EDS

Theory

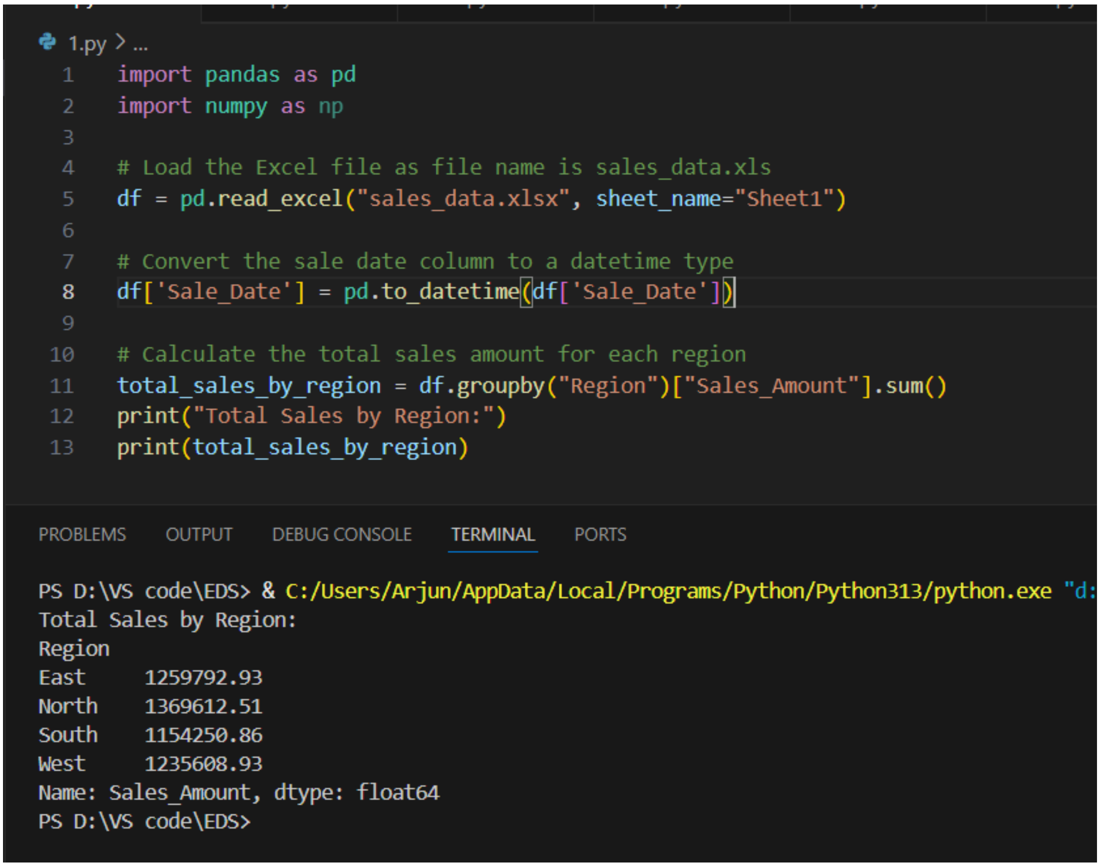
Activity No. 1

Name: Arjun Vitthal Mankuskar

Roll no: CS7 – 43

PRN:202401110038

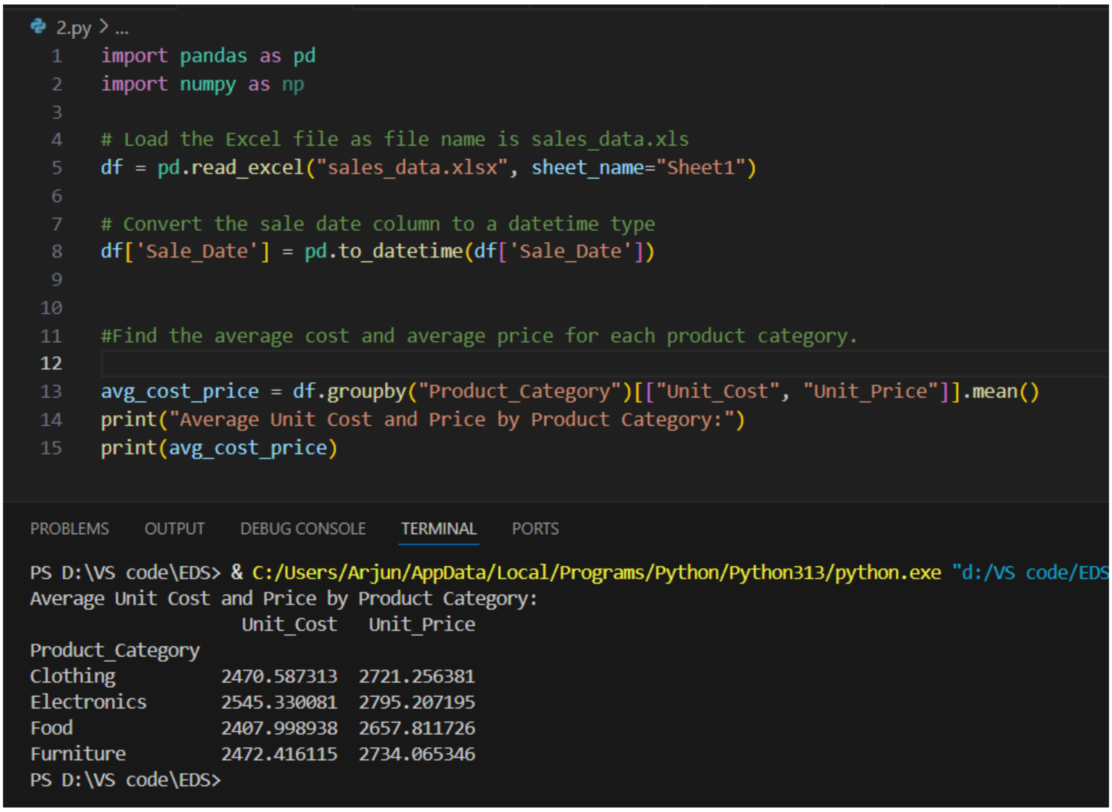
1. Calculate the total sales amount for each region.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xls
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

# Calculate the total sales amount for each region
total_sales_by_region = df.groupby("Region")["Sales_Amount"].sum()
print("Total Sales by Region:")
print(total_sales_by_region)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:
Total Sales by Region:
Region
East     1259792.93
North    1369612.51
South    1154250.86
West     1235608.93
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS>
```

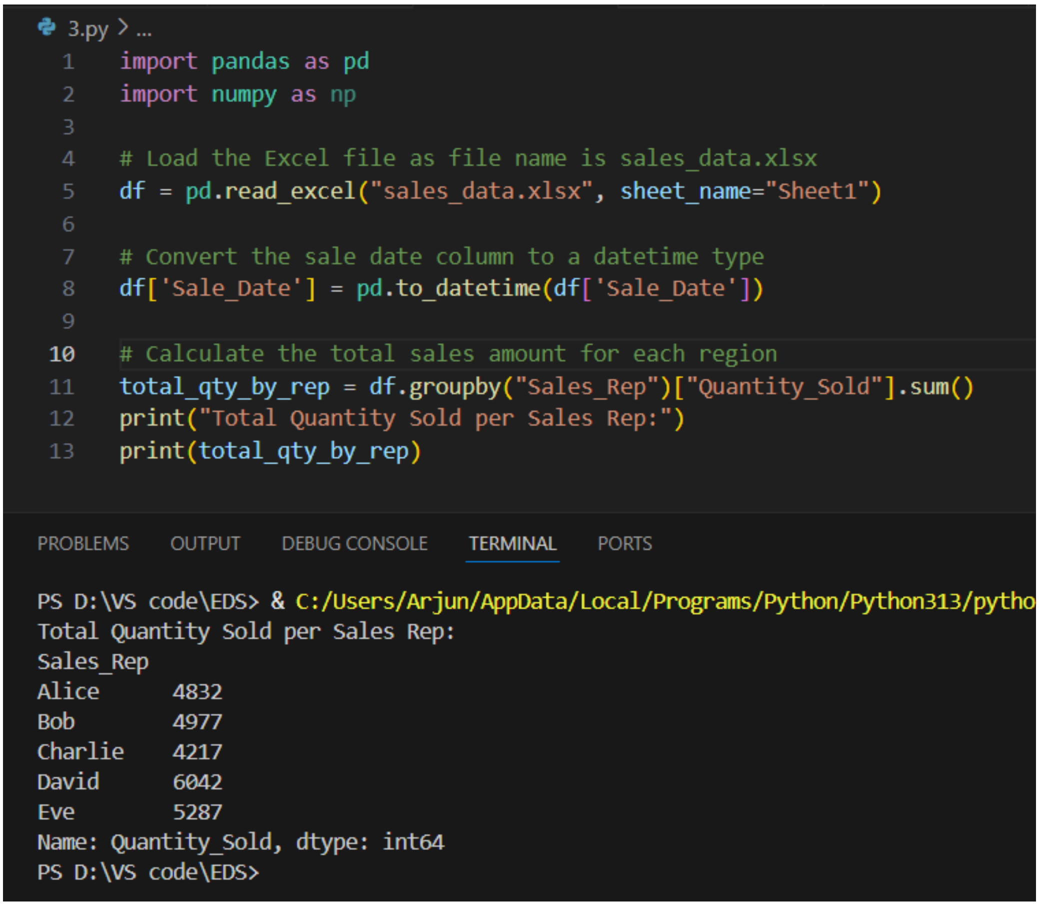
2. Find the average unit cost and unit price for each product category.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xls
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])


#Find the average cost and average price for each product category.

avg_cost_price = df.groupby("Product_Category")[["Unit_Cost", "Unit_Price"]].mean()
print("Average Unit Cost and Price by Product Category:")
print(avg_cost_price)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS
Average Unit Cost and Price by Product Category:
                    Unit_Cost   Unit_Price
Product_Category
Clothing          2470.587313  2721.256381
Electronics       2545.330081  2795.207195
Food              2407.998938  2657.811726
Furniture         2472.416115  2734.065346
PS D:\VS code\EDS>
```

