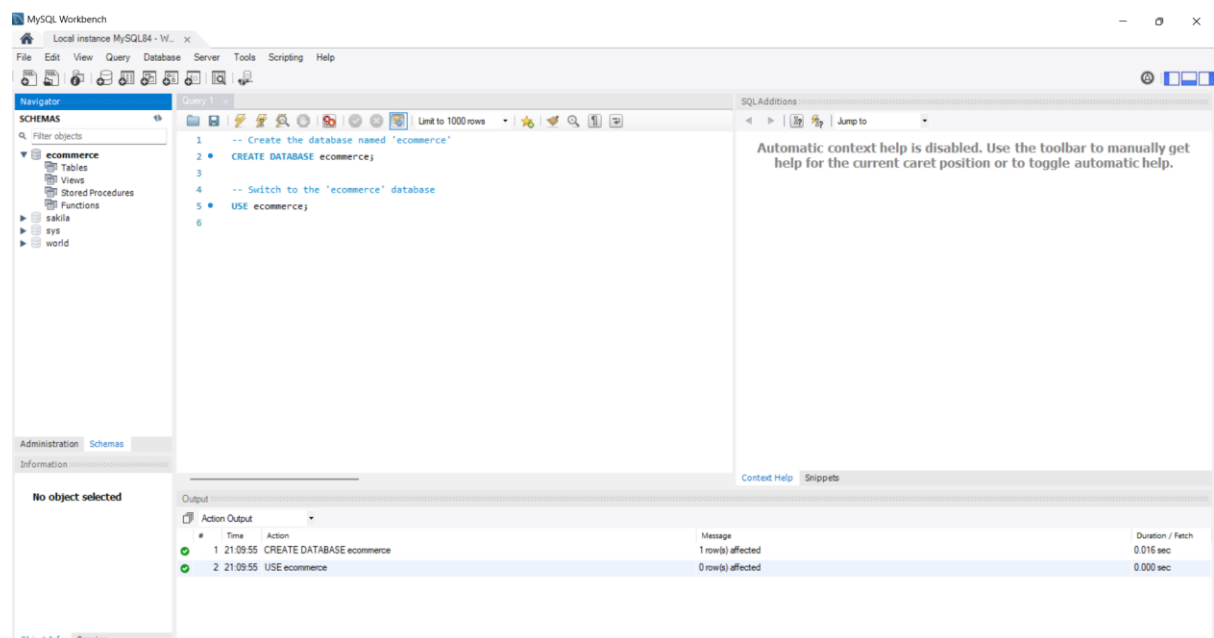
-- Create the database named 'ecommerce'

```
CREATE DATABASE ecommerce;
```

-- Switch to the 'ecommerce' database

```
USE ecommerce;
```

Output:



-- Create the 'customers' table to store customer details

```
CREATE TABLE customers (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each customer
```

```
    name VARCHAR(255) NOT NULL, -- Name of the customer
```

```
    email VARCHAR(255) NOT NULL UNIQUE, -- Email address of the customer
```

```
    address TEXT NOT NULL -- Address of the customer
```

```
);
```

-- Create the 'products' table to store product details

```
CREATE TABLE products (  
    id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each product  
    name VARCHAR(255) NOT NULL, -- Name of the product  
    price DECIMAL(10, 2) NOT NULL, -- Price of the product  
    description TEXT NOT NULL -- Description of the product  
);
```

-- Create the 'orders' table to store order details

```
CREATE TABLE orders (  
    id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each order  
    customer_id INT NOT NULL, -- ID of the customer who placed the order  
    order_date DATE NOT NULL, -- Date the order was placed  
    total_amount DECIMAL(10, 2) NOT NULL, -- Total amount for the order  
    FOREIGN KEY (customer_id) REFERENCES customers(id) -- Link to the 'customers' table  
);
```

Output:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree with 'ecommerce' selected. The main editor shows the SQL script for creating the 'orders' table. The bottom 'Output' panel shows the execution results of the commands.

