1) Create the following tables inside the database ‘global_store_db’.(Score :2)
‘products’ with columns:

- product_id (INT, auto_increment, primary key),
- name (VARCHAR(100)),
- price (DECIMAL(10,2)),
- quantity (INT).

‘orders’ with columns:

- order_id (INT, auto_increment, primary key),
- product_id (INT, foreign key referencing product_id in the inventory table),
- quantity_ordered (INT)
- order_date (DATE).

ANS)

```
CREATE TABLE products (
   product_id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(100),
   price DECIMAL(10,2),
   quantity INT
);
DROP TABLE IF EXISTS orders;
CREATE TABLE orders (
   order_id INT AUTO_INCREMENT PRIMARY KEY,
   product_id INT,
   quantity_ordered INT,
   order_date DATE,
   FOREIGN KEY (product_id) REFERENCES inventory(product_id)
);
```

2.)Alter the products table to add a new column named category (VARCHAR(50)) after the price column.

ANS)

```
ALTER TABLE products
ADD COLUMN category VARCHAR(50) AFTER price;
```

3) Rename the products table to inventory.

ANS) RENAME TABLE products TO inventory;

4) Insert at least 10 records into the inventory table and 5 records into orders table and display the tables.

ANS) INSERT INTO inventory (name, price, quantity, category) VALUES

('Product1', 100.00, 15, 'Category1'),

('Product2', 150.00, 20, 'Category2'),

('Product3', 200.00, 5, 'Category1'),

('Product4', 50.00, 30, 'Category3'),

('Product5', 120.00, 0, 'Category2'),

('Product6', 300.00, 12, 'Category1'),

('Product7', 250.00, 0, 'Category3'),

('Product8', 180.00, 10, 'Category2'),

('Product9', 90.00, 25, 'Category1'),

('Product10', 60.00, 40, 'Category3');