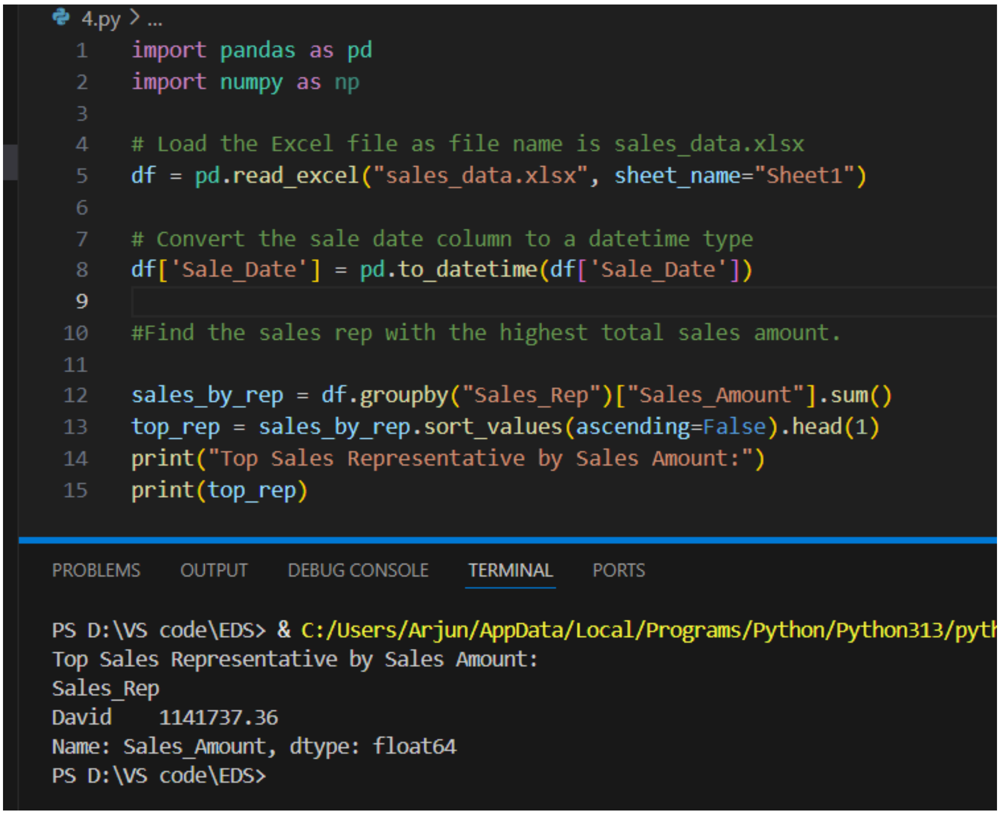
3. Determine the total quantity of items sold by each sales representative.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

# Calculate the total sales amount for each region
total_qty_by_rep = df.groupby("Sales_Rep")["Quantity_Sold"].sum()
print("Total Quantity Sold per Sales Rep:")
print(total_qty_by_rep)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/pytho
Total Quantity Sold per Sales Rep:
Sales_Rep
Alice      4832
Bob        4977
Charlie    4217
David      6042
Eve        5287
Name: Quantity_Sold, dtype: int64
PS D:\VS code\EDS>
```

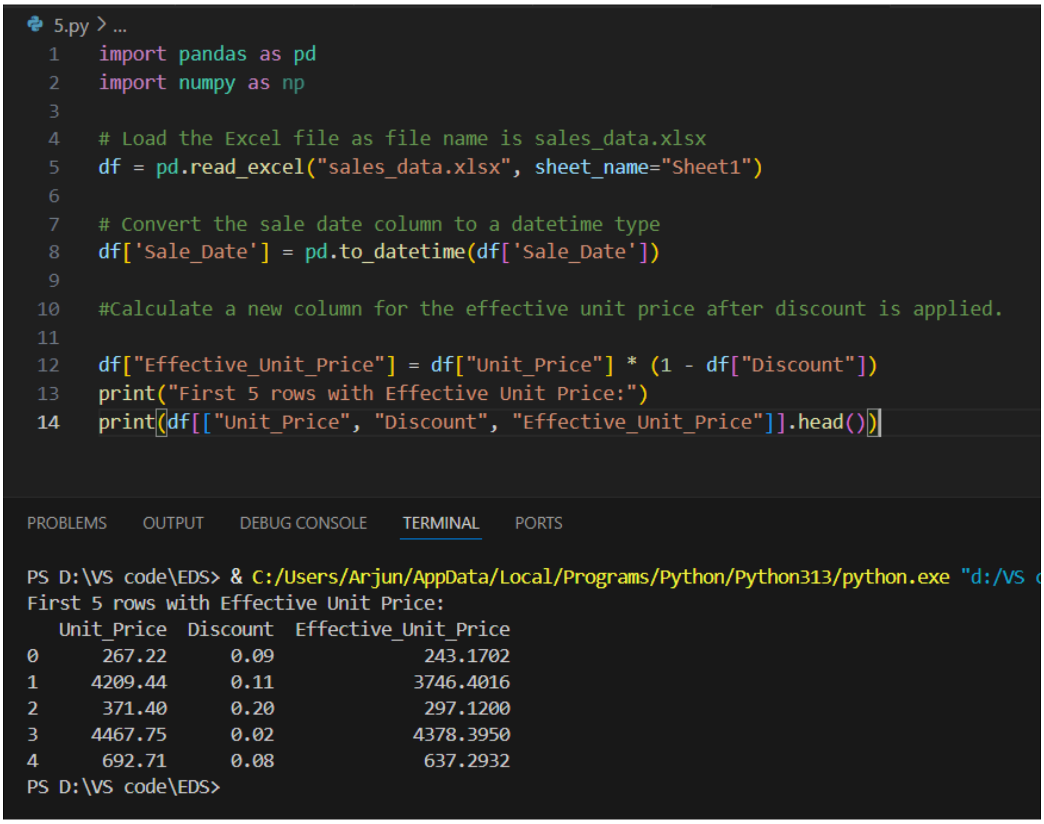
4. Identify the sales representative with the highest total sales.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Find the sales rep with the highest total sales amount.

sales_by_rep = df.groupby("Sales_Rep")["Sales_Amount"].sum()
top_rep = sales_by_rep.sort_values(ascending=False).head(1)
print("Top Sales Representative by Sales Amount:")
print(top_rep)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/pyth
Top Sales Representative by Sales Amount:
Sales_Rep
David    1141737.36
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS>
```

