Integration in Python

- Connection to the MySQL database using python

```
import mysql.connector
import pymysql
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings

warnings.filterwarnings('ignore')

# Database connection details
db_name = "SPARESPA"
db_host = "localhost"
db_username = "root"
db_password = "*********"

try:
    # Establish the connection
    conn = pymysql.connect(
        host=db_host,
        port=3306,
        user=db_username,
        passwd=db_password,
        db=db_name
    )
    print("Connection successful!")
except Exception as e:
    print("Error:", e)

# Create a cursor object
cursor = conn.cursor()

# Execute a query
query = "SELECT * FROM customer"
cursor.execute(query)

# Fetch and display results
results = cursor.fetchall()
for row in results:
    print(row)

# Close the cursor
cursor.close()
```

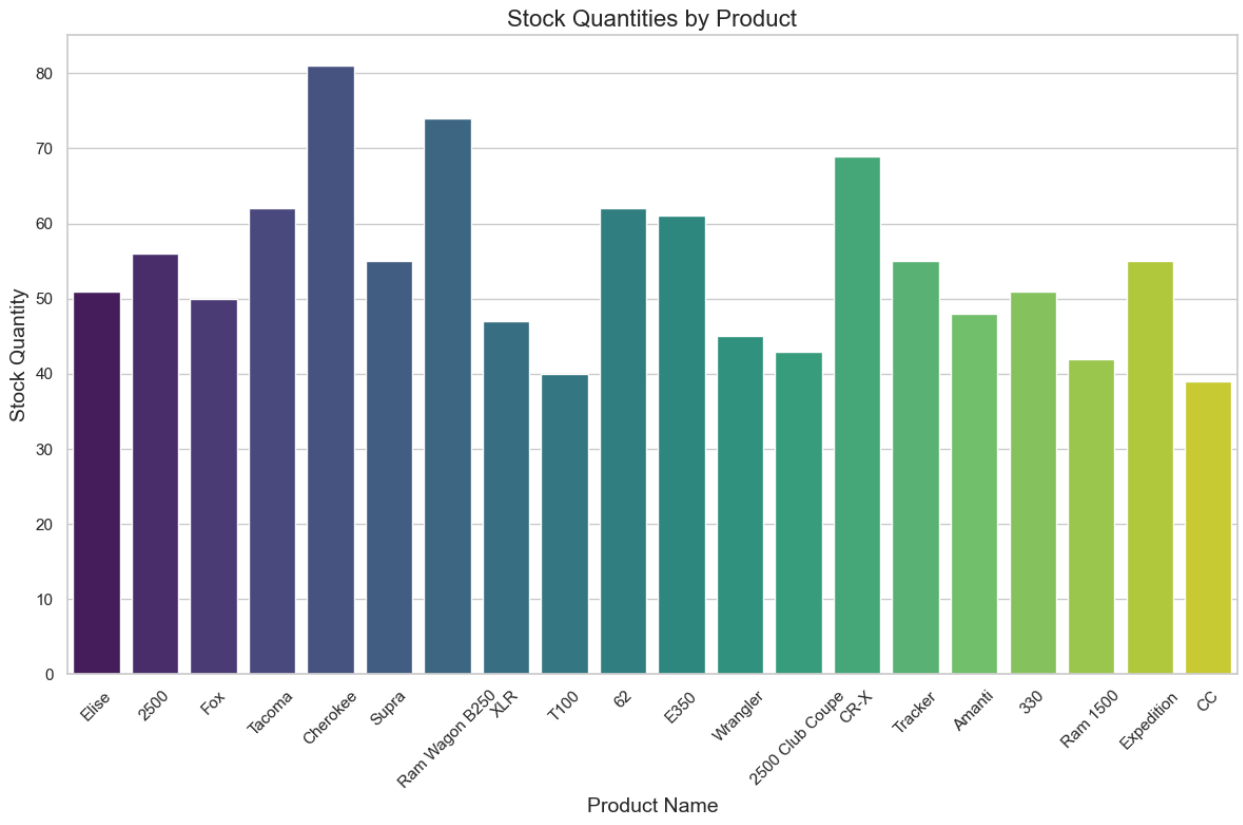
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Ensure that seaborn style is applied
sns.set_theme(style="whitegrid")
```

VISUALIZATIONS

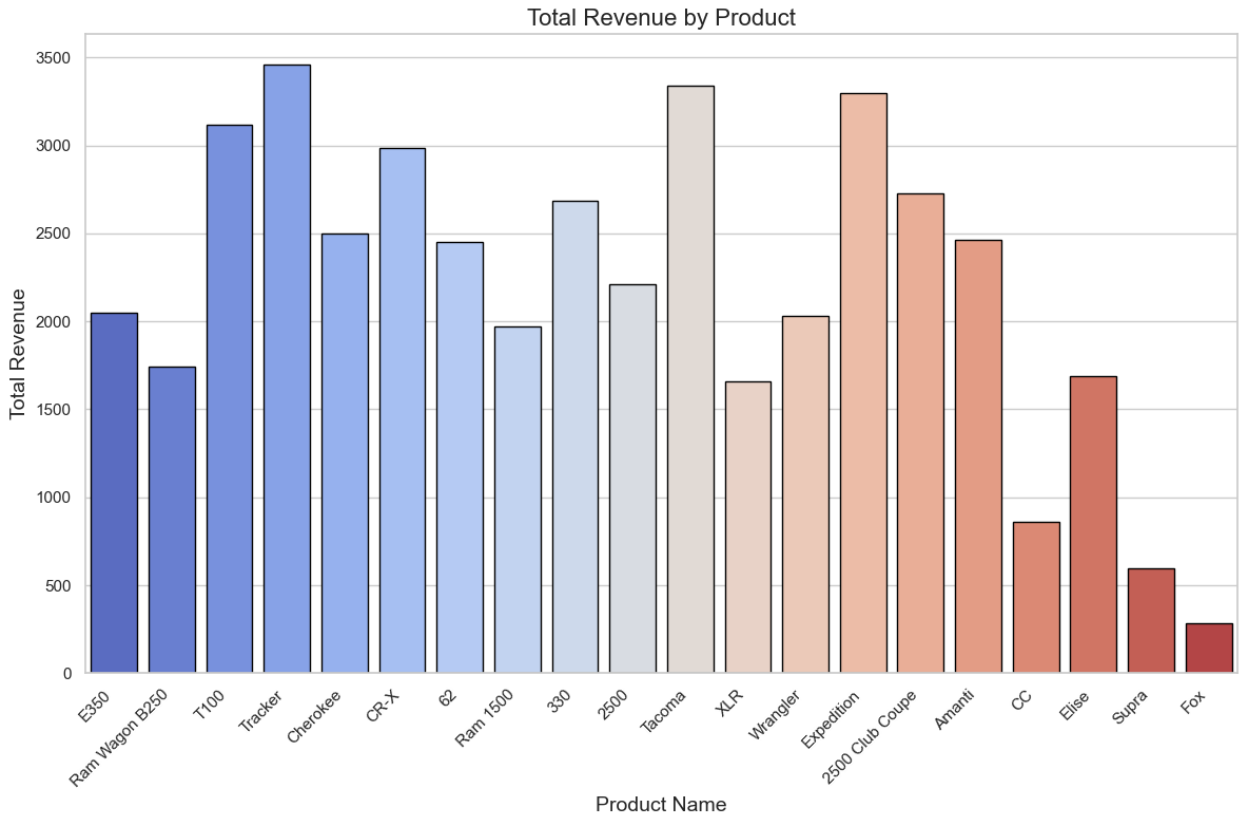
#1) Bar plot 1

```
product_df = dataframes["product"]
plt.figure(figsize=(12, 8))
sns.barplot(x='product_name', y='stock_quantity', data=product_df, palette='viridis')
plt.title("Stock Quantities by Product", fontsize=16)
plt.xlabel("Product Name", fontsize=14)
plt.ylabel("Stock Quantity", fontsize=14)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



 # 2) Bar plot 2

```
payments_df = dataframes["payments"]
plt.figure(figsize=(8, 6))
sns.boxplot(y='amount', data=payments_df, color='#5A9')
plt.title("Boxplot of Payment Amounts", fontsize=16)
plt.ylabel("Amount", fontsize=14)
plt.tight_layout()
plt.show()
```

