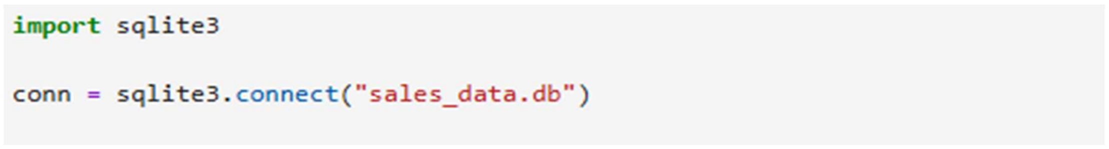
Task 7

Name: Jaavanika L

Email: ljaavanika01@gmail.com

1. Load SQLite Database

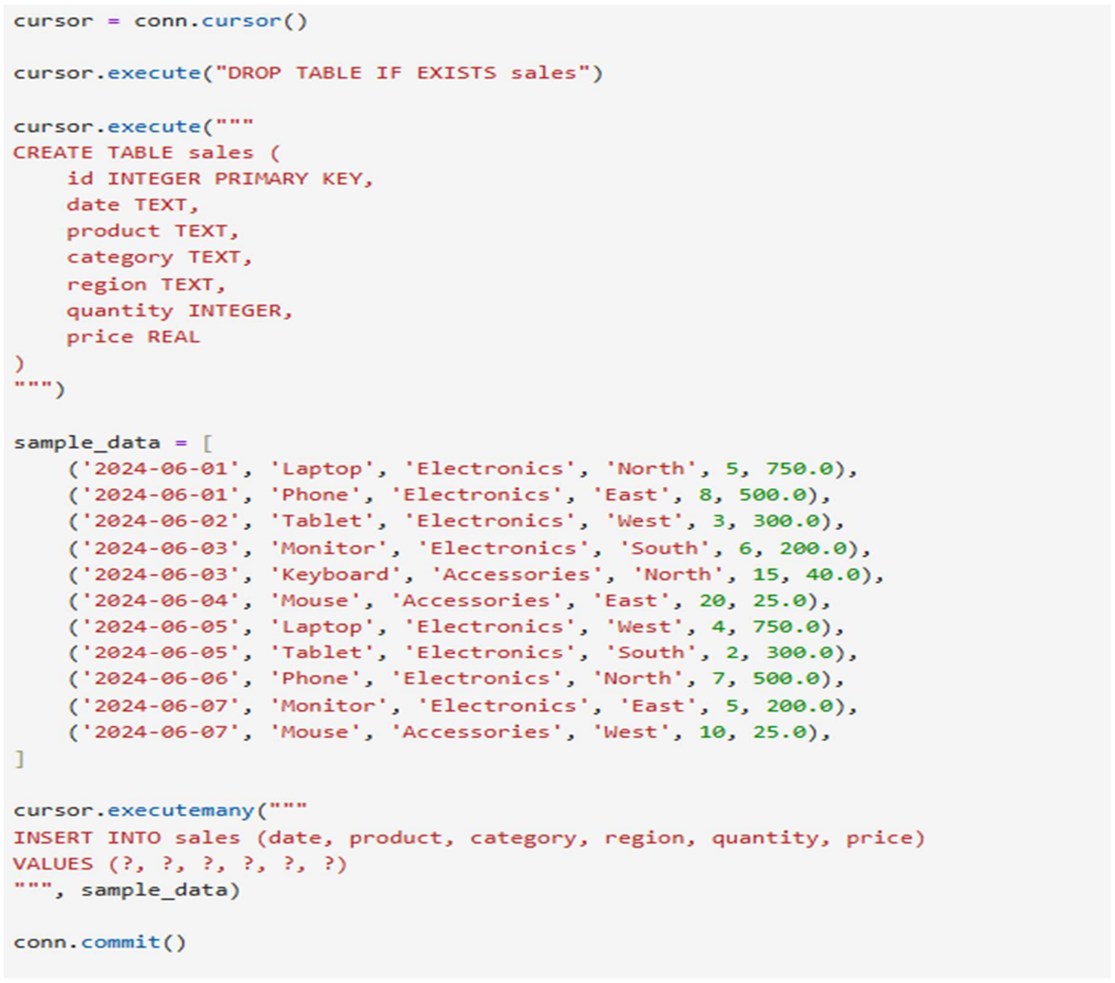
I started by importing the sqlite3 library and connecting to a local SQLite database named sales_data.db. This allowed me to interact with the database directly from Python.



```
import sqlite3

conn = sqlite3.connect("sales_data.db")
```