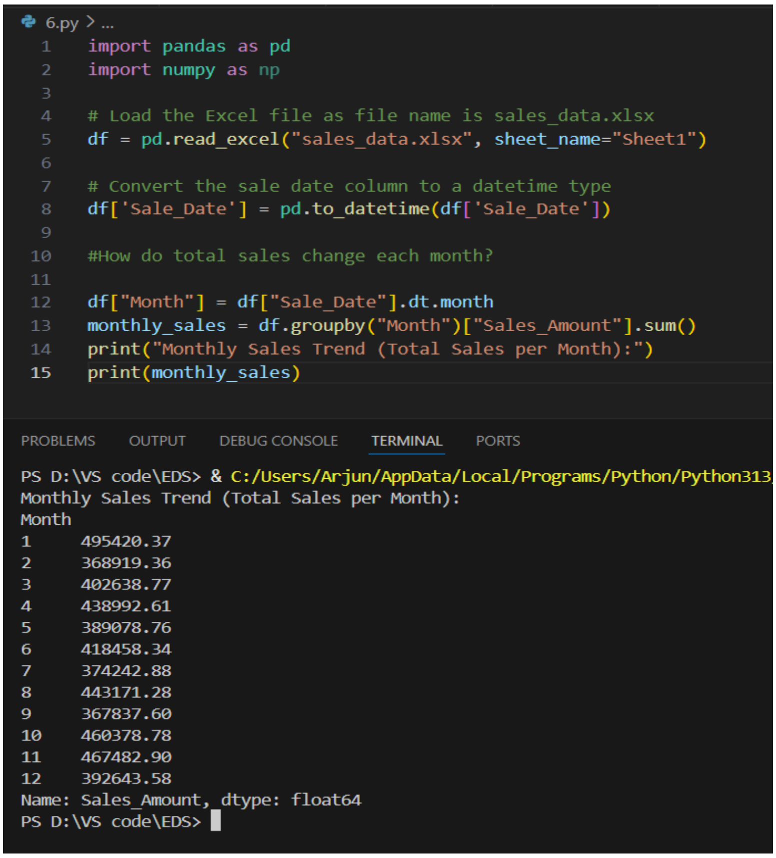
5. Calculate the effective unit price after discount for each sale.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Calculate a new column for the effective unit price after discount is applied.

df["Effective_Unit_Price"] = df["Unit_Price"] * (1 - df["Discount"])
print("First 5 rows with Effective Unit Price:")
print(df[["Unit_Price", "Discount", "Effective_Unit_Price"]].head())
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS
First 5 rows with Effective Unit Price:
   Unit_Price  Discount  Effective_Unit_Price
0      267.22      0.09              243.1702
1     4209.44      0.11             3746.4016
2      371.40      0.20              297.1200
3     4467.75      0.02             4378.3950
4      692.71      0.08              637.2932
PS D:\VS code\EDS>
```

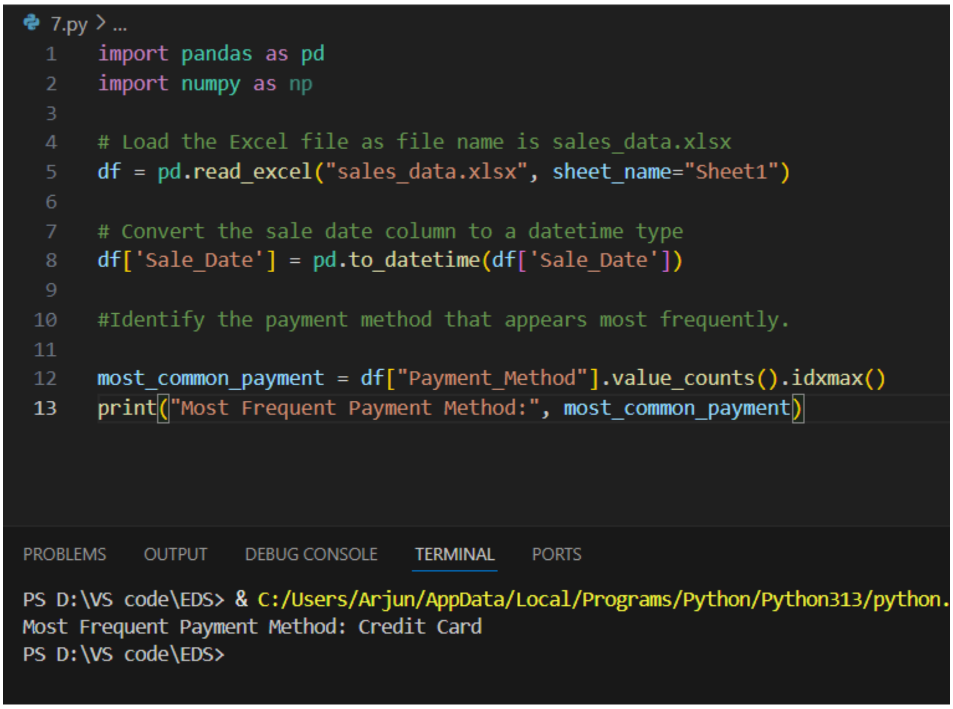
6. Analyze the monthly trend in total sales.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#How do total sales change each month?

df["Month"] = df["Sale_Date"].dt.month
monthly_sales = df.groupby("Month")["Sales_Amount"].sum()
print("Monthly Sales Trend (Total Sales per Month):")
print(monthly_sales)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313
Monthly Sales Trend (Total Sales per Month):
Month
1     495420.37
2     368919.36
3     402638.77
4     438992.61
5     389078.76
6     418458.34
7     374242.88
8     443171.28
9     367837.60
10    460378.78
11    467482.90
12    392643.58
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS> 
```

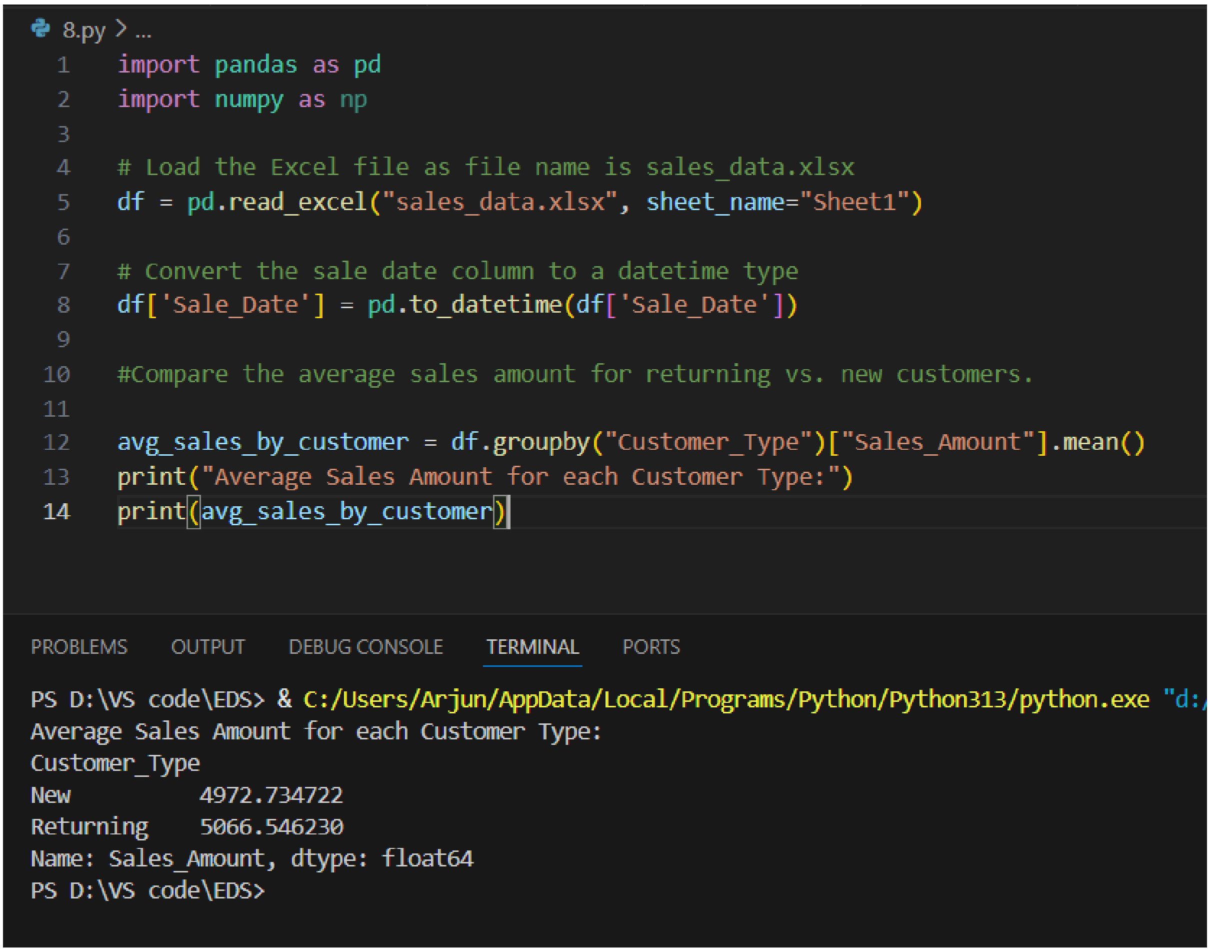
7. Find the most commonly used payment method.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Identify the payment method that appears most frequently.

