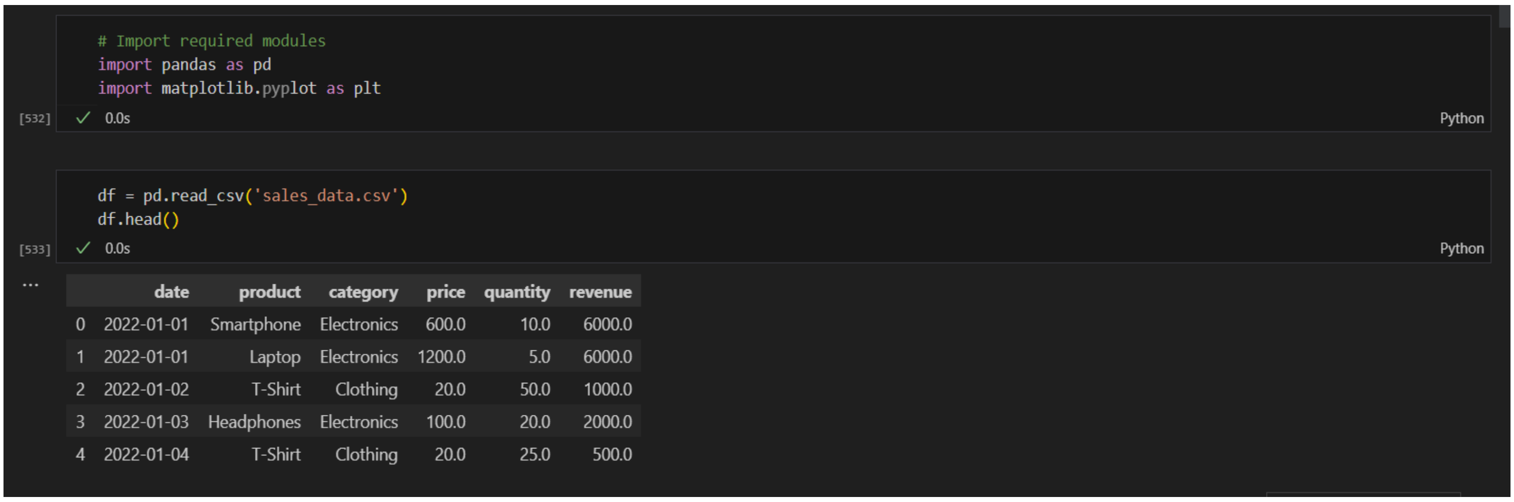
Data Analyst Internship Report

This report summarizes the results of a data analyst internship tasked with analyzing sales data. Through data cleaning, exploratory analysis and statistical techniques, trends and relationships were identified. The report includes insights and actionable recommendations to help improve sales performance.

Reading the dataset



```
# Import required modules
import pandas as pd
import matplotlib.pyplot as plt
```

[532] ✓ 0.0s Python

```
df = pd.read_csv('sales_data.csv')
df.head()
```

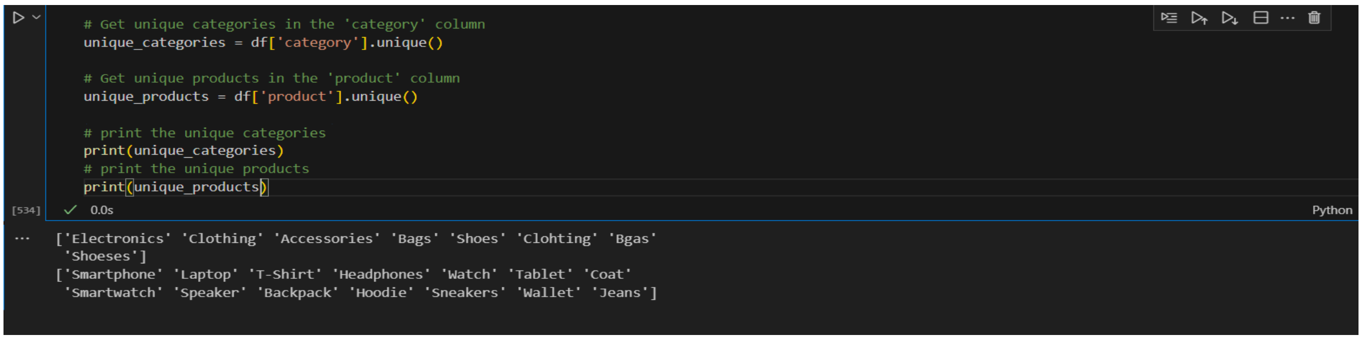
[533] ✓ 0.0s Python

	date	product	category	price	quantity	revenue
0	2022-01-01	Smartphone	Electronics	600.0	10.0	6000.0
1	2022-01-01	Laptop	Electronics	1200.0	5.0	6000.0
2	2022-01-02	T-Shirt	Clothing	20.0	50.0	1000.0
3	2022-01-03	Headphones	Electronics	100.0	20.0	2000.0
4	2022-01-04	T-Shirt	Clothing	20.0	25.0	500.0

Data Cleaning and preprocessing

Replacing misspelled categories with their right ones.

- Before Replacing

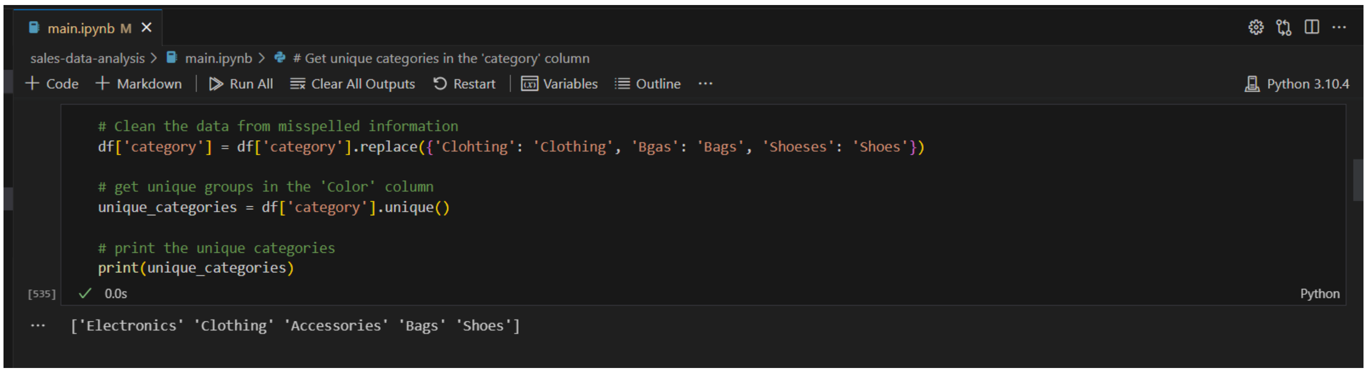


```
# Get unique categories in the 'category' column
unique_categories = df['category'].unique()

# Get unique products in the 'product' column
unique_products = df['product'].unique()

# print the unique categories
print(unique_categories)
# print the unique products
print(unique_products)
```