2. Create and Populate the Sales Table

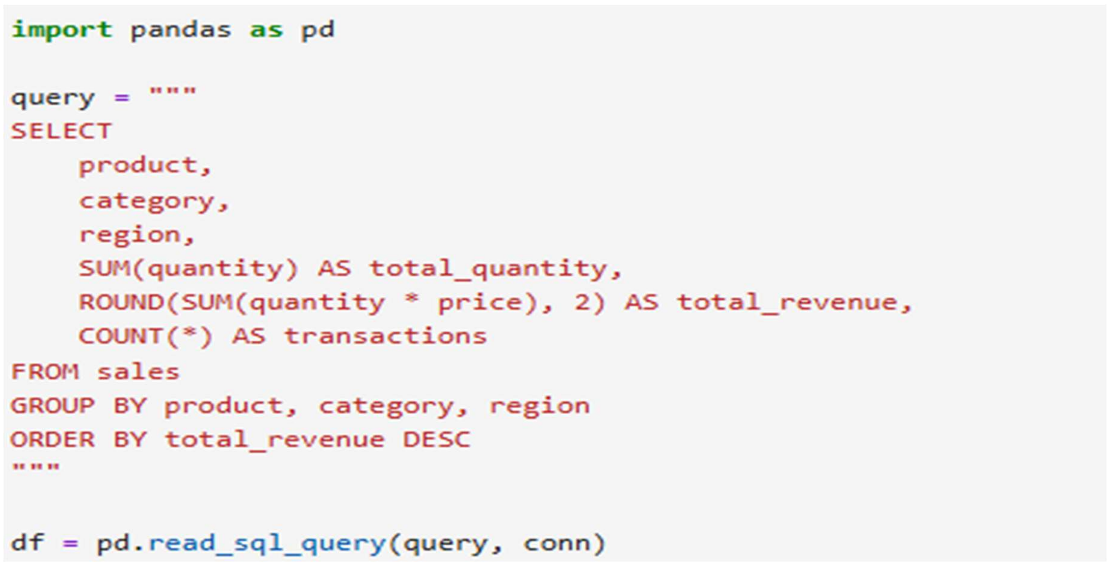
I created a sales table with fields like product, category, region, quantity, and price. Then, I inserted multiple rows of sample sales data to make the analysis meaningful and realistic.