#	Time	Action	Message	Duration / Fetch
1	21:09:55	CREATE DATABASE ecommerce	1 row(s) affected	0.016 sec
2	21:09:55	USE ecommerce	0 row(s) affected	0.000 sec
3	21:10:58	CREATE TABLE customers (id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each customer...	0 row(s) affected	0.047 sec
4	21:11:17	CREATE TABLE products (id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each product...	0 row(s) affected	0.031 sec
5	21:11:34	- Create the 'orders' table to store order details CREATE TABLE orders (id INT AUTO_INCREMENT PRIMARY...	Error Code: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MySQL serve...	0.000 sec
6	21:11:57	CREATE TABLE orders (id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each order...	0 row(s) affected	0.047 sec

2. Insert Sample Data

-- Insert sample data into the 'customers' table

```
INSERT INTO customers (name, email, address)
```

```
VALUES
```

```
('John Doe', 'john.doe@example.com', '123 Elm Street'),
```

```
('Jane Smith', 'jane.smith@example.com', '456 Oak Avenue'),
```

```
('Alice Brown', 'alice.brown@example.com', '789 Maple Drive');
```

-- Insert sample data into the 'products' table

```
INSERT INTO products (name, price, description)
```

```
VALUES
```

```
('Product A', 25.00, 'Description of Product A'),
```

```
('Product B', 35.00, 'Description of Product B'),
```

```
('Product C', 40.00, 'Description of Product C');
```

-- Insert sample data into the 'orders' table

```
INSERT INTO orders (customer_id, order_date, total_amount)
```

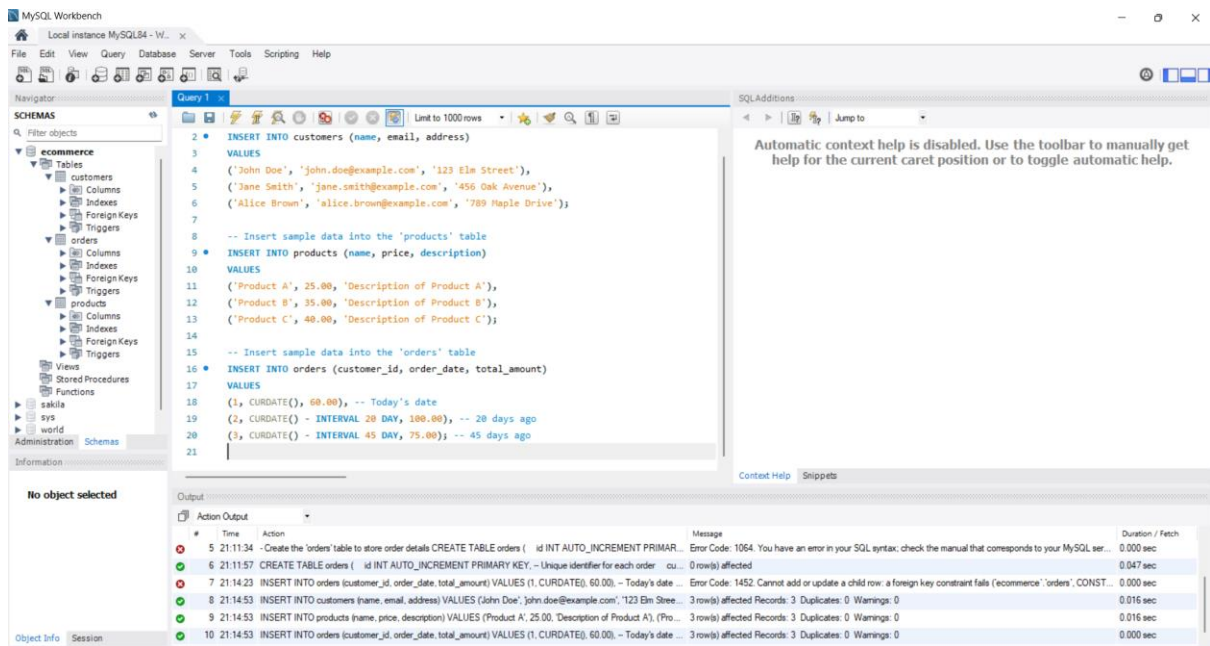
```
VALUES
```

```
(1, CURDATE(), 60.00), -- Today's date
```

```
(2, CURDATE() - INTERVAL 20 DAY, 100.00), -- 20 days ago
```

```
(3, CURDATE() - INTERVAL 45 DAY, 75.00); -- 45 days ago
```