[534] ✓ 0.0s Python

```
['Electronics' 'Clothing' 'Accessories' 'Bags' 'Shoes' 'Clohting' 'Bgas'
 'Shoeses']
['Smartphone' 'Laptop' 'T-Shirt' 'Headphones' 'Watch' 'Tablet' 'Coat'
 'Smartwatch' 'Speaker' 'Backpack' 'Hoodie' 'Sneakers' 'Wallet' 'Jeans']
```

- After Replacing



main.ipynb M
sales-data-analysis > main.ipynb > # Get unique categories in the 'category' column
+ Code + Markdown | Run All Clear All Outputs Restart | Variables Outline
Python 3.10.4

```
# Clean the data from misspelled information
df['category'] = df['category'].replace({'Clohting': 'Clothing', 'Bgas': 'Bags', 'Shoeses': 'Shoes'})

# get unique groups in the 'Color' column
unique_categories = df['category'].unique()

# print the unique categories
print(unique_categories)
```

[535] ✓ 0.0s Python

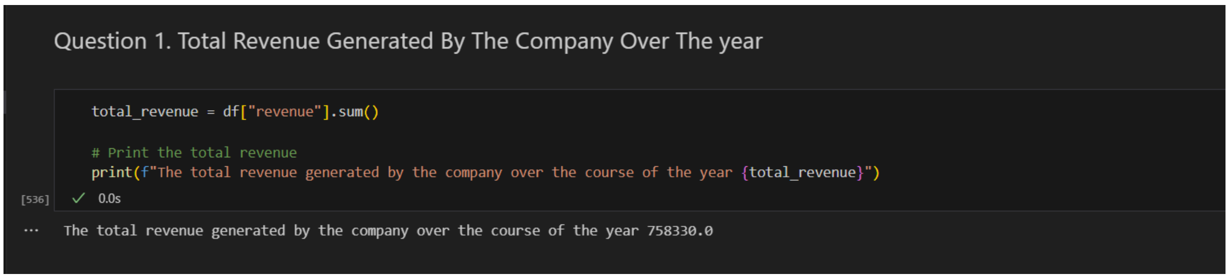
```
['Electronics' 'Clothing' 'Accessories' 'Bags' 'Shoes']
```

Questions

Analyzing the given dataset.

Question 1

Get the total revenue over the year by the cumulative summing of revenue for each record of data.



Question 1. Total Revenue Generated By The Company Over The year

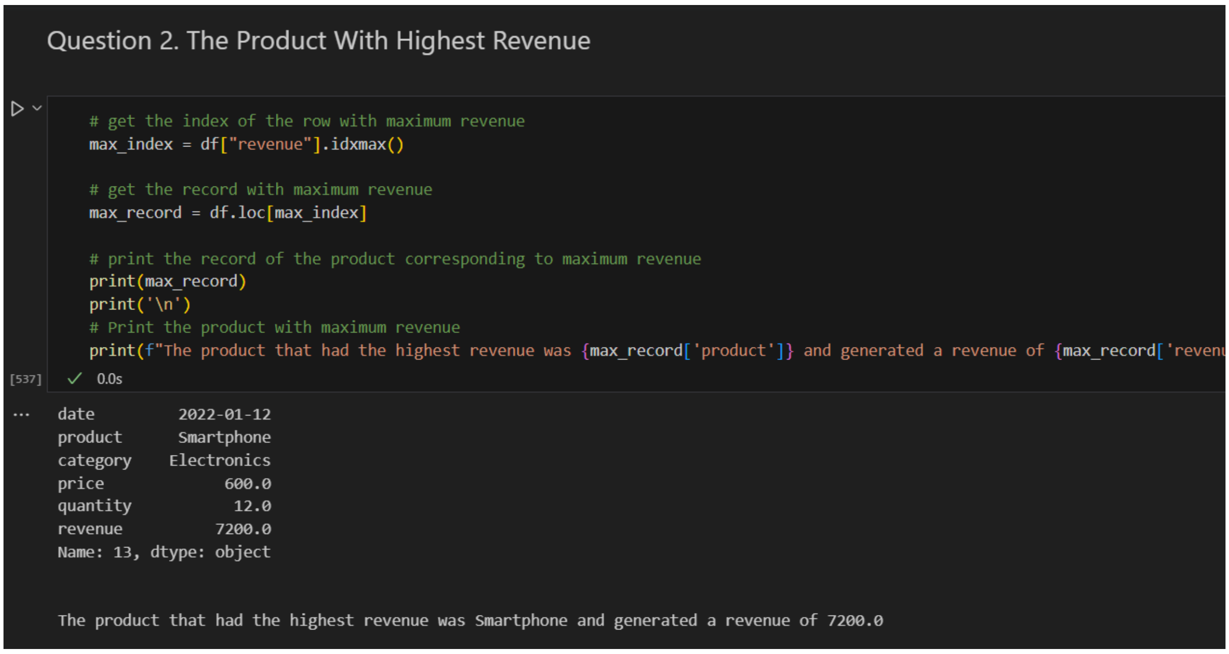
```python
total_revenue = df["revenue"].sum()

# Print the total revenue
print(f"The total revenue generated by the company over the course of the year {total_revenue}")
```

[536] ✓ 0.0s

```
The total revenue generated by the company over the course of the year 758330.0
```

Question 2

To get the product with highest revenue and its corresponding value, you first get the index corresponding to the maximum value of revenue, access the product with highest revenue from the data frame through the "product" column and get the value of revenue from the "revenue" column using the same index previously got.