INSERT INTO orders (product_id, quantity_ordered, order_date) VALUES

(1, 5, '2024-09-01'),

(2, 10, '2024-09-02'),

(3, 3, '2024-09-03'),

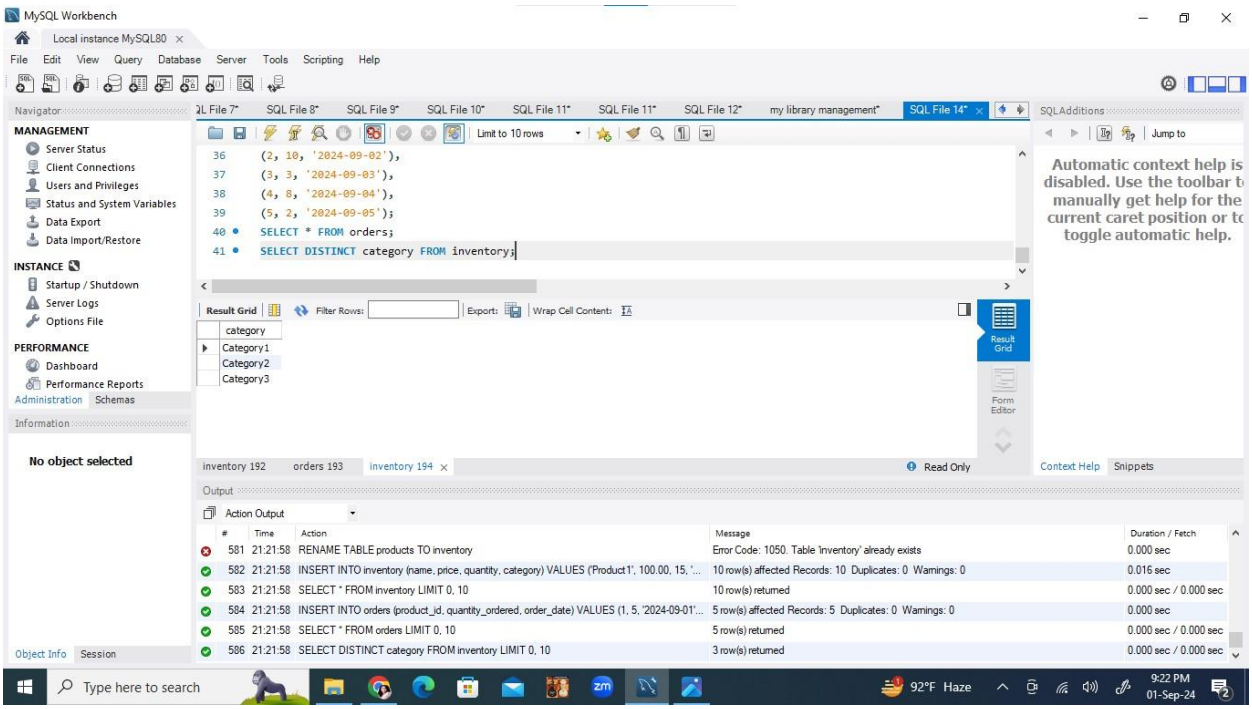
(4, 8, '2024-09-04'),

(5, 2, '2024-09-05');

5. Write queries for the following :

a) Write a query to display distinct categories from the inventory table.

Ans) SELECT DISTINCT category FROM inventory;

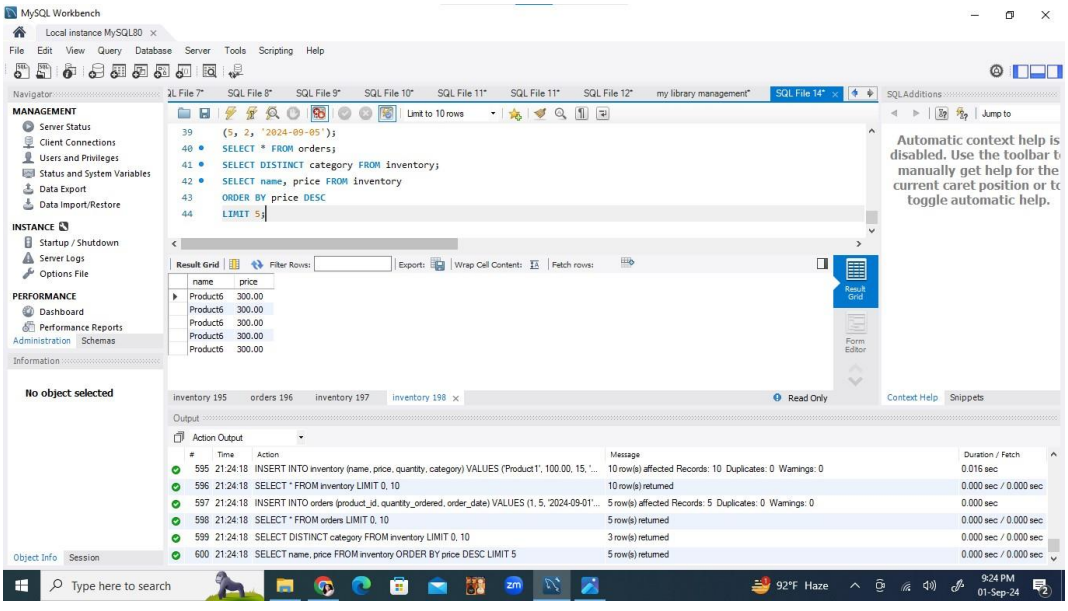


b) Select the top 5 products by their prices in descending order from the inventory table.