Output:



3. Queries

a. Retrieve all customers who have placed an order in the last 30 days

-- Select customers who have placed an order within the past 30 days

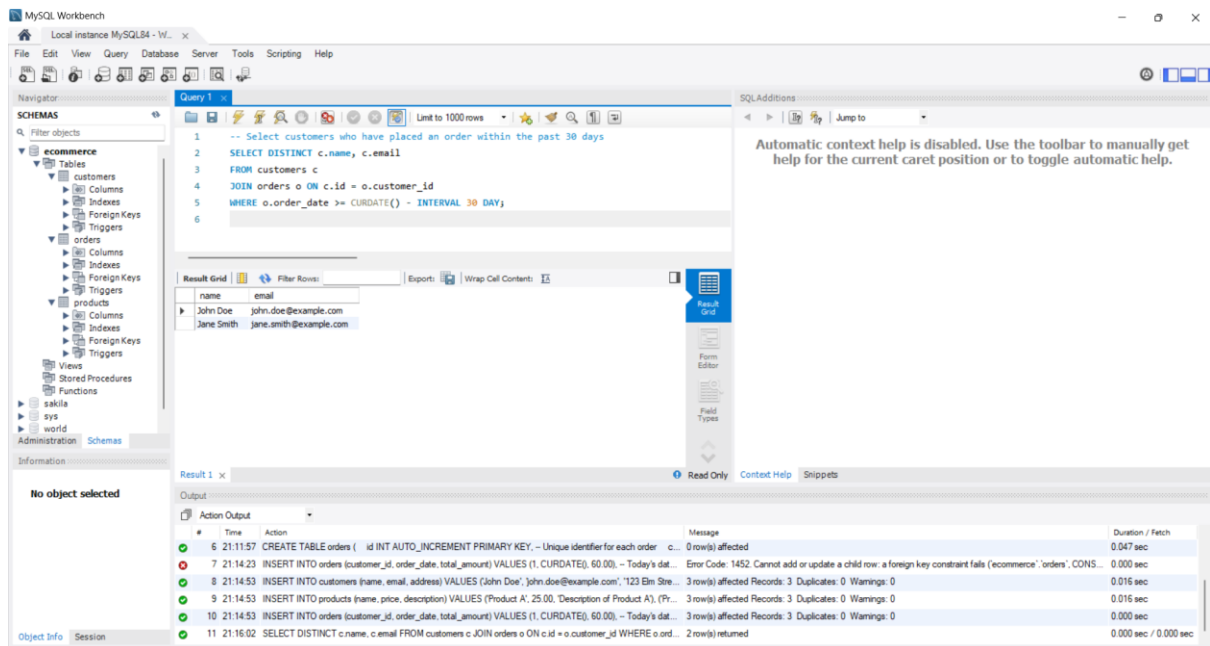
SELECT DISTINCT c.name, c.email

FROM customers c

JOIN orders o ON c.id = o.customer_id

WHERE o.order_date >= CURDATE() - INTERVAL 30 DAY;