most_common_payment = df["Payment_Method"].value_counts().idxmax()
print("Most Frequent Payment Method:", most_common_payment)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.
Most Frequent Payment Method: Credit Card
PS D:\VS code\EDS>
```

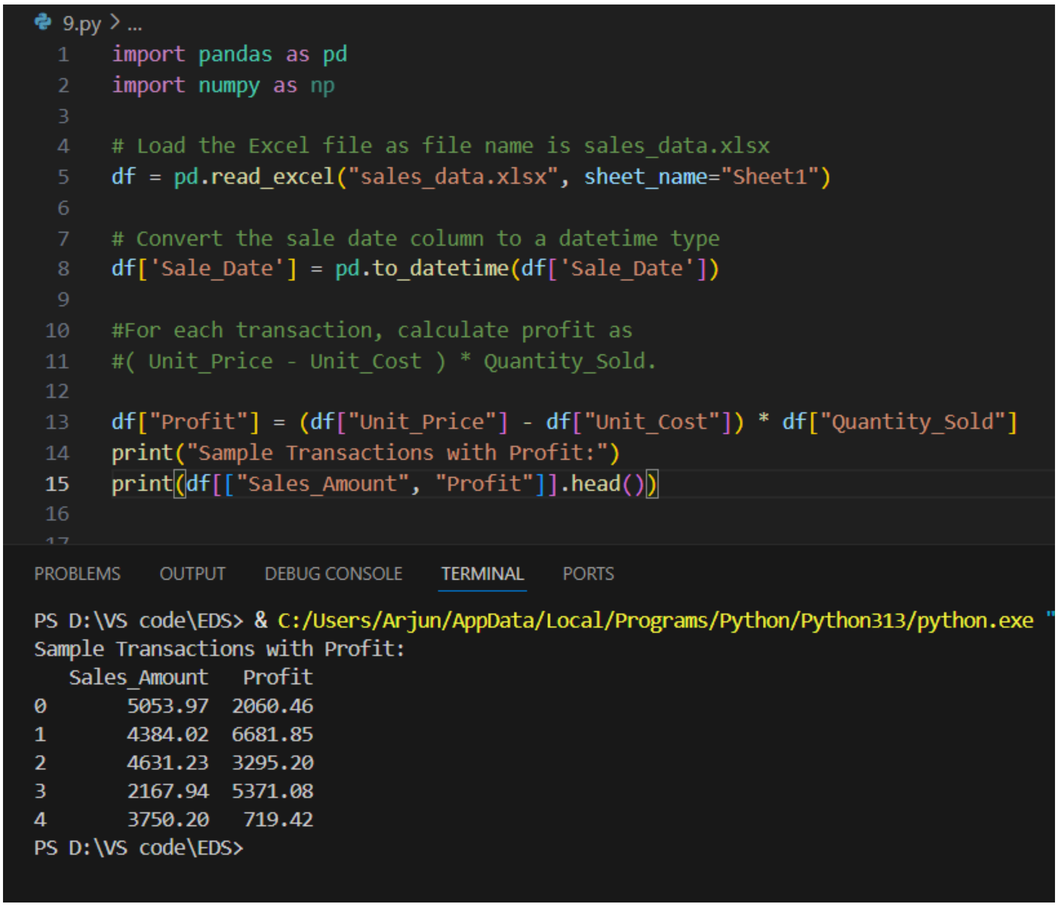
8. Compare the average sales amount for new vs. returning customers.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Compare the average sales amount for returning vs. new customers.

avg_sales_by_customer = df.groupby("Customer_Type")["Sales_Amount"].mean()
print("Average Sales Amount for each Customer Type:")
print(avg_sales_by_customer)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/
Average Sales Amount for each Customer Type:
Customer_Type
New          4972.734722
Returning    5066.546230
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS>
```

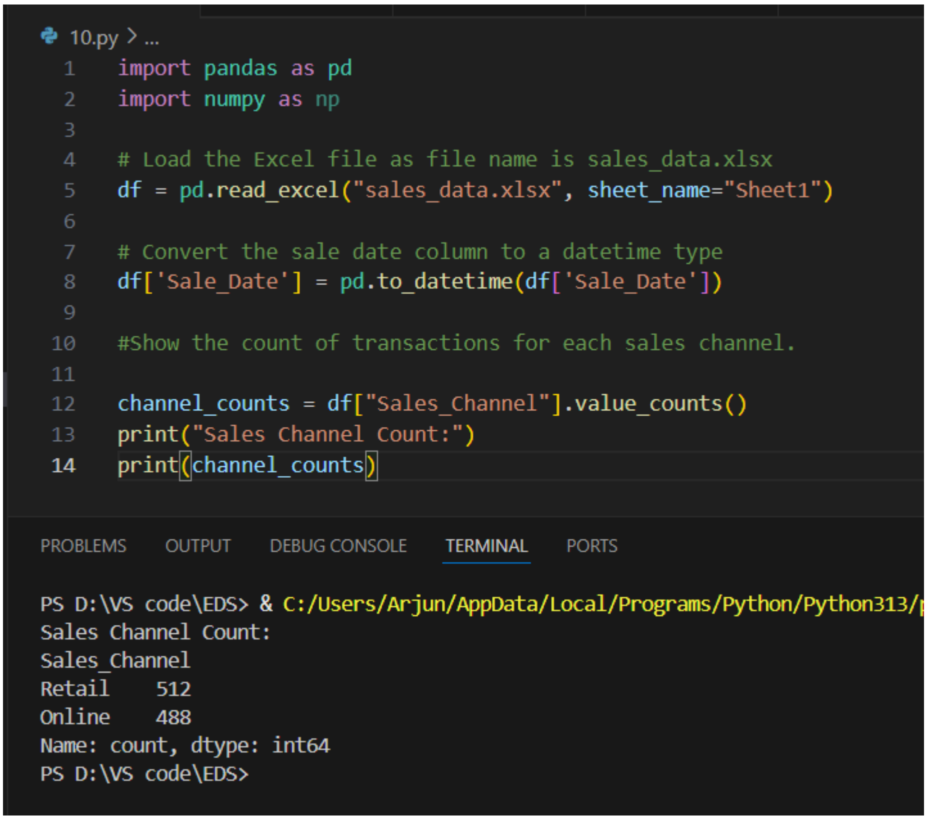
9. Calculate the profit for each transaction.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#For each transaction, calculate profit as
#( Unit_Price - Unit_Cost ) * Quantity_Sold.

df["Profit"] = (df["Unit_Price"] - df["Unit_Cost"]) * df["Quantity_Sold"]
print("Sample Transactions with Profit:")
print(df[["Sales_Amount", "Profit"]].head())
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "
Sample Transactions with Profit:
   Sales_Amount   Profit
0       5053.97  2060.46
1       4384.02  6681.85
2       4631.23  3295.20
3       2167.94  5371.08
4       3750.20   719.42
PS D:\VS code\EDS>
```

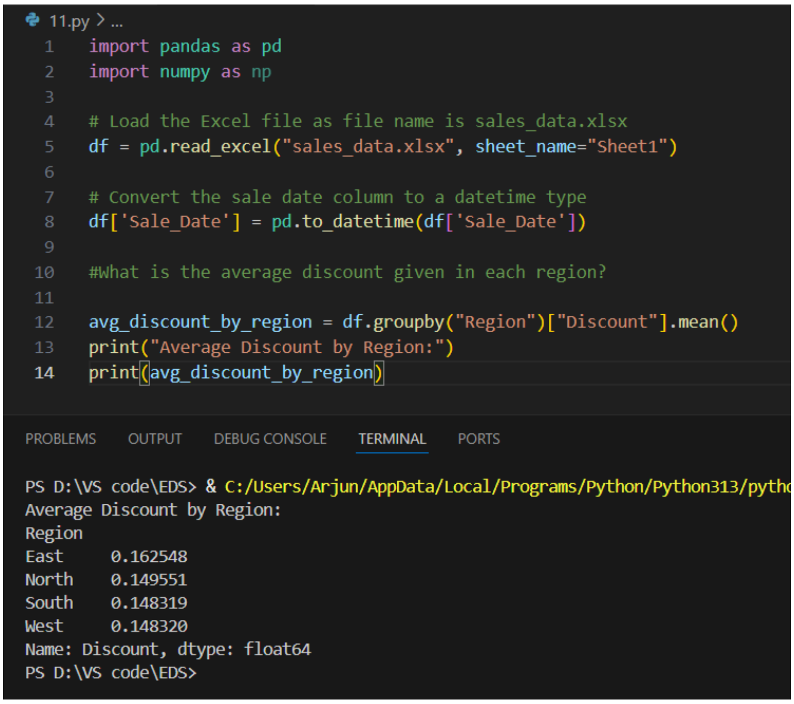
10. Count the number of transactions for each sales channel.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Show the count of transactions for each sales channel.

channel_counts = df["Sales_Channel"].value_counts()
print("Sales Channel Count:")
print(channel_counts)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/p
Sales Channel Count:
Sales_Channel
Retail    512
Online    488
Name: count, dtype: int64
PS D:\VS code\EDS>
```