Question 2. The Product With Highest Revenue

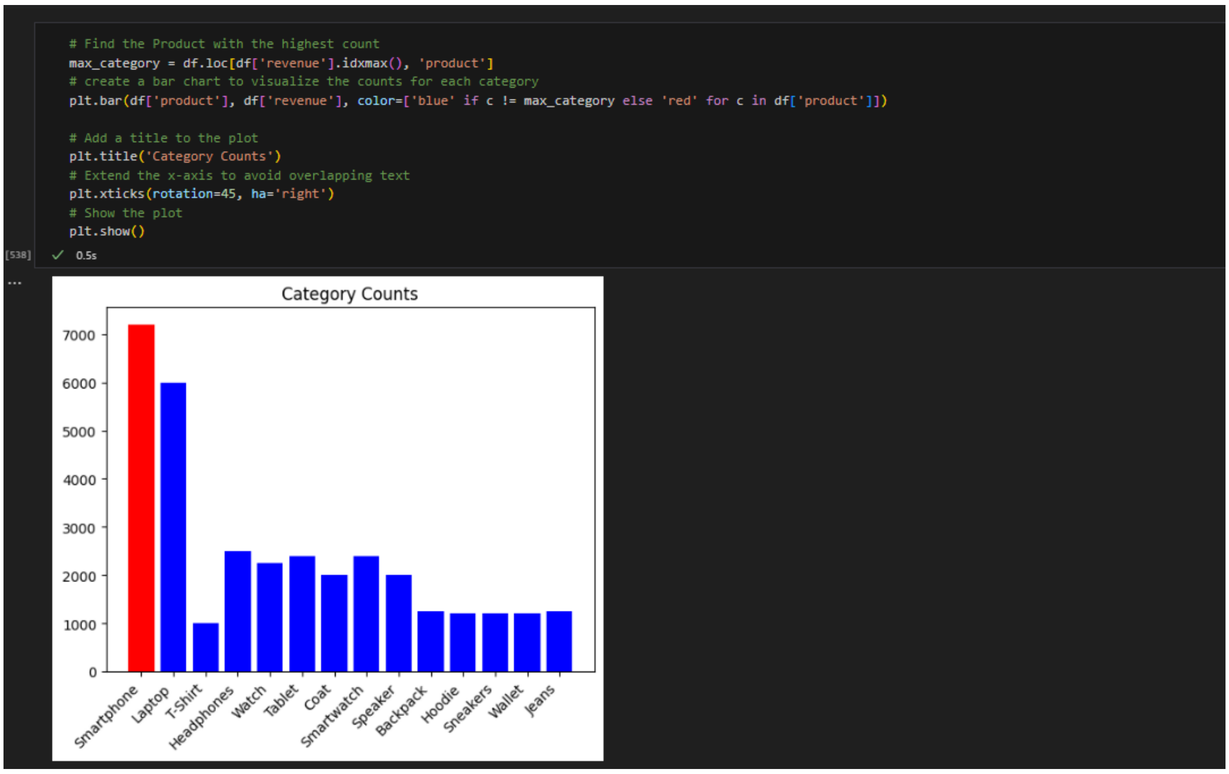
```python
# get the index of the row with maximum revenue
max_index = df["revenue"].idxmax()

# get the record with maximum revenue
max_record = df.loc[max_index]

# print the record of the product corresponding to maximum revenue
print(max_record)
print('\n')
# Print the product with maximum revenue
print(f"The product that had the highest revenue was {max_record['product']} and generated a revenue of {max_record['revenu
```

[537] ✓ 0.0s

```
date          2022-01-12
product       Smartphone
category     Electronics
price              600.0
quantity            12.0
revenue           7200.0
Name: 13, dtype: object


The product that had the highest revenue was Smartphone and generated a revenue of 7200.0
```

The visualization of results is shown below.

- Visualization of results



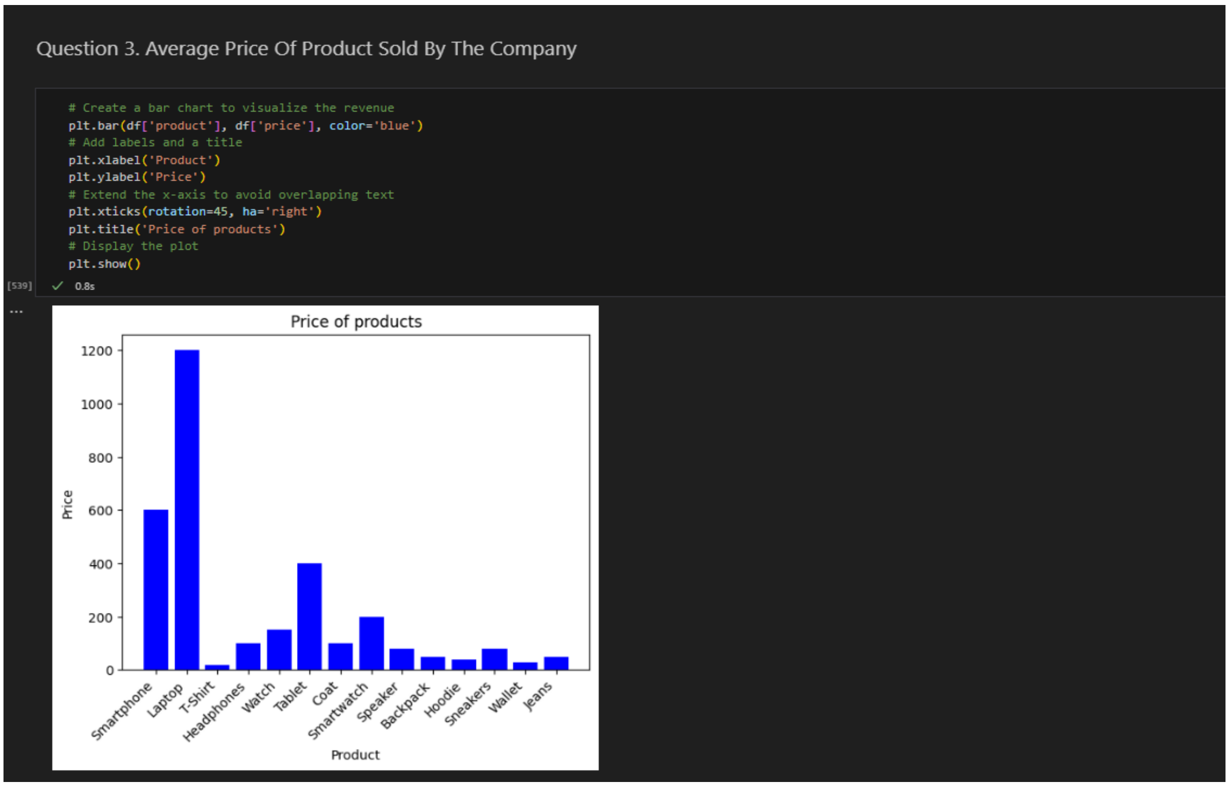
```
# Find the Product with the highest count
max_category = df.loc[df['revenue'].idxmax(), 'product']
# create a bar chart to visualize the counts for each category
plt.bar(df['product'], df['revenue'], color=['blue' if c != max_category else 'red' for c in df['product']])

# Add a title to the plot
plt.title('Category Counts')
# Extend the x-axis to avoid overlapping text
plt.xticks(rotation=45, ha='right')
# Show the plot
plt.show()
```

[538] ✓ 0.5s

Question 3

To get the average price of a product sold, you need to get the mean value for the "price" column of the data frame previously defined.

- The initial distribution of prices of products

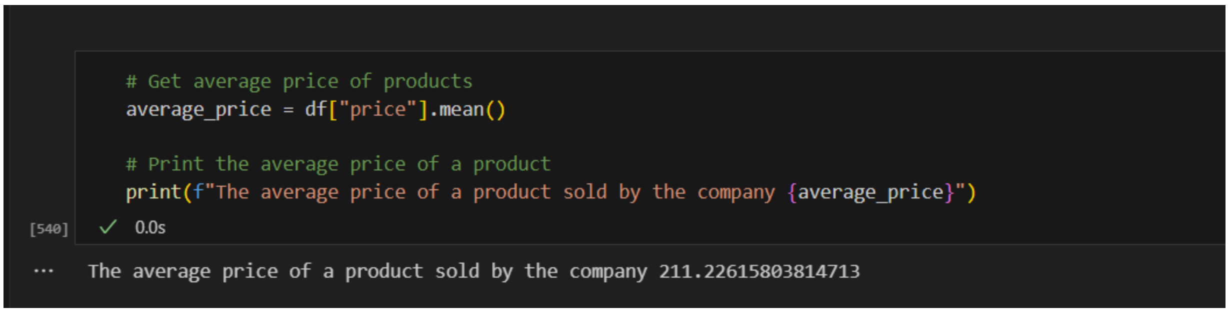


Question 3. Average Price Of Product Sold By The Company

```
# Create a bar chart to visualize the revenue
plt.bar(df['product'], df['price'], color='blue')
# Add labels and a title
plt.xlabel('Product')
plt.ylabel('Price')
# Extend the x-axis to avoid overlapping text
plt.xticks(rotation=45, ha='right')
plt.title('Price of products')
# Display the plot
plt.show()
```