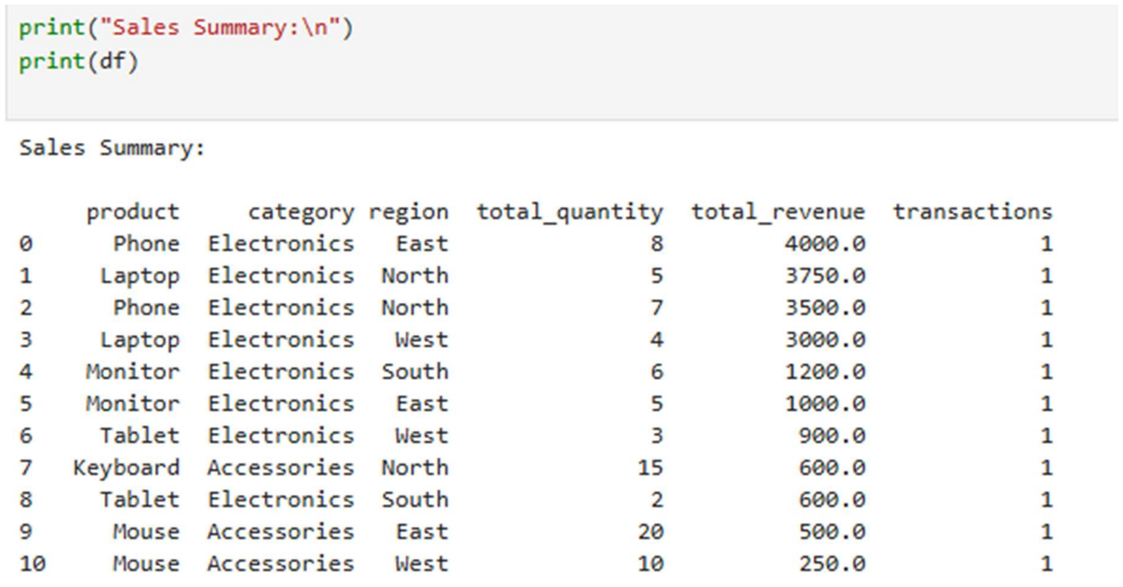
```
cursor = conn.cursor()

cursor.execute("DROP TABLE IF EXISTS sales")

cursor.execute("""
CREATE TABLE sales (
    id INTEGER PRIMARY KEY,
    date TEXT,
    product TEXT,
    category TEXT,
    region TEXT,
    quantity INTEGER,
    price REAL
)
""")

sample_data = [
    ('2024-06-01', 'Laptop', 'Electronics', 'North', 5, 750.0),
    ('2024-06-01', 'Phone', 'Electronics', 'East', 8, 500.0),
    ('2024-06-02', 'Tablet', 'Electronics', 'West', 3, 300.0),
    ('2024-06-03', 'Monitor', 'Electronics', 'South', 6, 200.0),
    ('2024-06-03', 'Keyboard', 'Accessories', 'North', 15, 40.0),
    ('2024-06-04', 'Mouse', 'Accessories', 'East', 20, 25.0),
    ('2024-06-05', 'Laptop', 'Electronics', 'West', 4, 750.0),
    ('2024-06-05', 'Tablet', 'Electronics', 'South', 2, 300.0),
    ('2024-06-06', 'Phone', 'Electronics', 'North', 7, 500.0),
    ('2024-06-07', 'Monitor', 'Electronics', 'East', 5, 200.0),
    ('2024-06-07', 'Mouse', 'Accessories', 'West', 10, 25.0),
]

cursor.executemany("""
INSERT INTO sales (date, product, category, region, quantity, price)
VALUES (?, ?, ?, ?, ?, ?)
""", sample_data)

conn.commit()
```

3. Run SQL Query 1 – Product-wise Summary

I wrote an SQL query that groups sales by product, category, and region, then calculates total quantity sold, total revenue (quantity * price), and number of transactions. I used pandas to load the query result into a DataFrame and printed it to get a detailed breakdown of product-level sales across regions.



```
import pandas as pd

query = """
SELECT
    product,
    category,
    region,
    SUM(quantity) AS total_quantity,
    ROUND(SUM(quantity * price), 2) AS total_revenue,
    COUNT(*) AS transactions
FROM sales
GROUP BY product, category, region
ORDER BY total_revenue DESC
"""

df = pd.read_sql_query(query, conn)
```