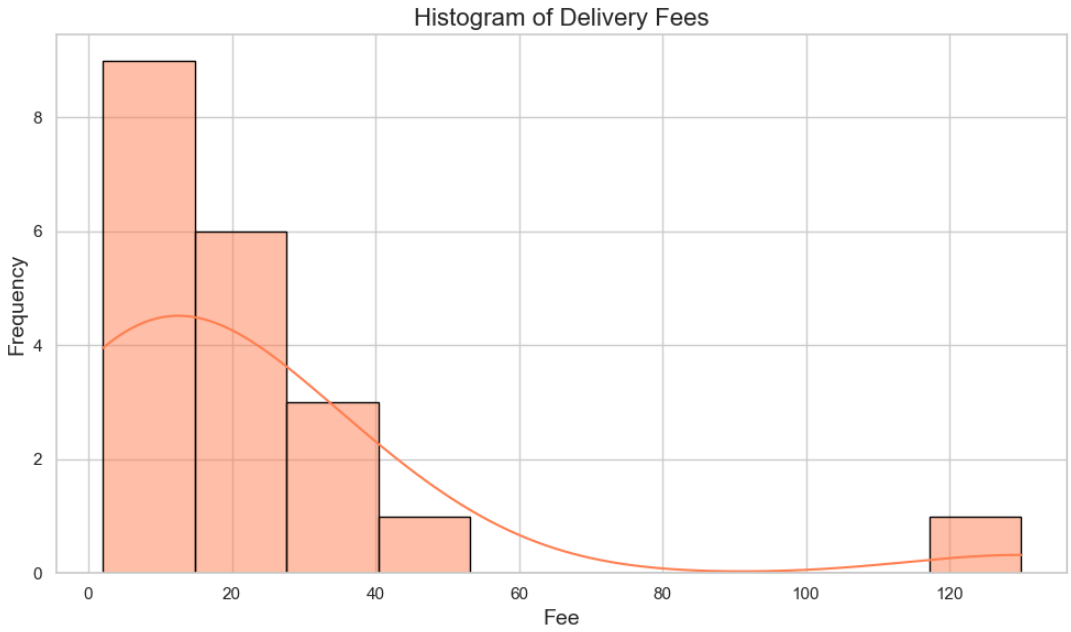


#3) Scatter plot

```
purchase_order_detail_df = dataframes["purchase_order_detail"]
plt.figure(figsize=(10, 6))
sns.scatterplot(x='unit_price', y='quantity', data=purchase_order_detail_df, s=100, color='teal')
plt.title("Unit Price vs Quantity in Purchase Order Detail", fontsize=16)
plt.xlabel("Unit Price", fontsize=14)
plt.ylabel("Quantity", fontsize=14)
plt.tight_layout()
plt.show()
```