[539] ✓ 0.8s

- Get the mean of the "prices" column

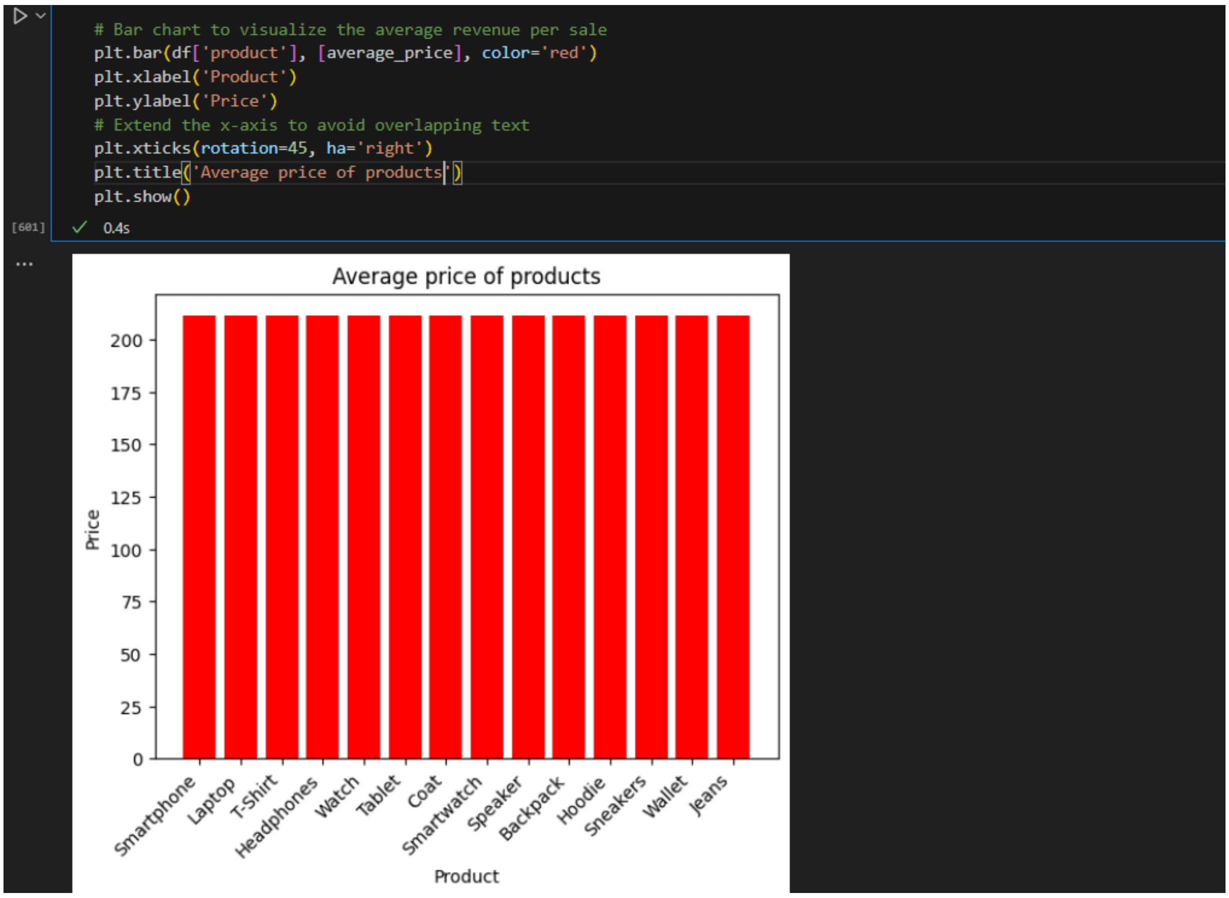


```
# Get average price of products
average_price = df["price"].mean()

# Print the average price of a product
print(f"The average price of a product sold by the company {average_price}")
```

```
[540]  ✓ 0.0s
```

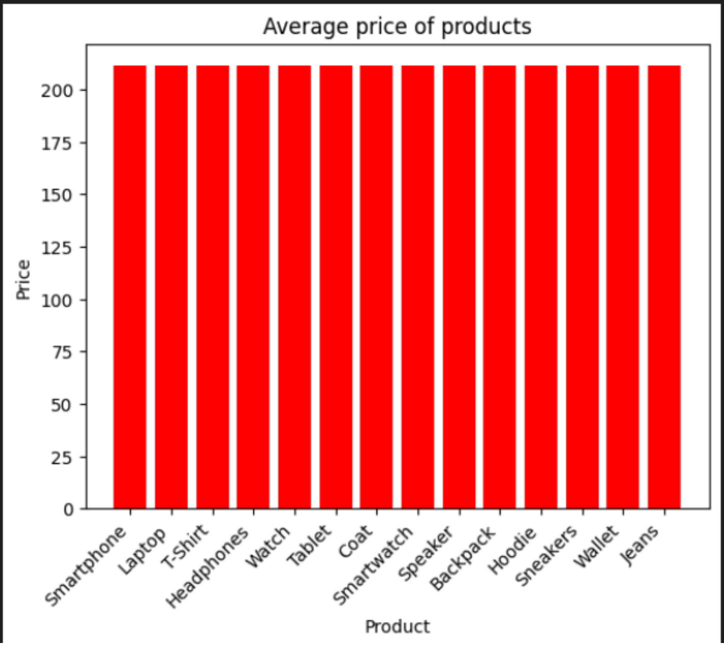
```
The average price of a product sold by the company 211.22615803814713
```

- Visualization for the mean of prices



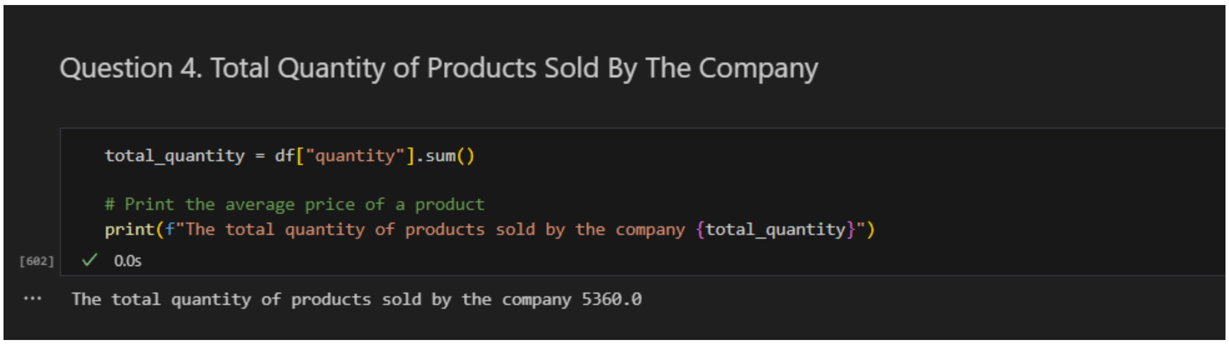
```
# Bar chart to visualize the average revenue per sale
plt.bar(df['product'], [average_price], color='red')
plt.xlabel('Product')
plt.ylabel('Price')
# Extend the x-axis to avoid overlapping text
plt.xticks(rotation=45, ha='right')
plt.title('Average price of products')
plt.show()
```

```
[601]  ✓ 0.4s
```



Question 4

To get the total quantity of products sold, you need to get the sum of the "quantity" column of the data frame previously defined.