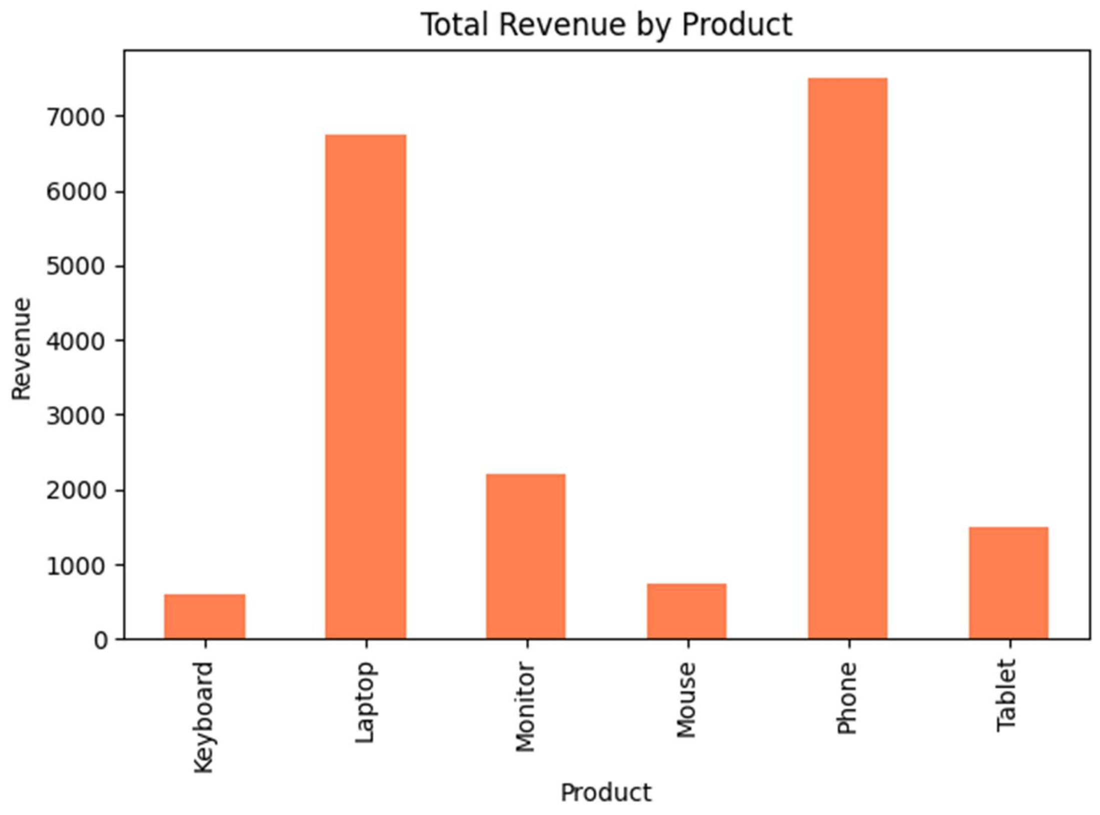
```
print("Sales Summary:\n")
print(df)
```

```
Sales Summary:

     product     category region  total_quantity  total_revenue  transactions
0      Phone  Electronics   East               8         4000.0             1
1     Laptop  Electronics  North               5         3750.0             1
2      Phone  Electronics  North               7         3500.0             1
3     Laptop  Electronics   West               4         3000.0             1
4    Monitor  Electronics  South               6         1200.0             1
5    Monitor  Electronics   East               5         1000.0             1
6     Tablet  Electronics   West               3          900.0             1
7   Keyboard  Accessories  North              15          600.0             1
8     Tablet  Electronics  South               2          600.0             1
9      Mouse  Accessories   East              20          500.0             1
10     Mouse  Accessories   West              10          250.0             1
```

4. Plot Bar Chart of Revenue per Product

I grouped the total revenue by product and used matplotlib to plot a bar chart showing revenue per product. This helped me visualize which products are generating the most income.

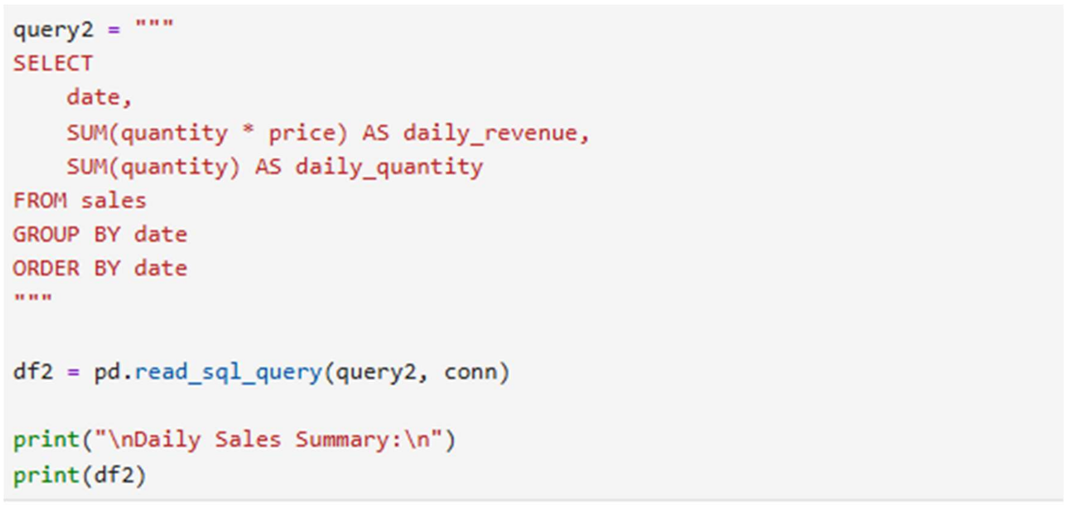
```
import matplotlib.pyplot as plt

chart_data = df.groupby('product')['total_revenue'].sum().reset_index()

chart_data.plot(kind='bar', x='product', y='total_revenue', legend=False, color='coral')

plt.title("Total Revenue by Product")
plt.xlabel("Product")
plt.ylabel("Revenue")
plt.tight_layout()

plt.show()

plt.savefig("sales_chart.png")
```