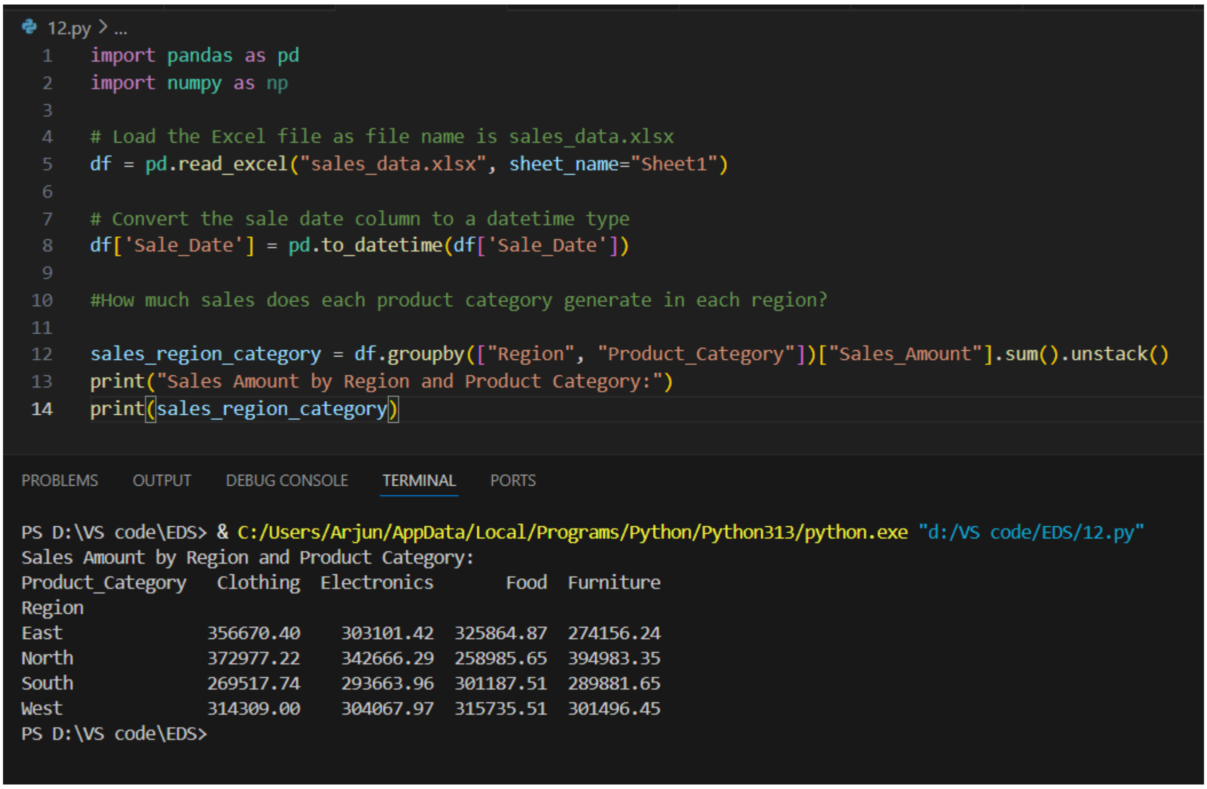
11. Calculate the average discount given in each region.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#What is the average discount given in each region?

avg_discount_by_region = df.groupby("Region")["Discount"].mean()
print("Average Discount by Region:")
print(avg_discount_by_region)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/pytho
Average Discount by Region:
Region
East     0.162548
North    0.149551
South    0.148319
West     0.148320
Name: Discount, dtype: float64
PS D:\VS code\EDS>
```

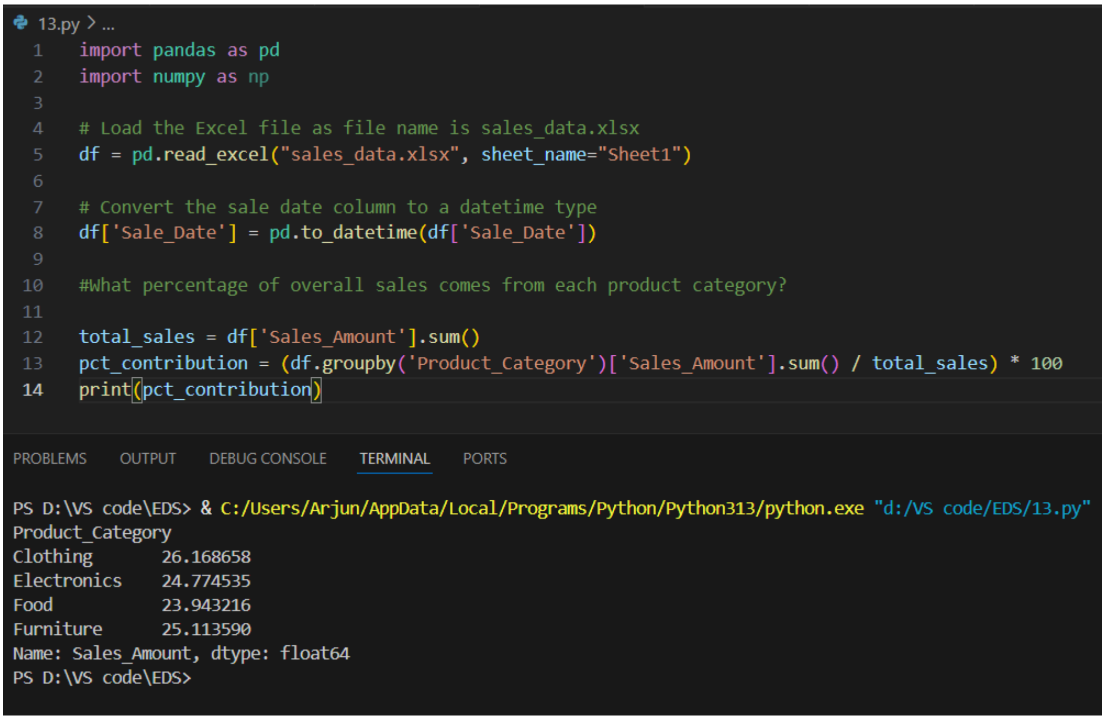
12. Determine the total sales for each product category in every region.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#How much sales does each product category generate in each region?

sales_region_category = df.groupby(["Region", "Product_Category"])["Sales_Amount"].sum().unstack()
print("Sales Amount by Region and Product Category:")
print(sales_region_category)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS/12.py"
Sales Amount by Region and Product Category:
Product_Category   Clothing  Electronics       Food  Furniture
Region
East              356670.40    303101.42  325864.87  274156.24
North             372977.22    342666.29  258985.65  394983.35
South             269517.74    293663.96  301187.51  289881.65
West              314309.00    304067.97  315735.51  301496.45
PS D:\VS code\EDS>
```