Question 4. Total Quantity of Products Sold By The Company

```python
total_quantity = df["quantity"].sum()

# Print the average price of a product
print(f"The total quantity of products sold by the company {total_quantity}")
```

[602] ✓ 0.0s

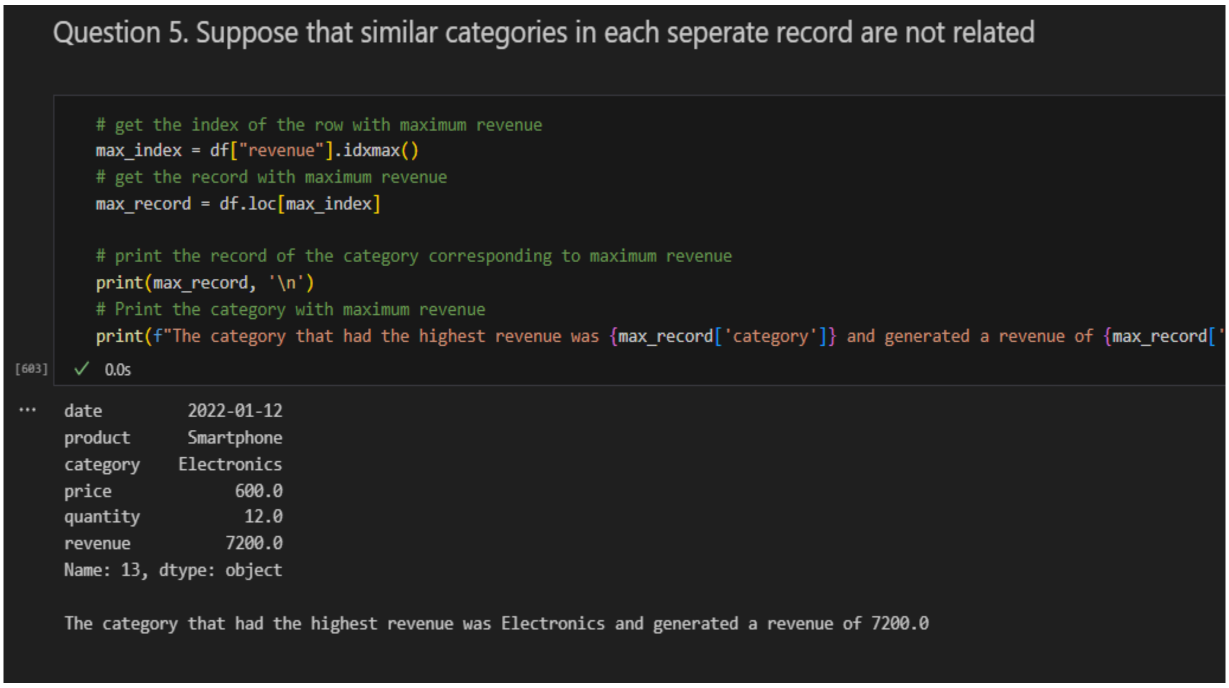
```
The total quantity of products sold by the company 5360.0
```

Question 5

To get the category with highest revenue there were 2 concepts, to either deal with the same categories of different records separately or to combine the revenues of different records having the same category by summing it (more logical).

- First Approach

We first get the index of row with highest revenue, we then use that index to get the whole record or row of data, where we can easily get the category from the "category" column and the corresponding revenue from the "revenue" column.



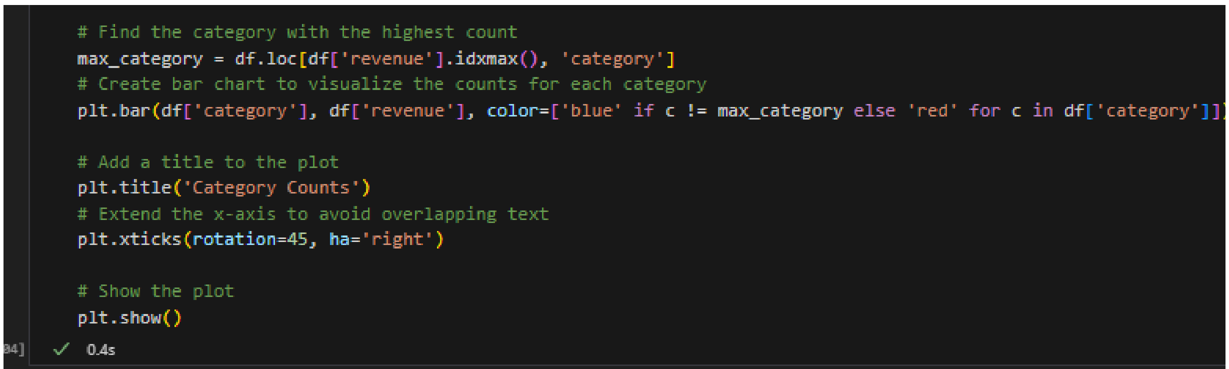
Question 5. Suppose that similar categories in each seperate record are not related

```python
# get the index of the row with maximum revenue
max_index = df["revenue"].idxmax()
# get the record with maximum revenue
max_record = df.loc[max_index]

# print the record of the category corresponding to maximum revenue
print(max_record, '\n')
# Print the category with maximum revenue
print(f"The category that had the highest revenue was {max_record['category']} and generated a revenue of {max_record['
```