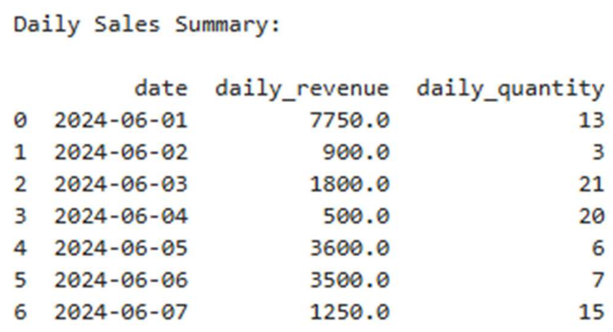
```
<Figure size 640x480 with 0 Axes>
```

5. Run SQL Query 2 – Daily Sales Trend

I created a second SQL query to calculate total daily revenue and quantity sold. This helped me understand how the business performed on different days. I used GROUP BY date and ORDER BY date to make the output more readable in chronological order.



```
query2 = """
SELECT
    date,
    SUM(quantity * price) AS daily_revenue,
    SUM(quantity) AS daily_quantity
FROM sales
GROUP BY date
ORDER BY date
"""

df2 = pd.read_sql_query(query2, conn)

print("\nDaily Sales Summary:\n")
print(df2)
```

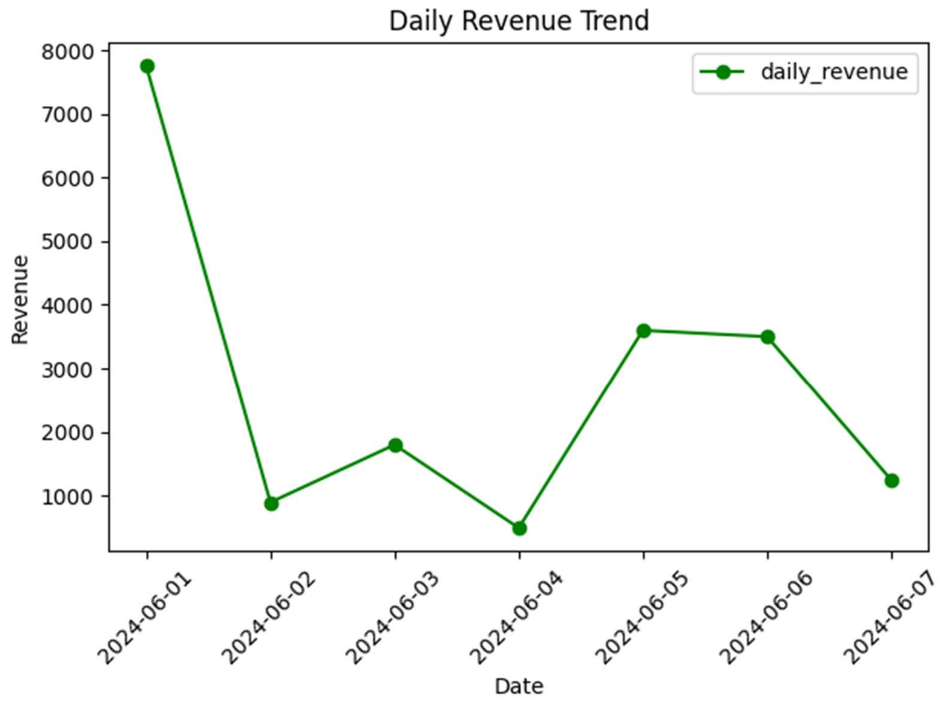


```
Daily Sales Summary:

         date  daily_revenue  daily_quantity
0  2024-06-01         7750.0              13
1  2024-06-02          900.0               3
2  2024-06-03         1800.0              21
3  2024-06-04          500.0              20
4  2024-06-05         3600.0               6
5  2024-06-06         3500.0               7
6  2024-06-07         1250.0              15
```

6. Plot Line Chart of Daily Revenue

I plotted a line chart showing how revenue changed day by day. This gave me a clear picture of sales performance trends over the week.

```python
df2.plot(kind='line', x='date', y='daily_revenue', marker='o', color='green')

plt.title("Daily Revenue Trend")
plt.xlabel("Date")
plt.ylabel("Revenue")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


Daily Revenue Trend
daily_revenue
Revenue
Date