Ans) SELECT name, price FROM inventory

ORDER BY price DESC

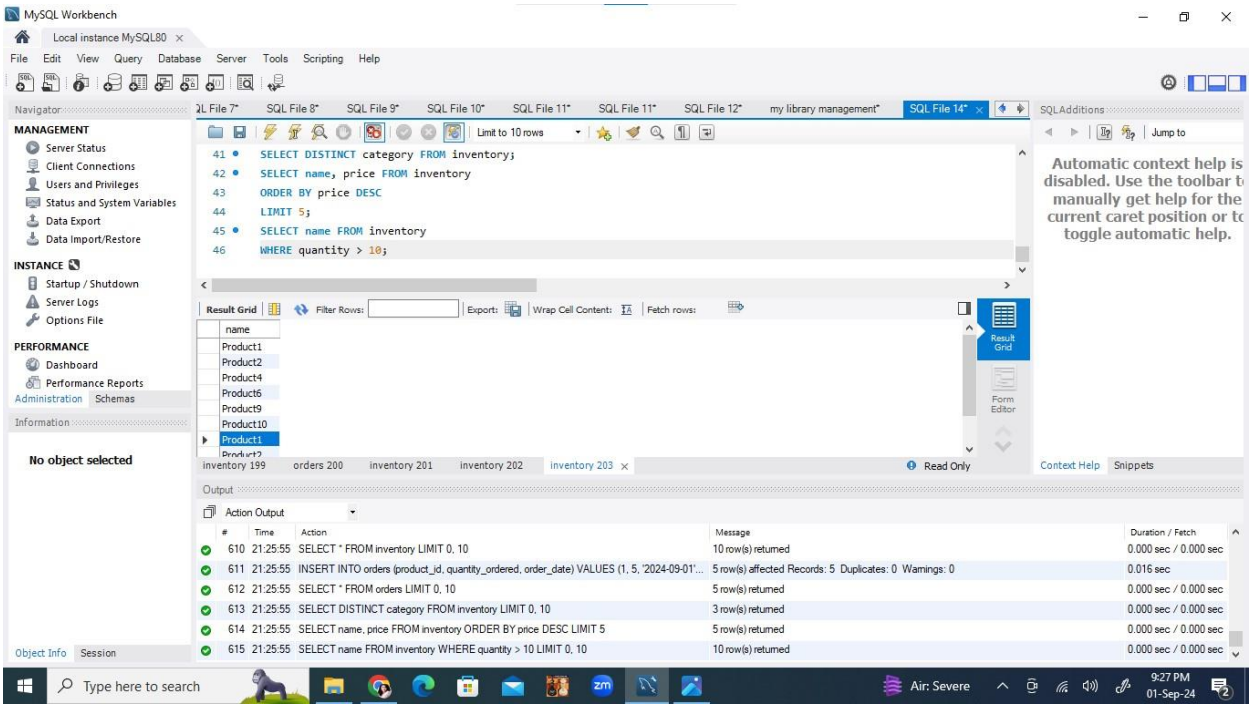
LIMIT 5;



c) Display the names of products with a quantity greater than 10 from the inventory table.

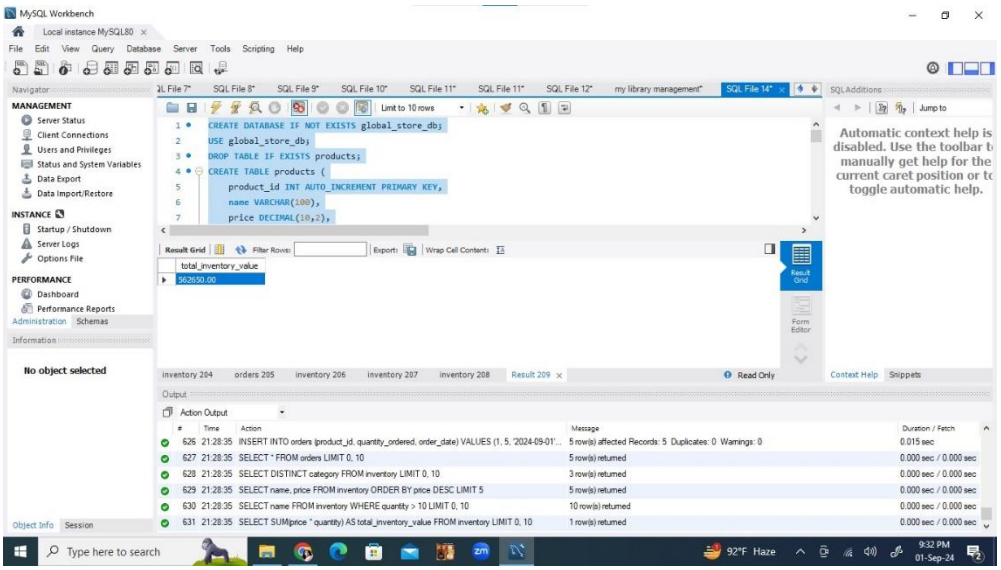
Ans) SELECT name FROM inventory

WHERE quantity > 10;



d) Use the SUM() function to calculate the total price of all products in the inventory table.