[603] ✓ 0.0s

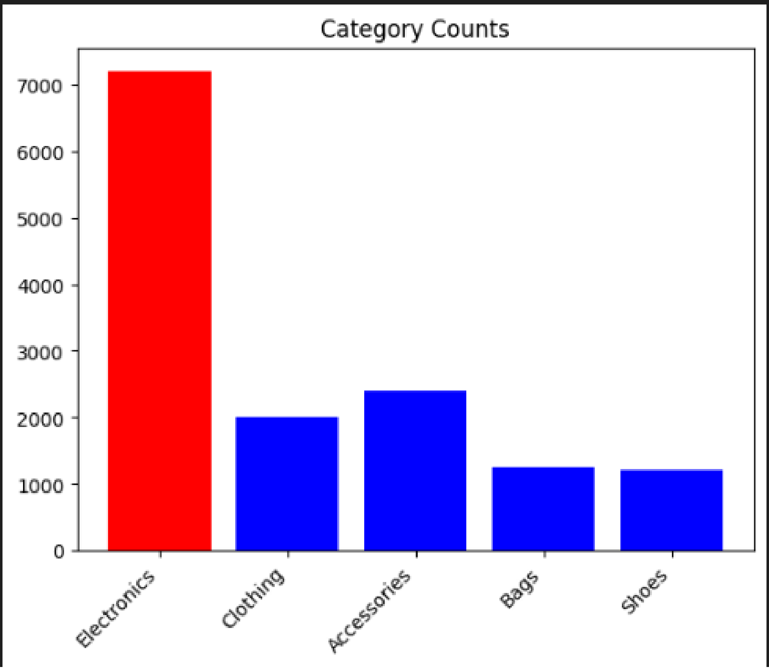
```
date          2022-01-12
product       Smartphone
category     Electronics
price              600.0
quantity            12.0
revenue           7200.0
Name: 13, dtype: object

The category that had the highest revenue was Electronics and generated a revenue of 7200.0
```

The visualization is shown below.



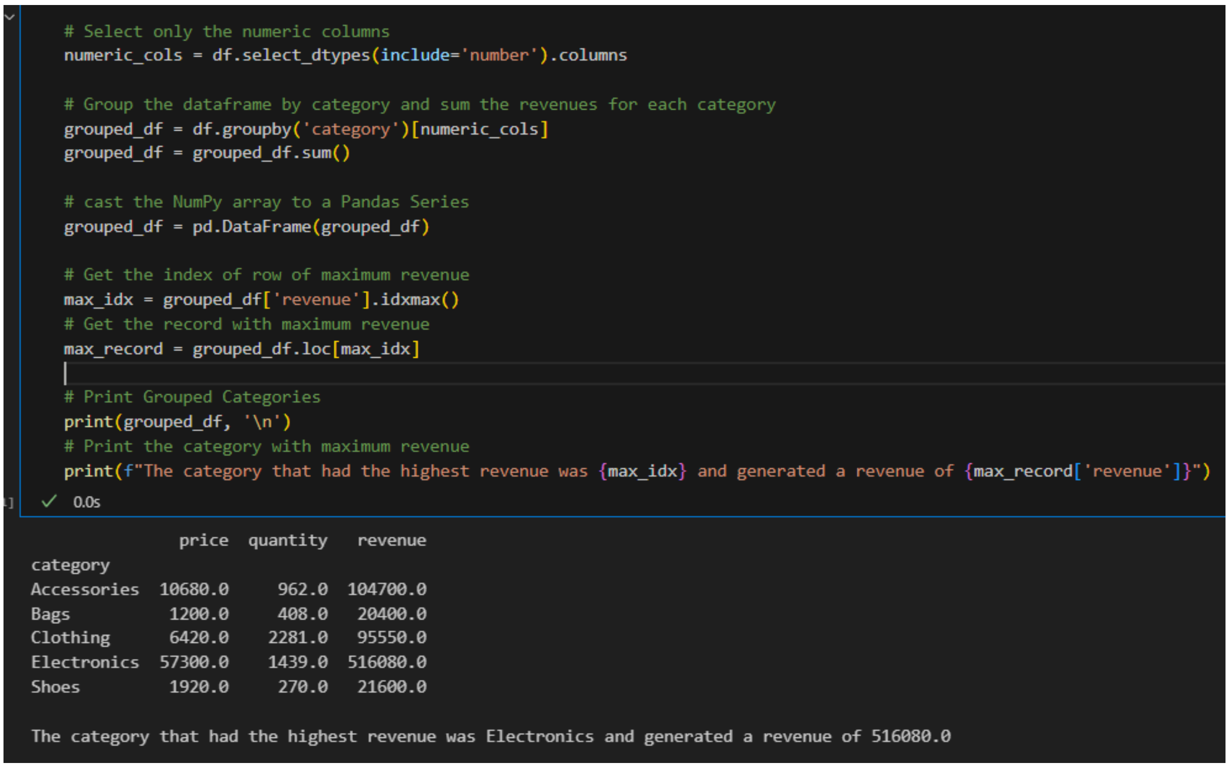
```python
# Find the category with the highest count
max_category = df.loc[df['revenue'].idxmax(), 'category']
# Create bar chart to visualize the counts for each category
plt.bar(df['category'], df['revenue'], color=['blue' if c != max_category else 'red' for c in df['category']]

# Add a title to the plot
plt.title('Category Counts')
# Extend the x-axis to avoid overlapping text
plt.xticks(rotation=45, ha='right')

# Show the plot
plt.show()
```

```
04]  ✓  0.4s
```



- Second Approach

We group the records by the category where we sum the revenues of the same category and then get the index of maximum revenue from all categories and get the name of it from the "category" column as well as the corresponding revenue from the "revenue" column.

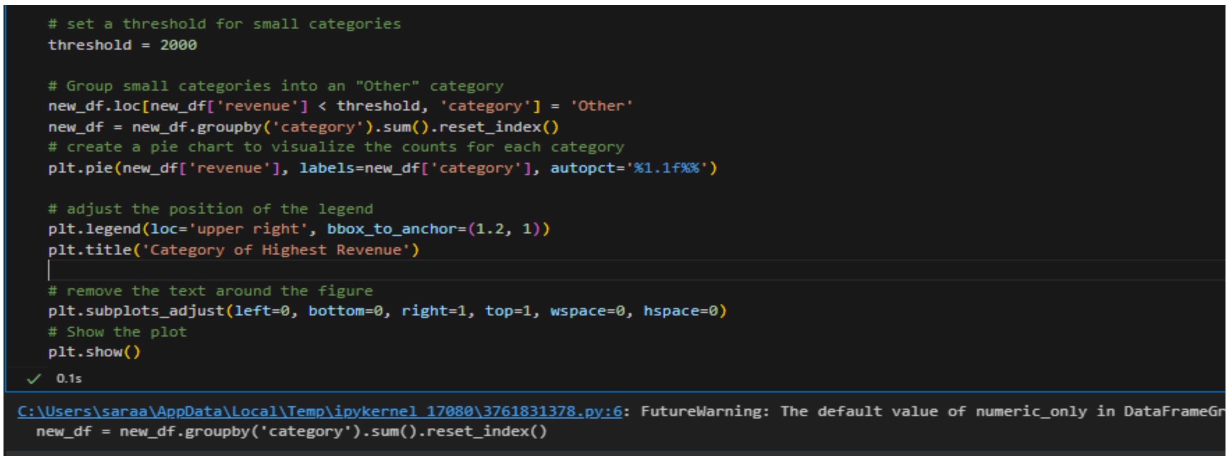


```
# Select only the numeric columns
numeric_cols = df.select_dtypes(include='number').columns

# Group the dataframe by category and sum the revenues for each category
grouped_df = df.groupby('category')[numeric_cols]
grouped_df = grouped_df.sum()

# cast the NumPy array to a Pandas Series
grouped_df = pd.DataFrame(grouped_df)

# Get the index of row of maximum revenue
max_idx = grouped_df['revenue'].idxmax()
# Get the record with maximum revenue
max_record = grouped_df.loc[max_idx]

# Print Grouped Categories
print(grouped_df, '\n')
# Print the category with maximum revenue
print(f"The category that had the highest revenue was {max_idx} and generated a revenue of {max_record['revenue']}")
```