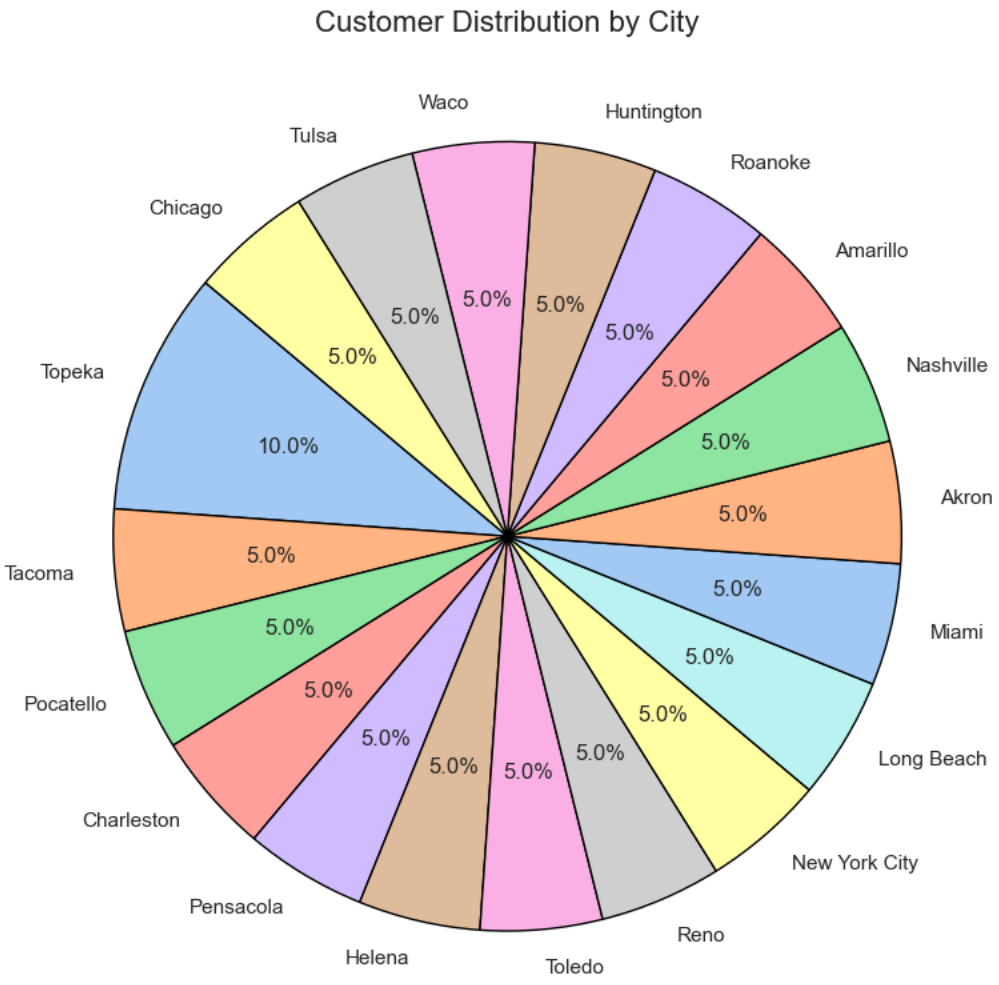
#4) Histogram

```
delivery_df = dataframes["delivery"]
plt.figure(figsize=(10, 6))
sns.histplot(delivery_df['fee'], bins=10, kde=True, color='coral')
plt.title("Histogram of Delivery Fees", fontsize=16)
plt.xlabel("Fee", fontsize=14)
plt.ylabel("Frequency", fontsize=14)
plt.tight_layout()
plt.show()
```



#5) Pie chart

```
customer_df = dataframes["customer"]
customer_city_counts = customer_df['address'].value_counts()
plt.figure(figsize=(10, 8))
plt.pie(customer_city_counts, labels=customer_city_counts.index, autopct='%1.1f%%',
startangle=140, colors=sns.color_palette("pastel"))
plt.title("Customer Distribution by City", fontsize=16)
plt.tight_layout()
plt.show()
```



- 3 QUERIES TO RETRIVE DATA FROM THE DATABASE AND VISUALIZING IT

```
import pymysql
import pandas as pd

# Database connection details
db_name = "SPARESPA"
db_host = "localhost"
db_username = "root"
db_password = "kalp1231"

# Establish the connection
try:
    conn = pymysql.connect(
        host=db_host,
        port=3306,
        user=db_username,
        passwd=db_password,
        db=db_name
    )
    print("Connection successful!")
except Exception as e:
    print("Error:", e)

# Function to execute a query and return results as a DataFrame
def execute_query(query, connection):
    try:
        return pd.read_sql(query, connection)
    except Exception as e:
        print(f"Error executing query: {query}\n{e}")
        return None
```

Query 1: Count of Orders by Product

```
query1 = """
SELECT product_id, COUNT(*) AS total_orders
FROM sales_order_detail
GROUP BY product_id
ORDER BY total_orders DESC;
"""

# Executing the query
orders_data = execute_query(query1, conn)

# Checking if data was retrieved
if orders_data is not None and not orders_data.empty:
    print("Query 1 Results:")
    print(orders_data)

    # Bar Plot
    plt.figure(figsize=(10, 6))
    plt.bar(orders_data['product_id'], orders_data['total_orders'], color='red', edgecolor='black')
    plt.title('Total Orders by Product', fontsize=16)
    plt.xlabel('Product ID', fontsize=14)
    plt.ylabel('Total Orders', fontsize=14)
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

else:
    print("No data to visualize.")
```

```
Query 1 Results:
    product_id  total_orders
0            4             9
1            9             8
2           15             8
3           14             7
4           13             7
5           10             6
6           18             6
7           19             6
8            7             5
9            5             5
10          17             5
11           2             5
12          16             5
13          11             4
14          12             4
15           8             3
16           1             3
17          20             2
18           6             1
19           3             1
```