Ans) SELECT SUM(price * quantity) AS total_inventory_value FROM inventory;

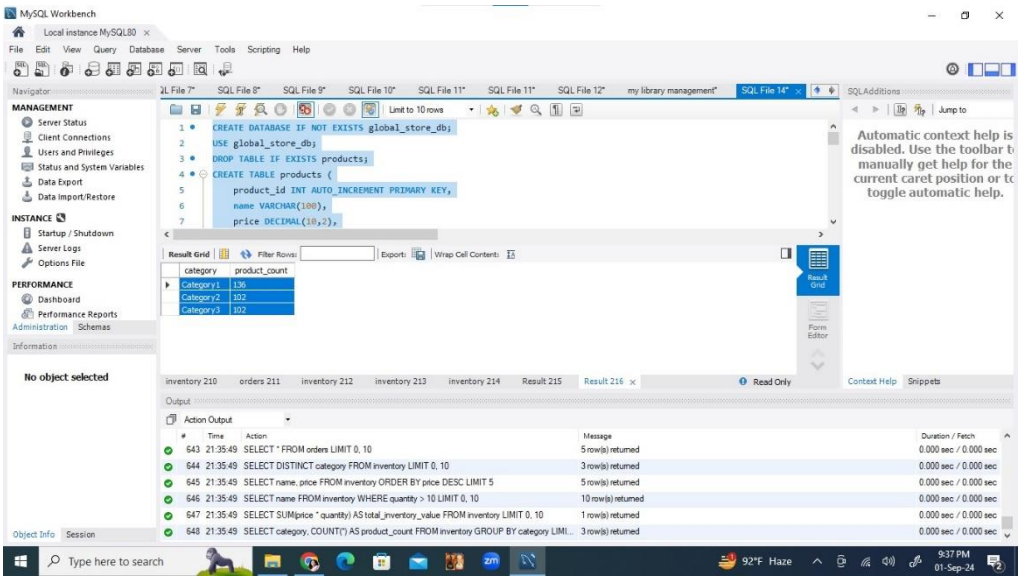


e) Group products by their categories and display the count of products in each category.

Ans) SELECT category, COUNT(*) AS product_count

FROM inventory

GROUP BY category;

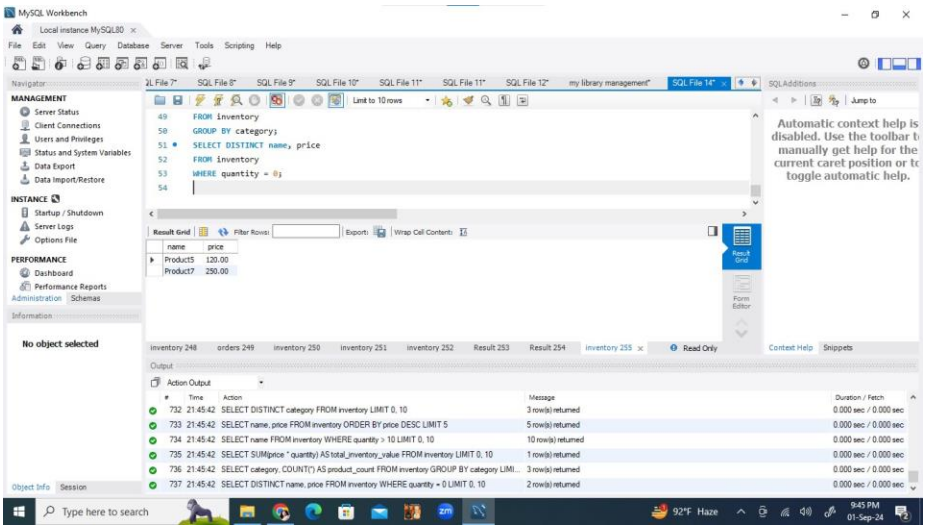


f) Write a query to identify products that are currently out of stock (i.e., quantity is zero). Display the product details including the product name and price.