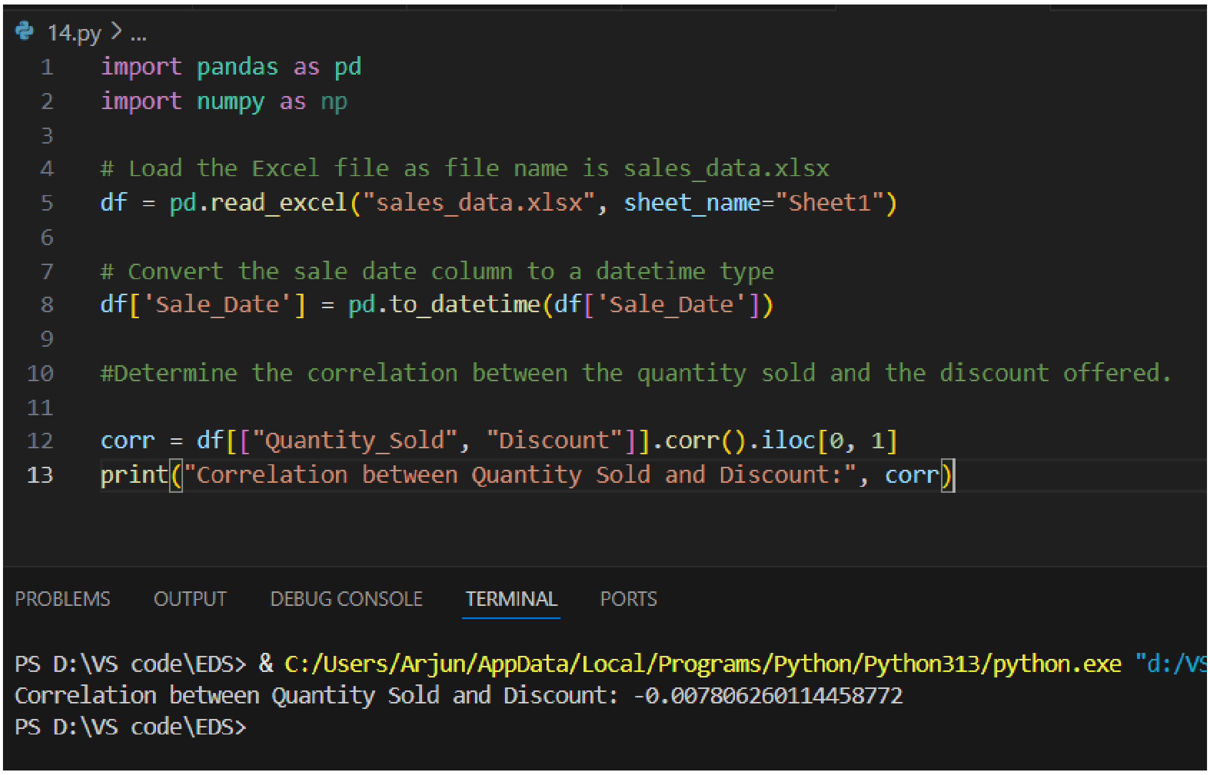
13. What percentage of overall sales comes from each product category?



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#What percentage of overall sales comes from each product category?

total_sales = df['Sales_Amount'].sum()
pct_contribution = (df.groupby('Product_Category')['Sales_Amount'].sum() / total_sales) * 100
print(pct_contribution)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS/13.py"
Product_Category
Clothing       26.168658
Electronics    24.774535
Food           23.943216
Furniture      25.113590
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS>
```

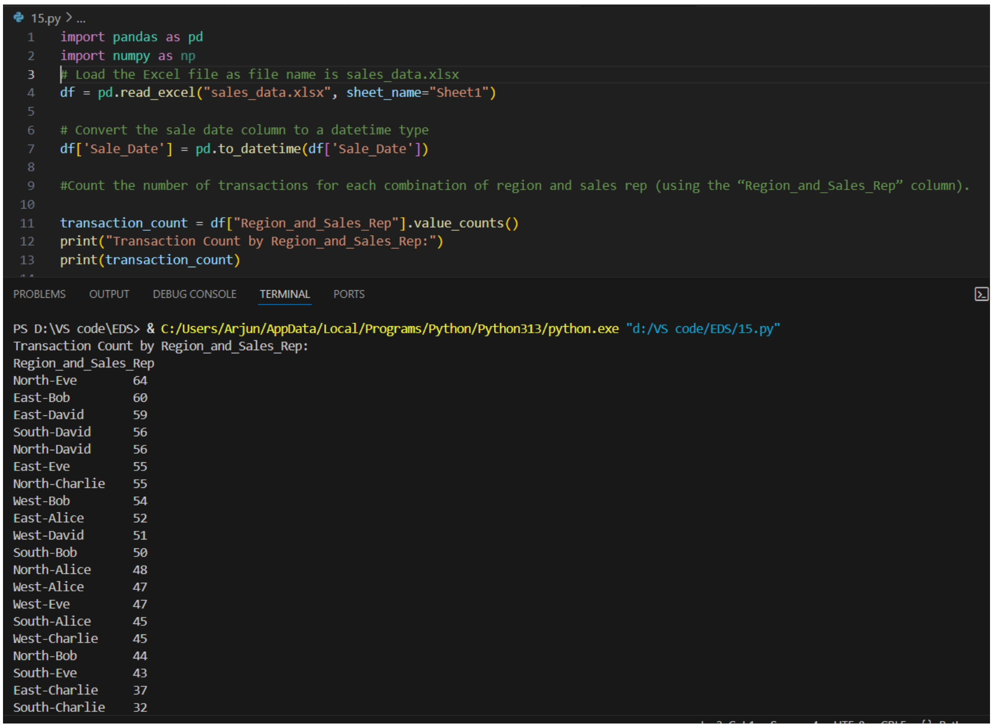
14. Examine the relationship between the number of items sold and the discount offered.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Determine the correlation between the quantity sold and the discount offered.

corr = df[["Quantity_Sold", "Discount"]].corr().iloc[0, 1]
print("Correlation between Quantity Sold and Discount:", corr)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS
Correlation between Quantity Sold and Discount: -0.007806260114458772
PS D:\VS code\EDS>
```

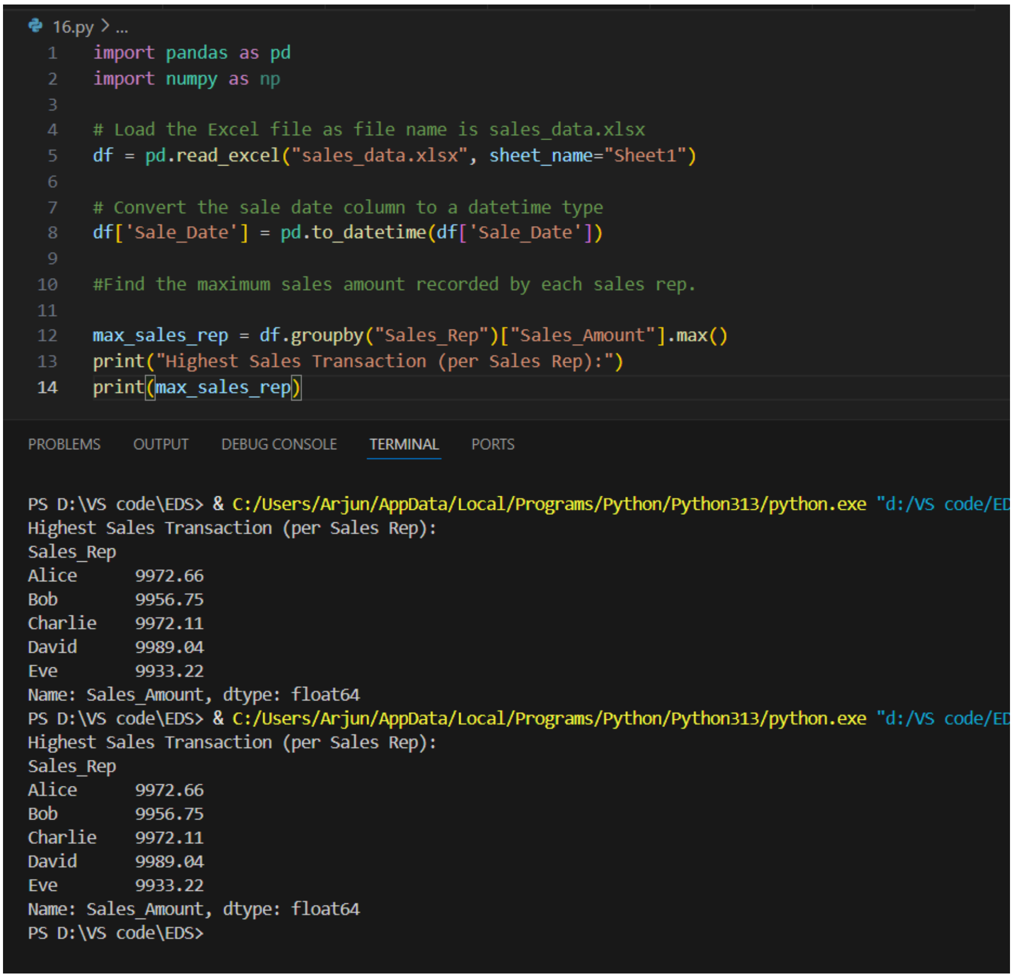
15. Count the number of transactions for each region and sales representative.



```python
import pandas as pd
import numpy as np
# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Count the number of transactions for each combination of region and sales rep (using the “Region_and_Sales_Rep” column).

transaction_count = df["Region_and_Sales_Rep"].value_counts()
print("Transaction Count by Region_and_Sales_Rep:")
print(transaction_count)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS/15.py"
Transaction Count by Region_and_Sales_Rep:
Region_and_Sales_Rep
North-Eve        64
East-Bob         60
East-David       59
South-David      56
North-David      56
East-Eve         55
North-Charlie    55
West-Bob         54
East-Alice       52
West-David       51
South-Bob        50
North-Alice      48
West-Alice       47
West-Eve         47
South-Alice      45
West-Charlie     45
North-Bob        44
South-Eve        43
East-Charlie     37
South-Charlie    32
```