✓ 0.0s

```
               price  quantity   revenue
category
Accessories  10680.0     962.0  104700.0
Bags          1200.0     408.0   20400.0
Clothing      6420.0    2281.0   95550.0
Electronics  57300.0    1439.0  516080.0
Shoes         1920.0     270.0   21600.0

The category that had the highest revenue was Electronics and generated a revenue of 516080.0
```

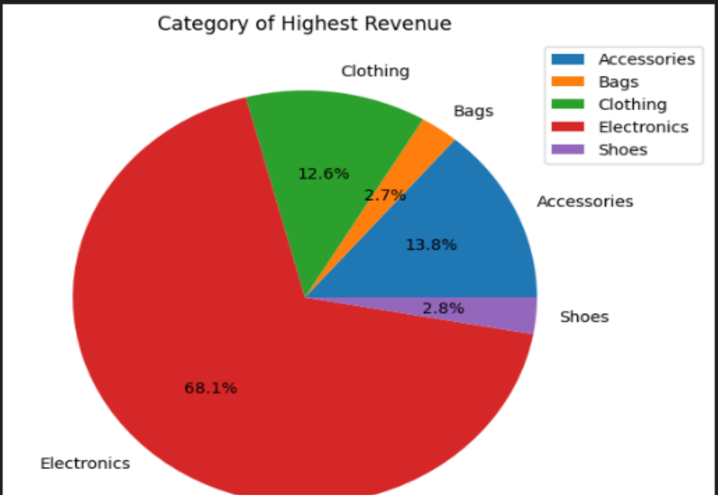
The visualization is shown below.



```
# set a threshold for small categories
threshold = 2000

# Group small categories into an "Other" category
new_df.loc[new_df['revenue'] < threshold, 'category'] = 'Other'
new_df = new_df.groupby('category').sum().reset_index()
# create a pie chart to visualize the counts for each category
plt.pie(new_df['revenue'], labels=new_df['category'], autopct='%1.1f%%')

# adjust the position of the legend
plt.legend(loc='upper right', bbox_to_anchor=(1.2, 1))
plt.title('Category of Highest Revenue')

# remove the text around the figure
plt.subplots_adjust(left=0, bottom=0, right=1, top=1, wspace=0, hspace=0)
# Show the plot
plt.show()
```

✓ 0.1s

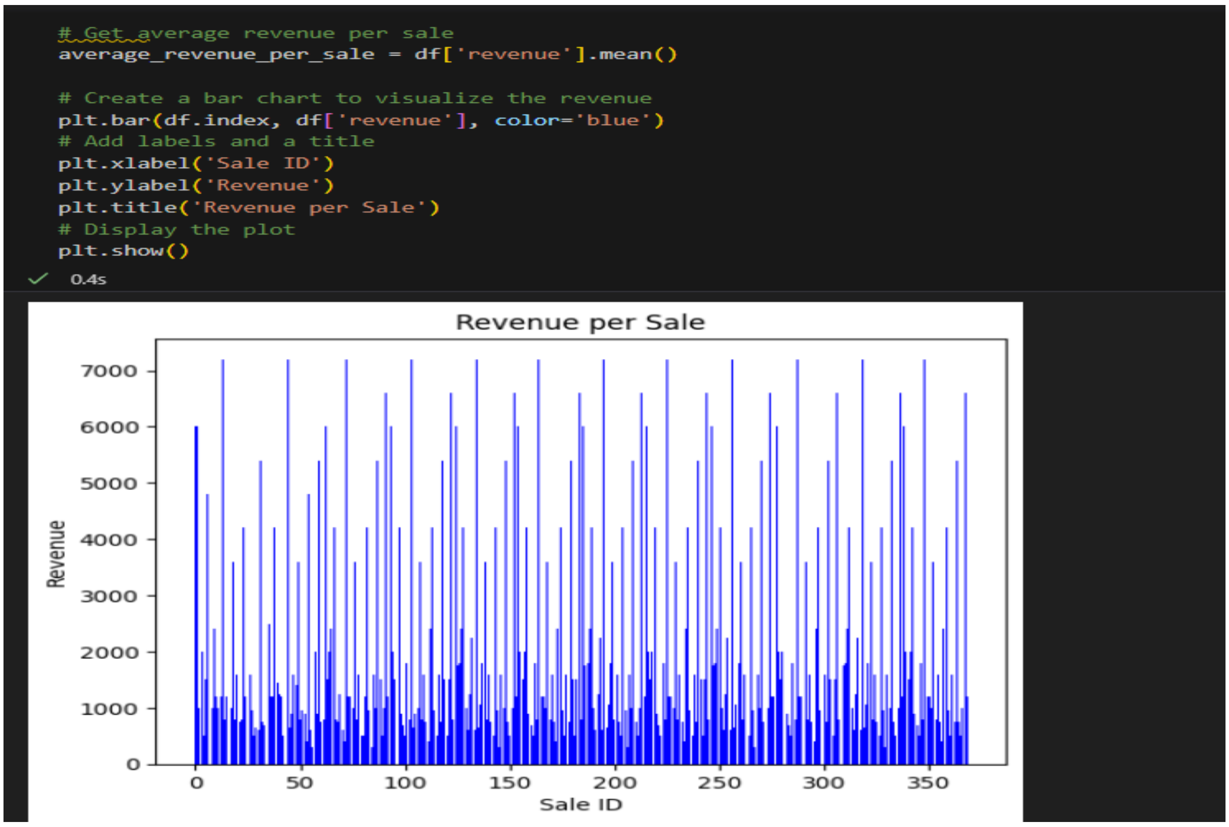
```
C:\Users\saraa\AppData\Local\Temp\ipykernel_17080\3761831378.py:6: FutureWarning: The default value of numeric_only in DataFrameGr
  new_df = new_df.groupby('category').sum().reset_index()
```