Output:



b. Get the total amount of all orders placed by each customer

-- Calculate the total order amount grouped by each customer

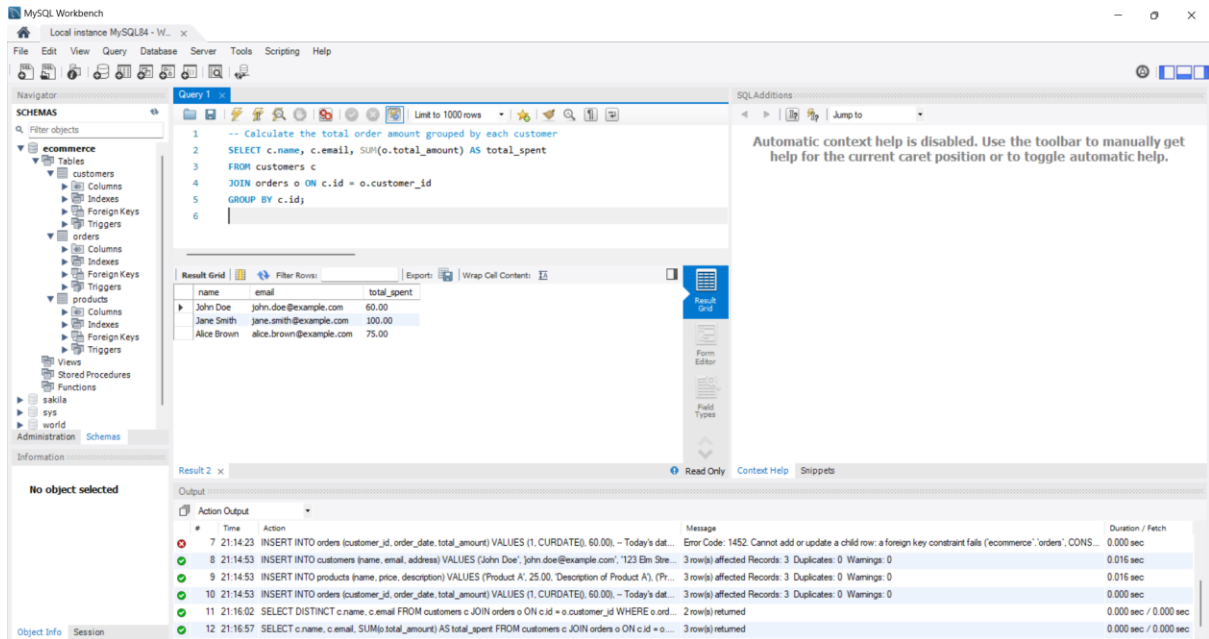
SELECT c.name, c.email, SUM(o.total_amount) AS total_spent

FROM customers c

JOIN orders o ON c.id = o.customer_id

GROUP BY c.id;

Output:



c. Update the price of Product C to 45.00

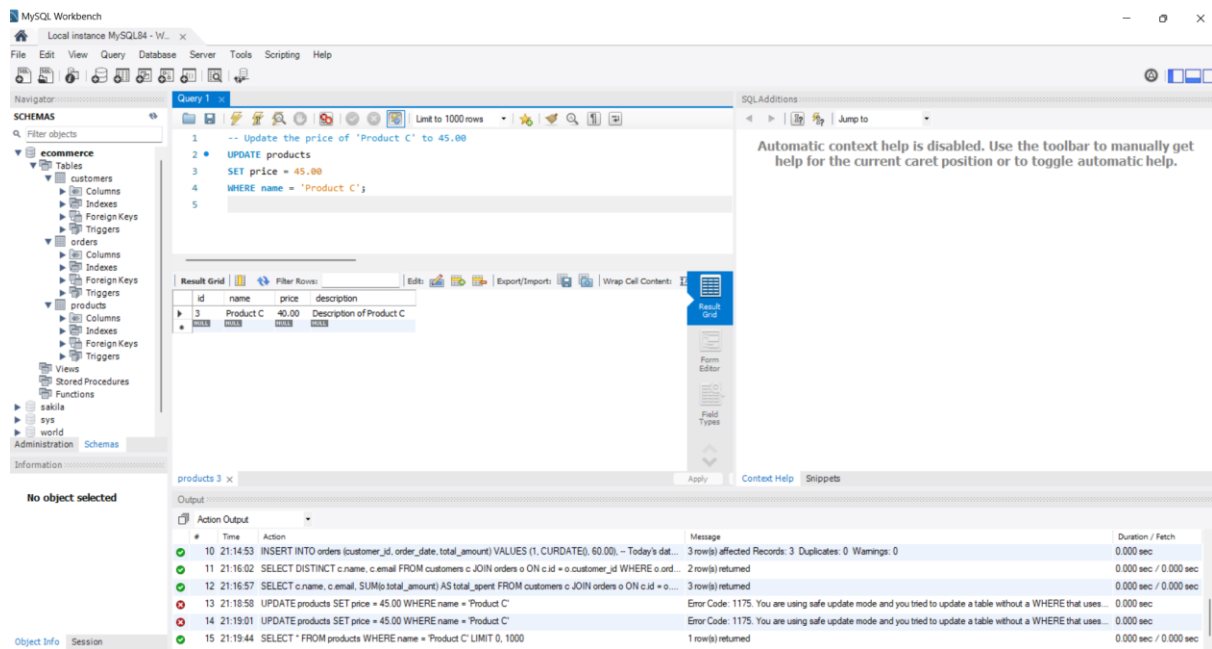
-- Update the price of 'Product C' to 45.00

UPDATE products

SET price = 45.00

WHERE name = 'Product C';

Output:



d. Add a new column discount to the products table

-- Add a new column named 'discount' with a default value of 0.00

ALTER TABLE products

ADD COLUMN discount DECIMAL(5, 2) DEFAULT 0.00;