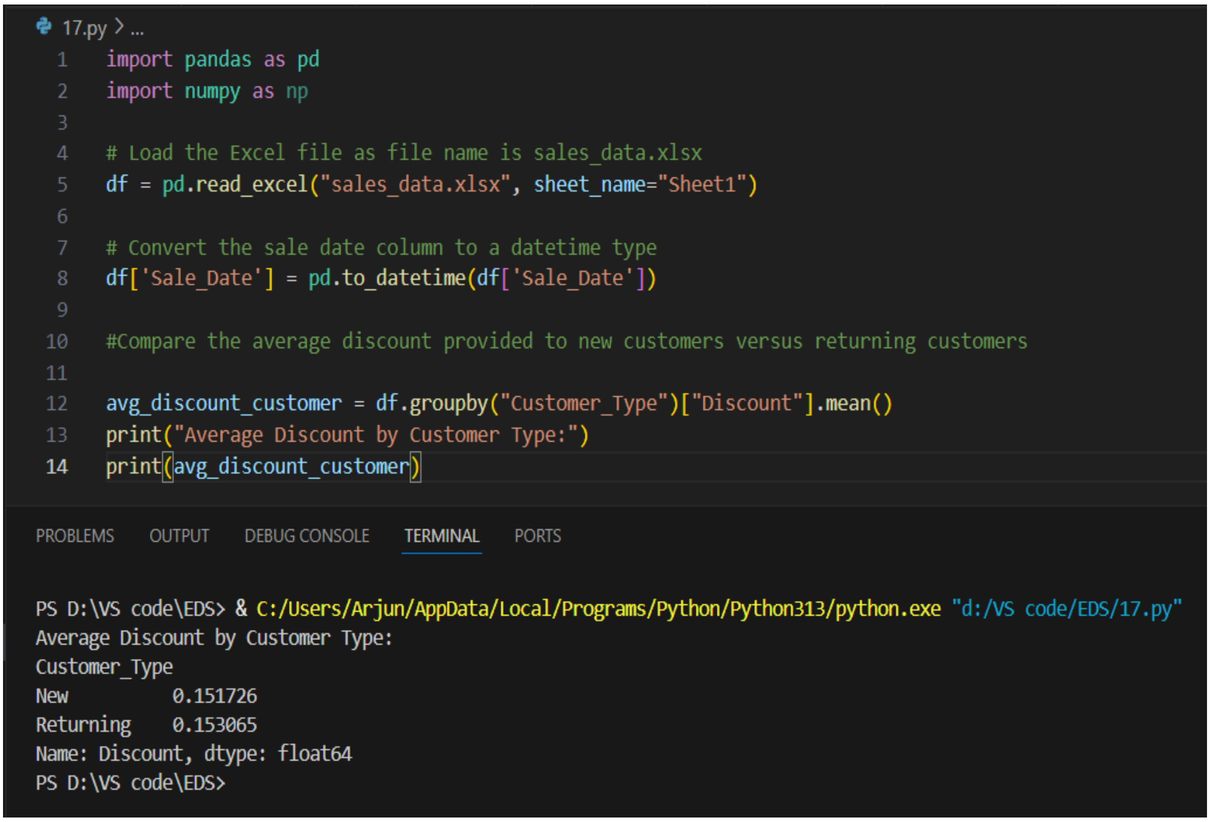
16. Find the highest single sales transaction for each sales representative.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Find the maximum sales amount recorded by each sales rep.

max_sales_rep = df.groupby("Sales_Rep")["Sales_Amount"].max()
print("Highest Sales Transaction (per Sales Rep):")
print(max_sales_rep)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/ED
Highest Sales Transaction (per Sales Rep):
Sales_Rep
Alice      9972.66
Bob        9956.75
Charlie    9972.11
David      9989.04
Eve        9933.22
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/ED
Highest Sales Transaction (per Sales Rep):
Sales_Rep
Alice      9972.66
Bob        9956.75
Charlie    9972.11
David      9989.04
Eve        9933.22
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS>
```

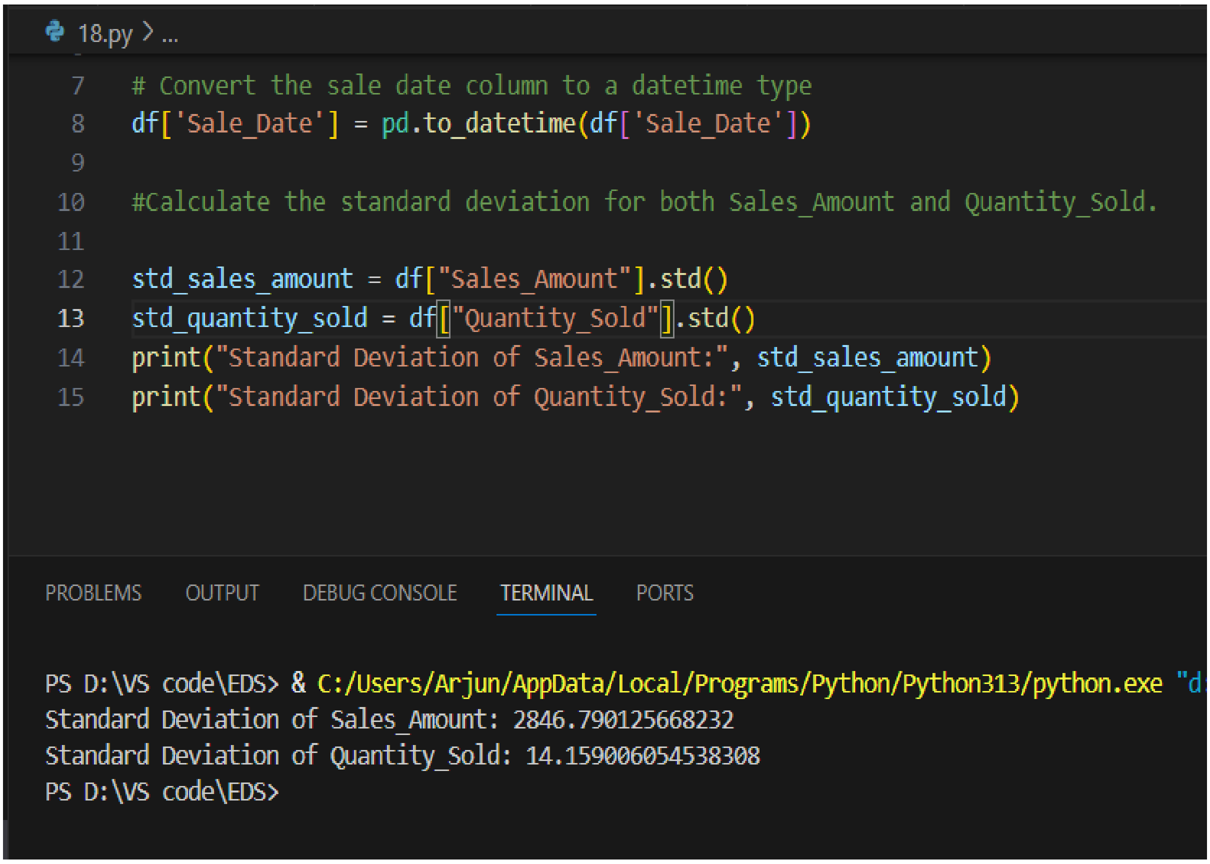
17. Compare the average discount for new vs. returning customers.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Compare the average discount provided to new customers versus returning customers

avg_discount_customer = df.groupby("Customer_Type")["Discount"].mean()
print("Average Discount by Customer Type:")
print(avg_discount_customer)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS/17.py"
Average Discount by Customer Type:
Customer_Type
New          0.151726
Returning    0.153065
Name: Discount, dtype: float64
PS D:\VS code\EDS>
```