Ans) SELECT DISTINCT name, price

FROM inventory

WHERE quantity = 0;

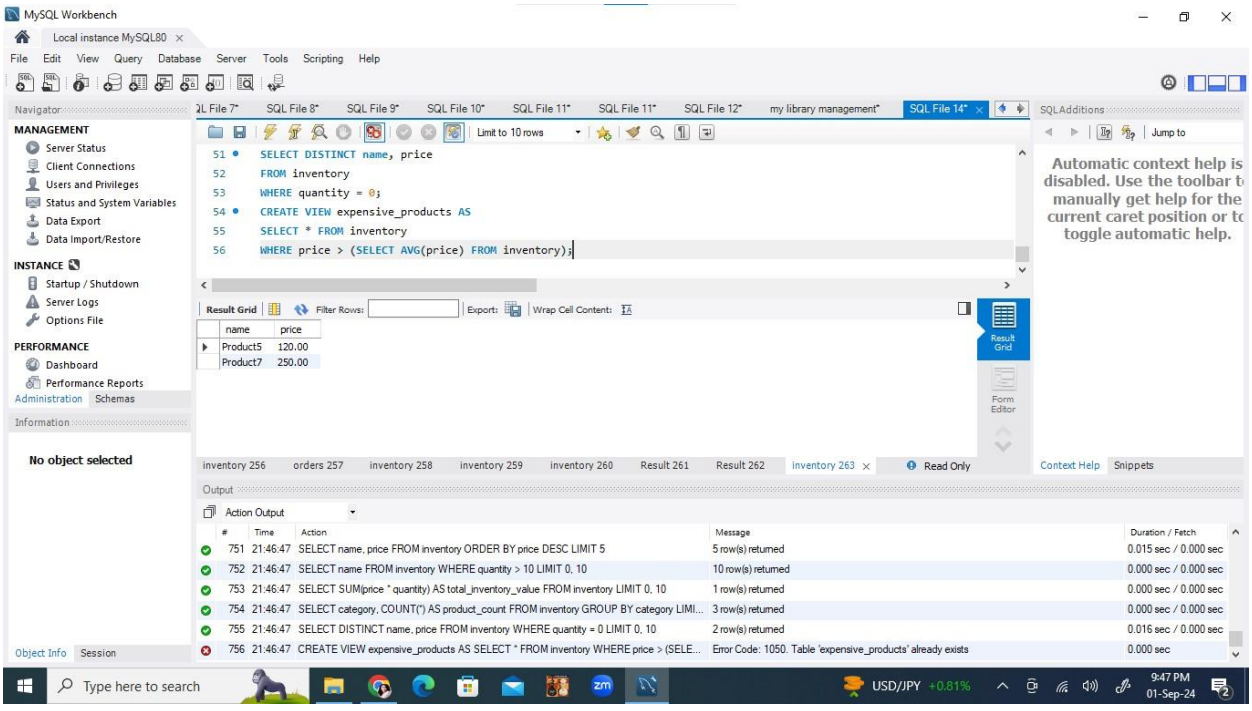


6. Create a view named expensive_products that displays the details of products with a price above the average price of all products.

CREATE VIEW expensive_products AS

SELECT * FROM inventory

WHERE price > (SELECT AVG(price) FROM inventory);



7. Write a join query to display the names of products along with the corresponding order quantities from the inventory and orders tables.

Ans) SELECT inventory.name, orders.quantity_ordered

FROM inventory

JOIN orders ON inventory.product_id = orders.product_id;