Query 2: Top 5 Products with the Highest Stock Quantities

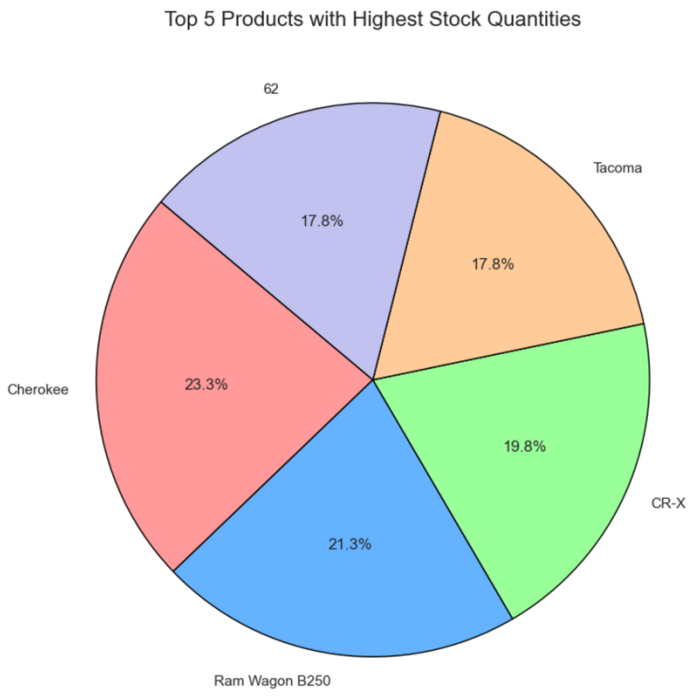
```
query2 = """
SELECT product_name, stock_quantity
FROM product
ORDER BY stock_quantity DESC
LIMIT 5;
"""
# Executing the query
top_stock_products = execute_query(query2, conn)

# Checking if data was retrieved
if top_stock_products is not None and not top_stock_products.empty:
    print("Query 2 Results:")
    print(top_stock_products)

  # Pie Chart (Visualization)
    plt.figure(figsize=(8, 8))
    colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99', '#c2c2f0']  # Define a list of colors
    plt.pie(
    top_stock_products['stock_quantity'],
    labels=top_stock_products['product_name'],
    autopct='%1.1f%%',
    startangle=140,
    colors=colors,  # Use the defined color list
    wedgeprops={'edgecolor': 'black'}  # Add black borders
    )
    plt.title('Top 5 Products with Highest Stock Quantities', fontsize=16)
    plt.tight_layout()
    plt.show()

else:
    print("No data to visualize.")
```

```
Query 2 Results:
     product_name  stock_quantity
0        Cherokee              81
1  Ram Wagon B250              74
2            CR-X              69
3         Tacoma              62
4              62              62
```



Query 3: Listing All Products with Associated Purchase Order Details Limiting to 10

```
query3 = """
SELECT p.product_name, pod.quantity
FROM product p
RIGHT JOIN purchase_order_detail pod ON p.Product_id = pod.Product_id
LIMIT 10;
"""
# Executing the query
result = execute_query(query3, conn)

# Checking if data was retrieved
if result is not None and not result.empty:
    print("Query 3 Results:")
    print(result)

    # Scatter Plot (Visualization)
    plt.figure(figsize=(10, 6))
    plt.scatter(result['product_name'], result['quantity'], color='red', s=100, edgecolor='black')
    plt.title('Product Quantity', fontsize=16)
    plt.xlabel('Product Name', fontsize=14)
    plt.ylabel('Quantity', fontsize=14)
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

else:
    print("No data retrieved or query returned empty results.")
```

```
Query 3 Results:
      product_name  quantity
0              Fox         8
1  2500 Club Coupe         1
2               62         7
3           Amanti        11
4           Amanti         9
5       Expedition        10
6             T100        15
7           Tacoma         5
8               62        14
9             T100        15
```


```
 # Closing the database connection
conn.close()
```