e. Retrieve the top 3 products with the highest price

-- Fetch the top 3 most expensive products

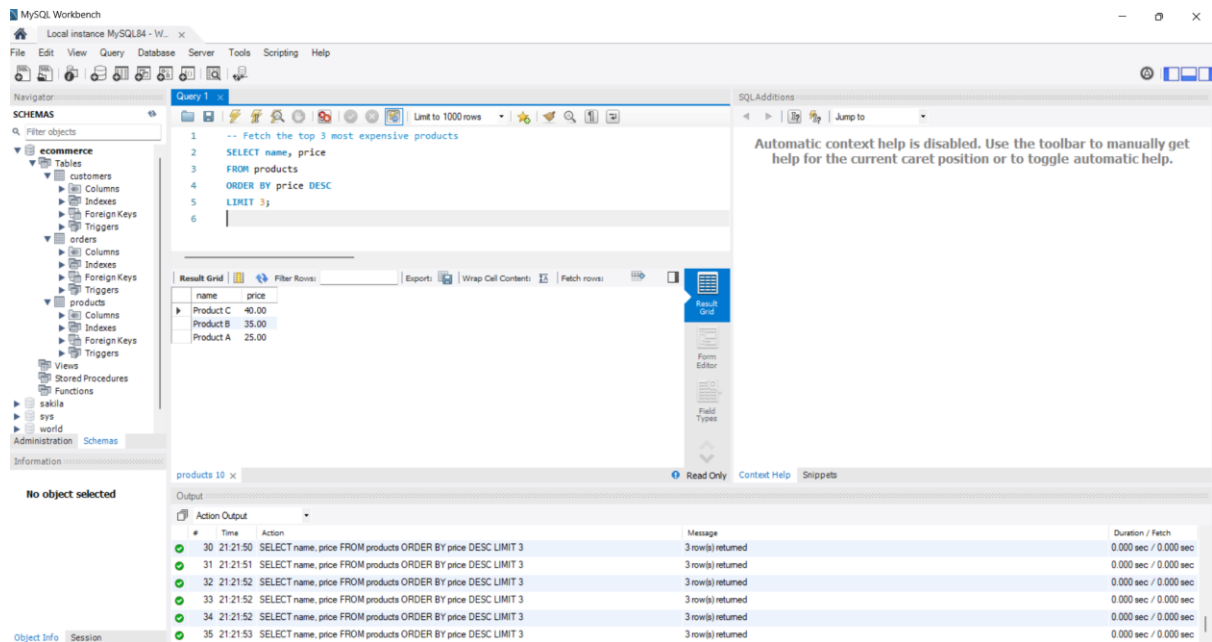
SELECT name, price

FROM products

ORDER BY price DESC

LIMIT 3;

Output:



f. Get the names of customers who have ordered Product A

-- Retrieve names of customers who have ordered 'Product A'

SELECT DISTINCT c.name

FROM customers c

JOIN orders o ON c.id = o.customer_id

JOIN order_items oi ON o.id = oi.order_id

JOIN products p ON oi.product_id = p.id

WHERE p.name = 'Product A';

Output:

The screenshot shows the MySQL Workbench interface. The 'Query' tab is active, displaying a SQL query that selects distinct customer names from the 'customers' table, joined with 'orders' and 'products' tables. The 'Results' tab shows a single row of data: Product C with a price of 40.00 and a description 'Description of Product C'. The 'Output' tab shows the execution log, including a warning message: 'Error Code: 1146. Table 'ecommerce.order_items' doesn't exist'.

```
2 • SELECT DISTINCT c.name
3 FROM customers c
4 JOIN orders o ON c.id = o.customer_id
5 JOIN order_items oi ON o.id = oi.order_id
6 JOIN products p ON oi.product_id = p.id
7 WHERE p.name = 'Product A';
8
```

#	id	name	price	description
1	3	Product C	40.00	Description of Product C

#	Time	Action	Message	Duration / Fetch
32	21:21:52	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
33	21:21:52	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
34	21:21:52	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
35	21:21:53	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
36	21:23:24	SELECT DISTINCT c.name FROM customers c JOIN orders o ON c.id = o.customer_id JOIN order_items oi ON o.id = oi.order_id JOIN products p ON oi.product_id = p.id WHERE p.name = 'Product A'	Error Code: 1146. Table 'ecommerce.order_items' doesn't exist	0.000 sec
37	21:23:40	SELECT * FROM products WHERE name = 'Product C' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

g. Join orders and customers tables to retrieve customer name and order date

-- Get the customer's name and the date of each order

SELECT c.name AS customer_name, o.order_date

FROM orders o

JOIN customers c ON o.customer_id = c.id;

Output:

The screenshot shows the MySQL Workbench interface. The 'Query' tab is active, displaying a SQL query that selects customer names and order dates from the 'customers' and 'orders' tables. The 'Results' tab shows three rows of data: John Doe (2024-11-21), Jane Smith (2024-11-01), and Alice Brown (2024-09-07). The 'Output' tab shows the execution log, including a warning message: 'Error Code: 1146. Table 'ecommerce.order_items' doesn't exist'.

```
1 -- Get the customer's name and the date of each order
2 • SELECT c.name AS customer_name, o.order_date
3 FROM orders o
4 JOIN customers c ON o.customer_id = c.id;
5
```