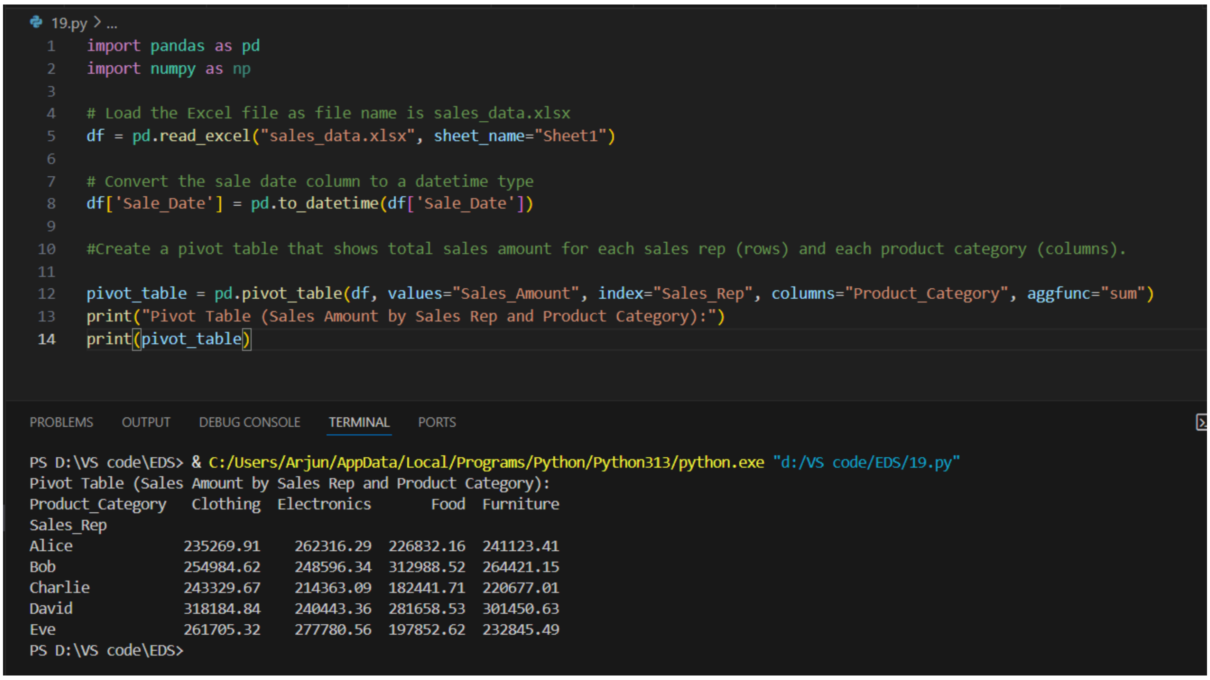
18. Calculate the standard deviation of sales amounts and quantities sold.



```python
# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Calculate the standard deviation for both Sales_Amount and Quantity_Sold.

std_sales_amount = df["Sales_Amount"].std()
std_quantity_sold = df["Quantity_Sold"].std()
print("Standard Deviation of Sales_Amount:", std_sales_amount)
print("Standard Deviation of Quantity_Sold:", std_quantity_sold)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d
Standard Deviation of Sales_Amount: 2846.790125668232
Standard Deviation of Quantity_Sold: 14.159006054538308
PS D:\VS code\EDS>
```

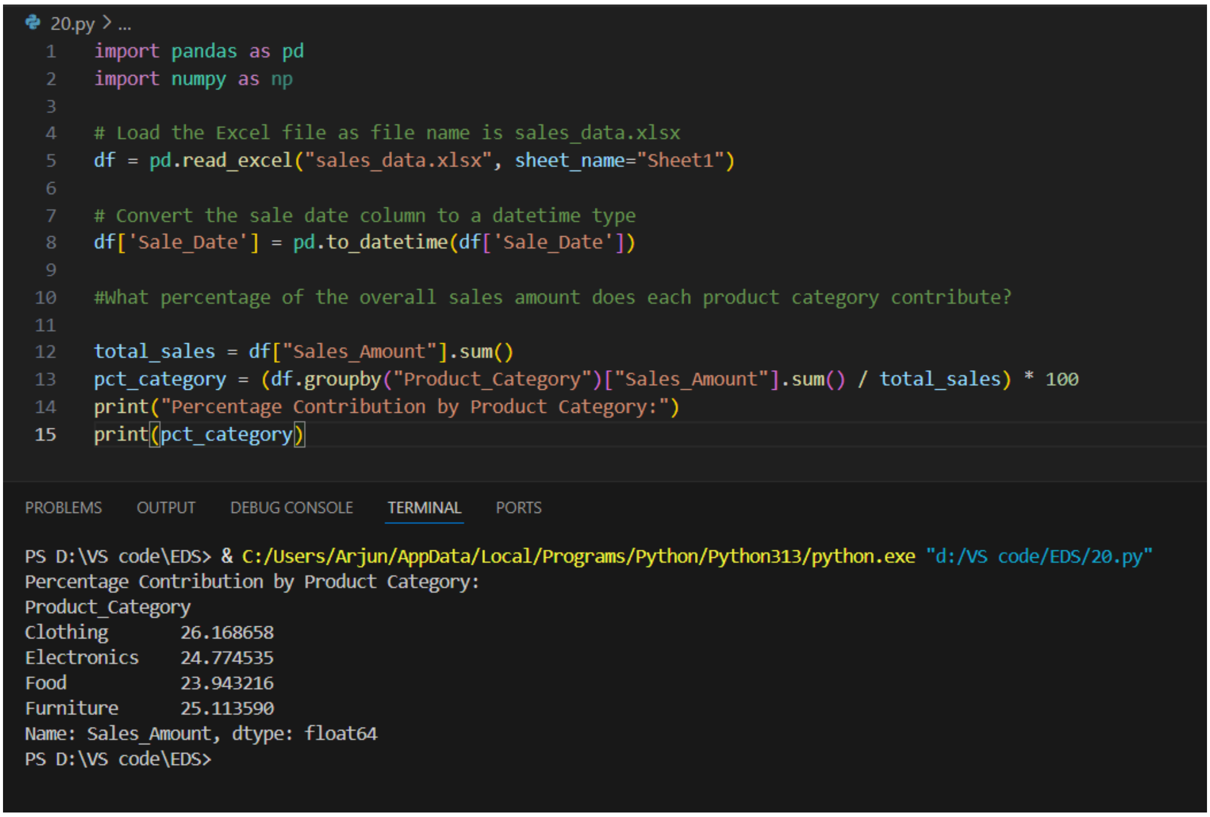
19. Create a table showing total sales by sales representative and product category.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#Create a pivot table that shows total sales amount for each sales rep (rows) and each product category (columns).

pivot_table = pd.pivot_table(df, values="Sales_Amount", index="Sales_Rep", columns="Product_Category", aggfunc="sum")
print("Pivot Table (Sales Amount by Sales Rep and Product Category):")
print(pivot_table)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS/19.py"
Pivot Table (Sales Amount by Sales Rep and Product Category):
Product_Category   Clothing  Electronics       Food  Furniture
Sales_Rep
Alice             235269.91    262316.29  226832.16  241123.41
Bob               254984.62    248596.34  312988.52  264421.15
Charlie           243329.67    214363.09  182441.71  220677.01
David             318184.84    240443.36  281658.53  301450.63
Eve               261705.32    277780.56  197852.62  232845.49
PS D:\VS code\EDS>
```

20. Calculate the percentage contribution of each product category to total sales.



```python
import pandas as pd
import numpy as np

# Load the Excel file as file name is sales_data.xlsx
df = pd.read_excel("sales_data.xlsx", sheet_name="Sheet1")

# Convert the sale date column to a datetime type
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])

#What percentage of the overall sales amount does each product category contribute?

total_sales = df["Sales_Amount"].sum()
pct_category = (df.groupby("Product_Category")["Sales_Amount"].sum() / total_sales) * 100
print("Percentage Contribution by Product Category:")
print(pct_category)
```

```
PS D:\VS code\EDS> & C:/Users/Arjun/AppData/Local/Programs/Python/Python313/python.exe "d:/VS code/EDS/20.py"
Percentage Contribution by Product Category:
Product_Category
Clothing       26.168658
Electronics    24.774535
Food           23.943216
Furniture      25.113590
Name: Sales_Amount, dtype: float64
PS D:\VS code\EDS>
```