Question 6

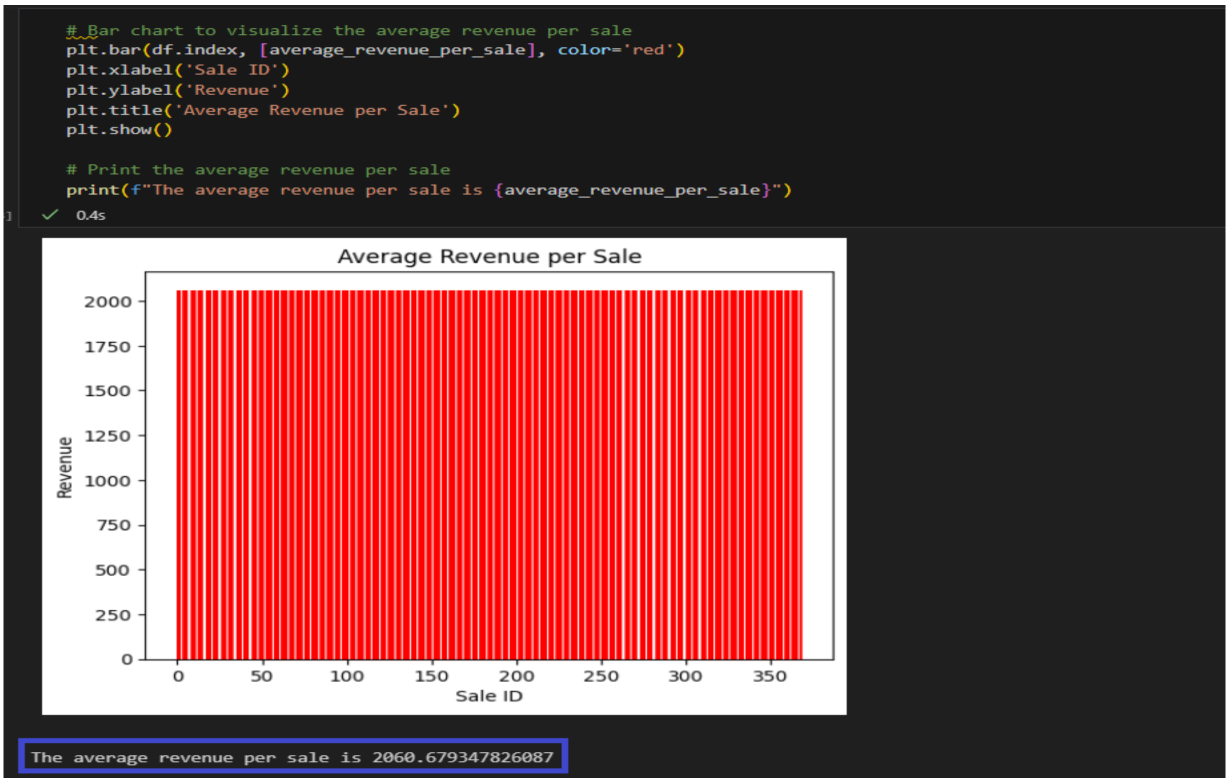
To get the average revenue per sale, you will need to get the mean of the "revenue" column.

- Original revenue per sale



```
# Get average revenue per sale
average_revenue_per_sale = df['revenue'].mean()

# Create a bar chart to visualize the revenue
plt.bar(df.index, df['revenue'], color='blue')
# Add labels and a title
plt.xlabel('Sale ID')
plt.ylabel('Revenue')
plt.title('Revenue per Sale')
# Display the plot
plt.show()
```

- Average revenue per sale

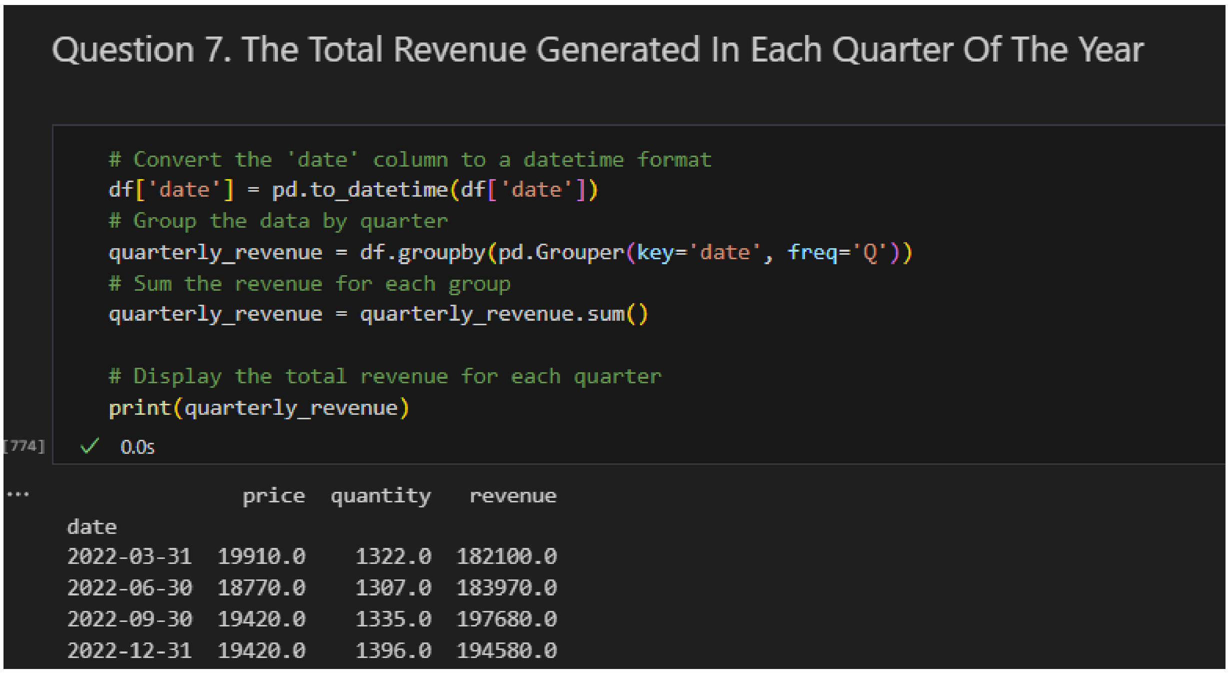


```
# Bar chart to visualize the average revenue per sale
plt.bar(df.index, [average_revenue_per_sale], color='red')
plt.xlabel('Sale ID')
plt.ylabel('Revenue')
plt.title('Average Revenue per Sale')
plt.show()

# Print the average revenue per sale
print(f"The average revenue per sale is {average_revenue_per_sale}")
```

```
The average revenue per sale is 2060.679347826087
```

Question 7

To get the total revenue generated in each quarter of the year you will need to convert the "date" column to a date format to be able to deal with it, then we will group the data by "date" column and will choose a "freq" of value equals to ‘Q’, that means that we will divide the data to four quarters based on the "date" column, and finally we will get sum of revenues for each quarter and print them.



Question 7. The Total Revenue Generated In Each Quarter Of The Year

```python
# Convert the 'date' column to a datetime format
df['date'] = pd.to_datetime(df['date'])
# Group the data by quarter
quarterly_revenue = df.groupby(pd.Grouper(key='date', freq='Q'))
# Sum the revenue for each group
quarterly_revenue = quarterly_revenue.sum()

# Display the total revenue for each quarter
print(quarterly_revenue)
```

[774] ✓ 0.0s

```
              price  quantity   revenue
date
2022-03-31  19910.0    1322.0  182100.0
2022-06-30  18770.0    1307.0  183970.0
2022-09-30  19420.0    1335.0  197680.0
2022-12-31  19420.0    1396.0  194580.0
```

Comments

Based on the above results it is clear that the electronics in general have the highest revenues, so the marketing strategies must be great along with the quality of electronics, while the shoes and bags categories had the least revenues, thus the company has to develop marketing strategies to address these 2 categories or either focus on electronics cause the revenues seem promising.