#	customer_name	order_date
1	John Doe	2024-11-21
2	Jane Smith	2024-11-01
3	Alice Brown	2024-09-07

#	Time	Action	Message	Duration / Fetch
33	21:21:52	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
34	21:21:52	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
35	21:21:53	SELECT name, price FROM products ORDER BY price DESC LIMIT 3	3 row(s) returned	0.000 sec / 0.000 sec
36	21:23:24	SELECT DISTINCT c.name FROM customers c JOIN orders o ON c.id = o.customer_id JOIN order_items oi ON o.id = oi.order_id JOIN products p ON oi.product_id = p.id WHERE p.name = 'Product A'	Error Code: 1146. Table 'ecommerce.order_items' doesn't exist	0.000 sec
37	21:23:40	SELECT * FROM products WHERE name = 'Product C' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
38	21:24:54	SELECT c.name AS customer_name, o.order_date FROM orders o JOIN customers c ON o.customer_id = c.id	3 row(s) returned	0.000 sec / 0.000 sec

h. Retrieve orders with a total amount greater than 150.00

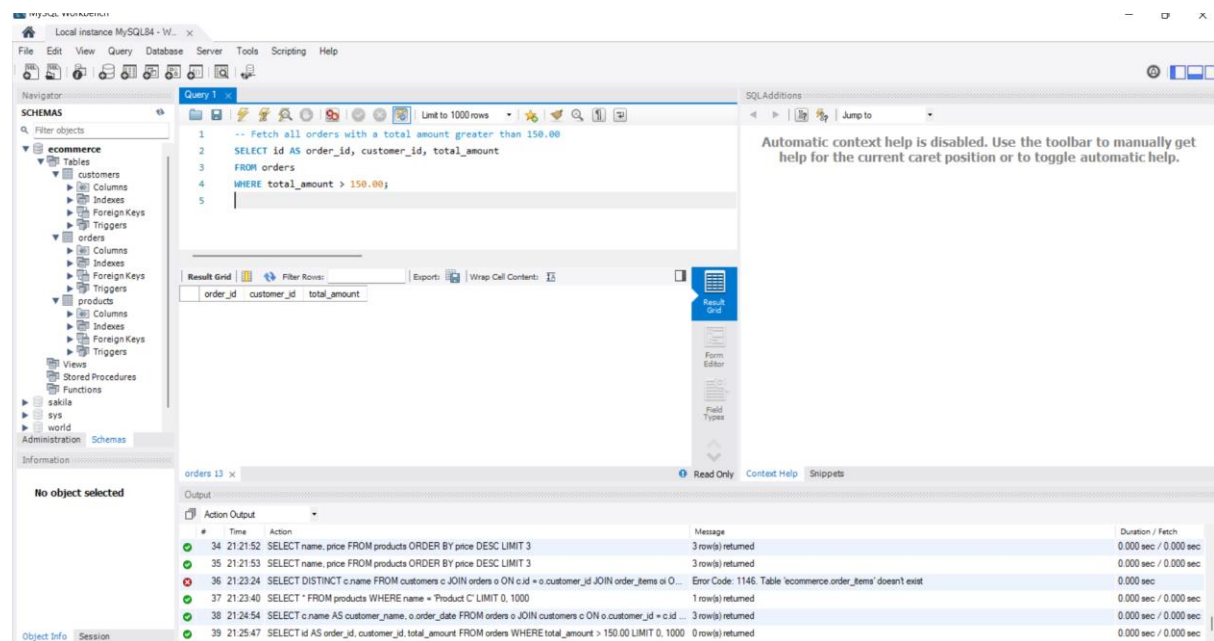
-- Fetch all orders with a total amount greater than 150.00

```
SELECT id AS order_id, customer_id, total_amount
```

```
FROM orders
```

```
WHERE total_amount > 150.00;
```

Output:



i. Normalize the database (add order_items table)

-- Create the 'order_items' table to store details of individual items in an order

```
CREATE TABLE order_items (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each order item
```

```
    order_id INT NOT NULL, -- ID of the order
```

```
    product_id INT NOT NULL, -- ID of the product
```

```
    quantity INT NOT NULL, -- Quantity of the product ordered
```

```
    FOREIGN KEY (order_id) REFERENCES orders(id), -- Link to the 'orders' table
```

```
    FOREIGN KEY (product_id) REFERENCES products(id) -- Link to the 'products' table
```

```
);
```

-- Remove the 'total_amount' column from the 'orders' table to calculate it dynamically

ALTER TABLE orders DROP COLUMN total_amount;

-- Insert sample data into the 'order_items' table

INSERT INTO order_items (order_id, product_id, quantity)

VALUES

(1, 1, 2), -- 2 units of Product A

(2, 2, 3), -- 3 units of Product B

(3, 3, 1); -- 1 unit of Product C

Output:

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' tree with 'ecommerce' selected. The main editor shows a SQL script with the following queries:

```
1 -- Create the 'order_items' table to store details of individual items in an order
2 CREATE TABLE order_items (
3   id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each order item
4   order_id INT NOT NULL, -- ID of the order
5   product_id INT NOT NULL, -- ID of the product
6   quantity INT NOT NULL, -- Quantity of the product ordered
7   FOREIGN KEY (order_id) REFERENCES orders(id), -- Link to the 'orders' table
8   FOREIGN KEY (product_id) REFERENCES products(id) -- Link to the 'products' table
9 );
10
11 -- Remove the 'total_amount' column from the 'orders' table to calculate it dynamically
12 ALTER TABLE orders DROP COLUMN total_amount;
13
14 -- Insert sample data into the 'order_items' table
15 INSERT INTO order_items (order_id, product_id, quantity)
16 VALUES
17 (1, 1, 2), -- 2 units of Product A
18 (2, 2, 3), -- 3 units of Product B
19 (3, 3, 1); -- 1 unit of Product C
20
```

The right sidebar shows the 'SQL Additions' panel with a message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

The bottom panel shows the 'Output' tab with the following results:

#	Time	Action	Message	Duration / Fetch
36	21:23:24	SELECT DISTINCT c.name FROM customers c JOIN orders o ON c.id = o.customer_id JOIN order_items oi ON o.id = oi.order_id	Error Code: 1146. Table 'ecommerce.order_items' doesn't exist	0.000 sec
37	21:23:40	SELECT * FROM products WHERE name = 'Product C' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
38	21:24:54	SELECT c.name AS customer_name, o.order_date FROM orders o JOIN customers c ON o.customer_id = c.id	3 row(s) returned	0.000 sec / 0.000 sec
39	21:25:47	SELECT oi AS order_id, oi AS product_id, oi AS total_amount FROM orders WHERE total_amount > 150.00 LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
40	21:26:38	CREATE TABLE order_items (id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each order item	0 row(s) affected	0.062 sec
41	21:26:38	ALTER TABLE orders DROP COLUMN total_amount	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec

j. Retrieve the average total of all orders

-- Calculate the average total amount for all orders dynamically

SELECT AVG(oi.quantity * p.price) AS average_order_total

FROM orders o

JOIN order_items oi ON o.id = oi.order_id

JOIN products p ON oi.product_id = p.id;

Output:

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' tree with 'ecommerce' selected. The main query editor contains the following SQL code:

```
1 -- Calculate the average total amount for all orders dynamically
2 SELECT AVG(oi.quantity * p.price) AS average_order_total
3 FROM orders o
4 JOIN order_items oi ON o.id = oi.order_id
5 JOIN products p ON oi.product_id = p.id;
```

The 'Result Grid' pane shows a single column 'average_order_total' with a value of 1000. The 'Action Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
38	21:24:54	SELECT c.name AS customer_name, o.order_date FROM orders o JOIN customers c ON o.customer_id = c.id	3 row(s) returned	0.000 sec / 0.000 sec
39	21:25:47	SELECT id AS order_id, customer_id, total_amount FROM orders WHERE total_amount > 150.00 LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
40	21:26:38	CREATE TABLE order_items (id INT AUTO_INCREMENT PRIMARY KEY, -- Unique identifier for each ord...	0 row(s) affected	0.062 sec
41	21:26:38	ALTER TABLE orders DROP COLUMN total_amount	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec