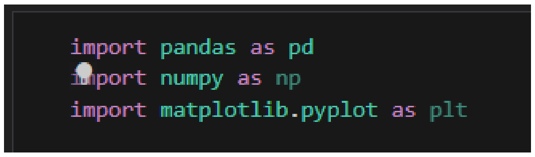
Name: Parikshit Acharya

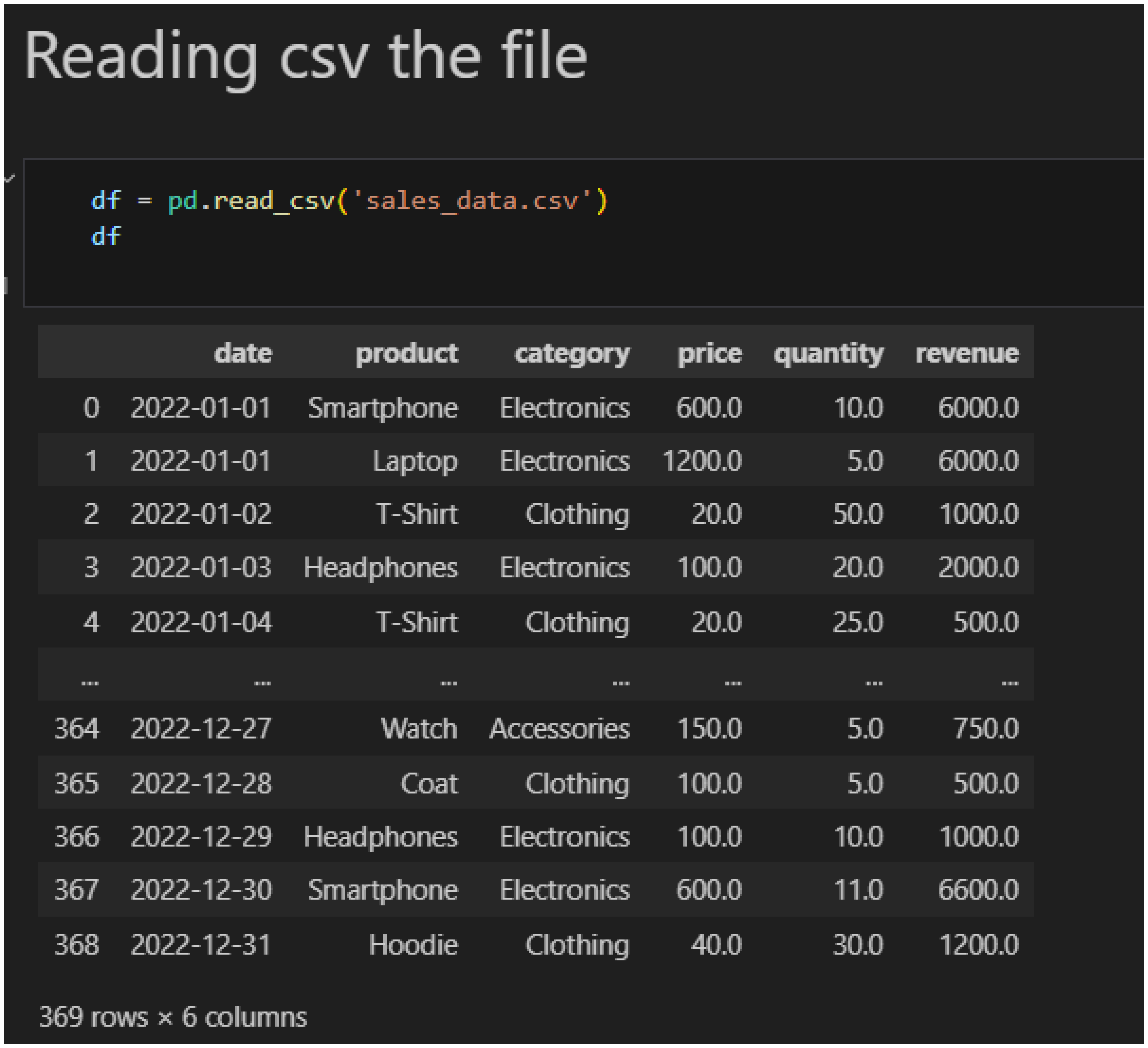
Table of Contents

Importing libraries required.



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Reading CSV



Reading csv the file

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

	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
...	...	...	...	...	...	...
364	2022-12-27	Watch	Accessories	150.0	5.0	750.0
365	2022-12-28	Coat	Clothing	100.0	5.0	500.0
366	2022-12-29	Headphones	Electronics	100.0	10.0	1000.0
367	2022-12-30	Smartphone	Electronics	600.0	11.0	6600.0
368	2022-12-31	Hoodie	Clothing	40.0	30.0	1200.0

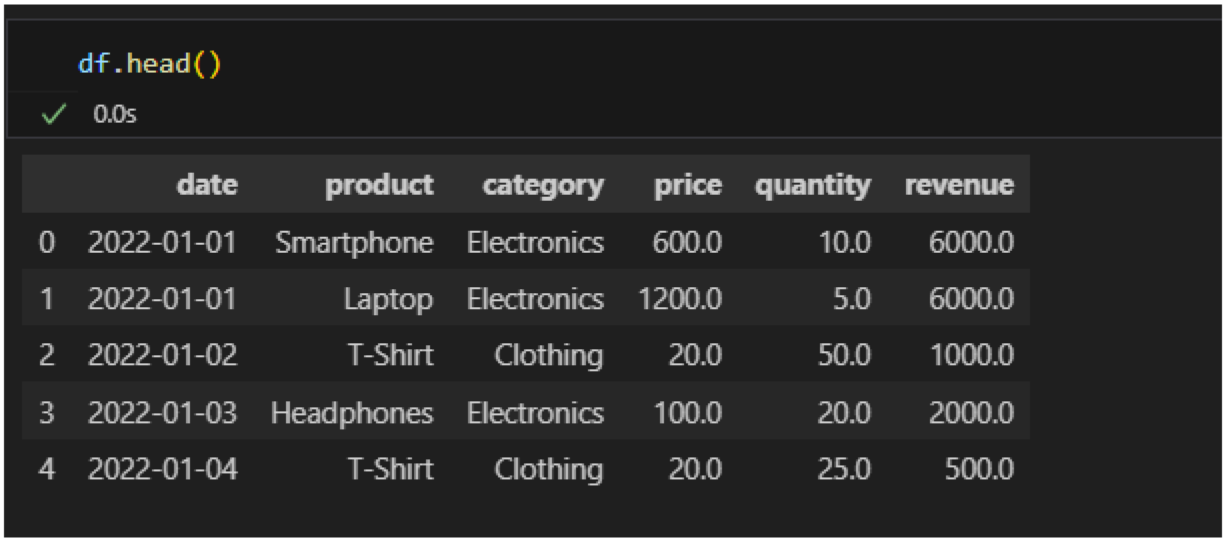
369 rows × 6 columns

Data Exploration / findings

Findings

Here I have a uncleaned csv file with 6 columns and 369 rows with missing rows, duplicated rows and with missing value. Date, product, category is of object data type and price, quantity and revenue is of float data type. I need to analyze data after cleaning and exploring the data presented in csv files.

Checking head

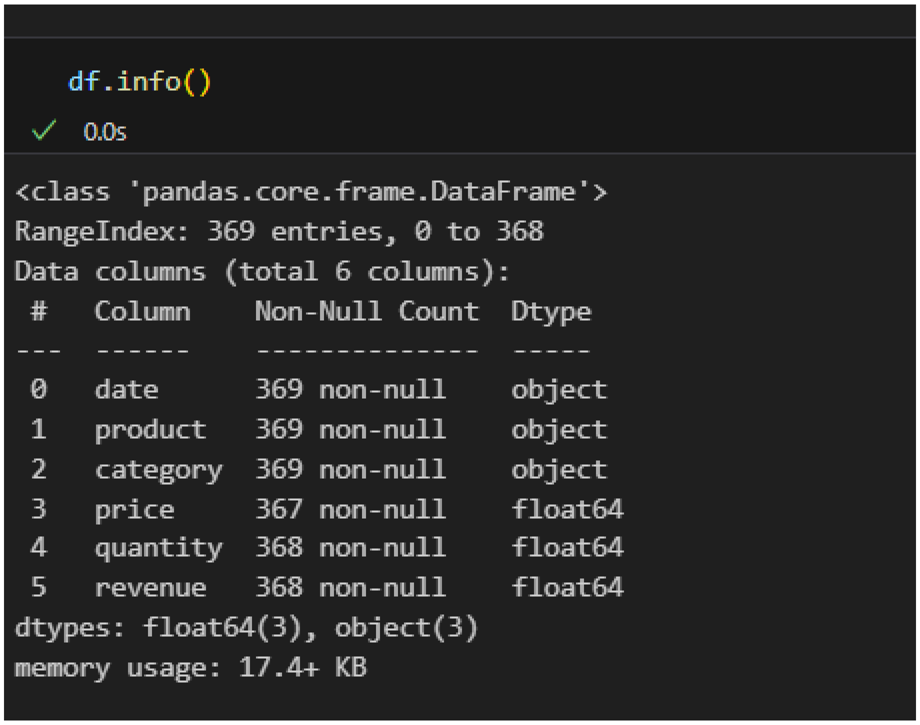


```
df.head()
```

✓ 0.0s

```
         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
```

Checking information

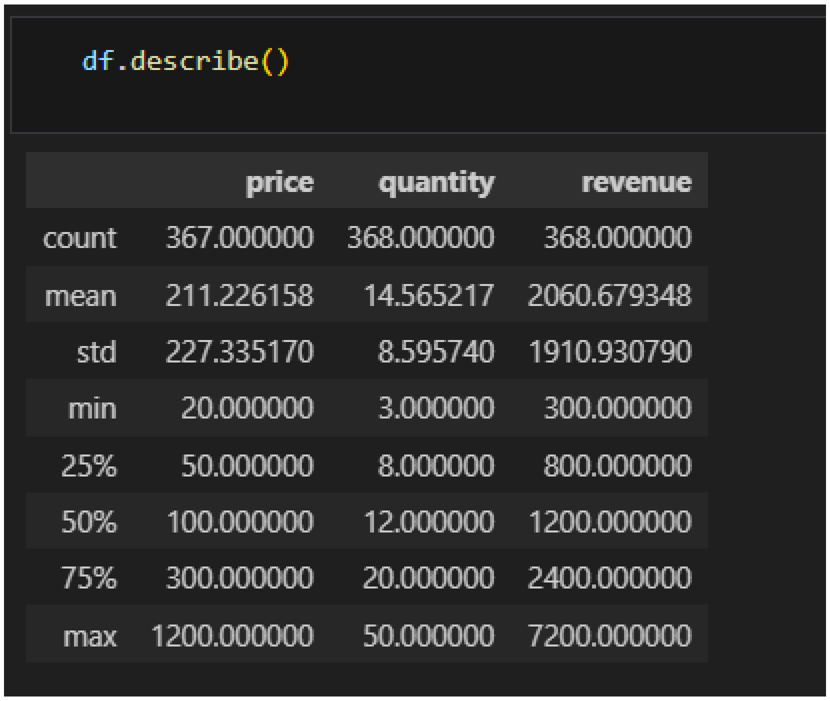


```
df.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 369 entries, 0 to 368
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   date      369 non-null    object 
 1   product   369 non-null    object 
 2   category  369 non-null    object 
 3   price     367 non-null    float64
 4   quantity  368 non-null    float64
 5   revenue   368 non-null    float64
dtypes: float64(3), object(3)
memory usage: 17.4+ KB
```

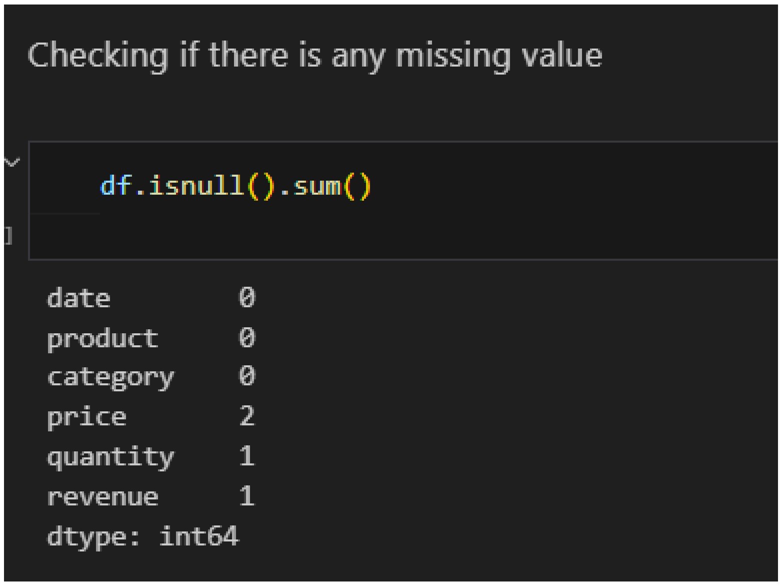
Describing the file dataframe



```
df.describe()
```

	price	quantity	revenue
count	367.000000	368.000000	368.000000
mean	211.226158	14.565217	2060.679348
std	227.335170	8.595740	1910.930790
min	20.000000	3.000000	300.000000
25%	50.000000	8.000000	800.000000
50%	100.000000	12.000000	1200.000000
75%	300.000000	20.000000	2400.000000
max	1200.000000	50.000000	7200.000000

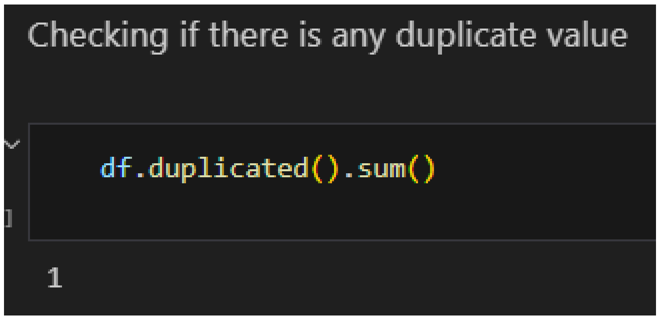
Checking for the missing value


Checking if there is any missing value

```
df.isnull().sum()
```

```
date        0
product     0
category    0
price       2
quantity    1
revenue     1
dtype: int64
```

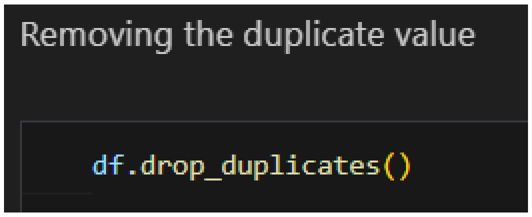
Checking for duplicate


Checking if there is any duplicate value

```
df.duplicated().sum()
```

```
1
```

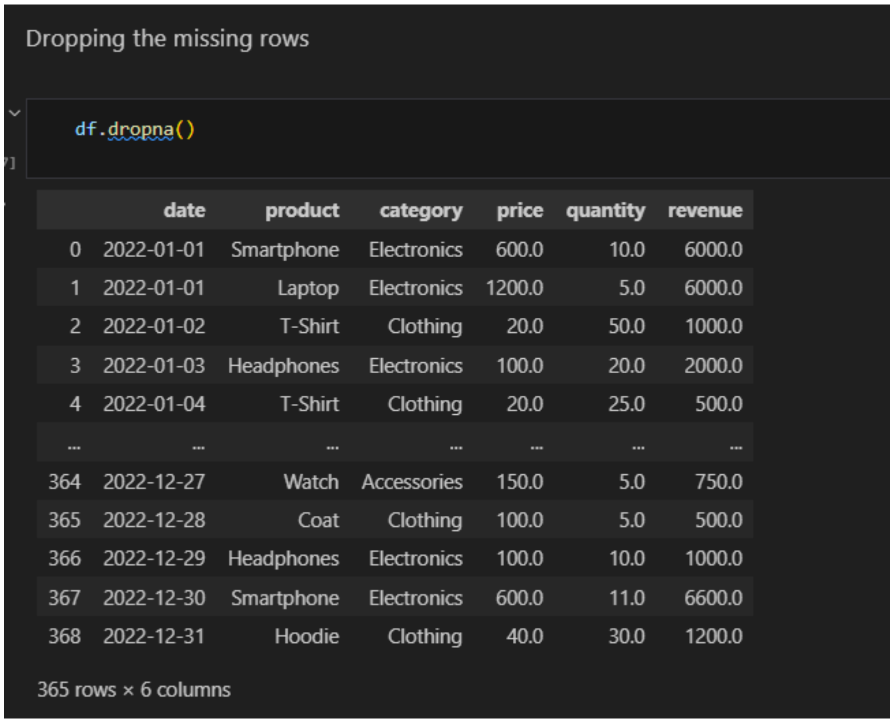
Removing duplicates


Removing the duplicate value

```
df.drop_duplicates()
```

Handling the missing values

Dropping the missing row



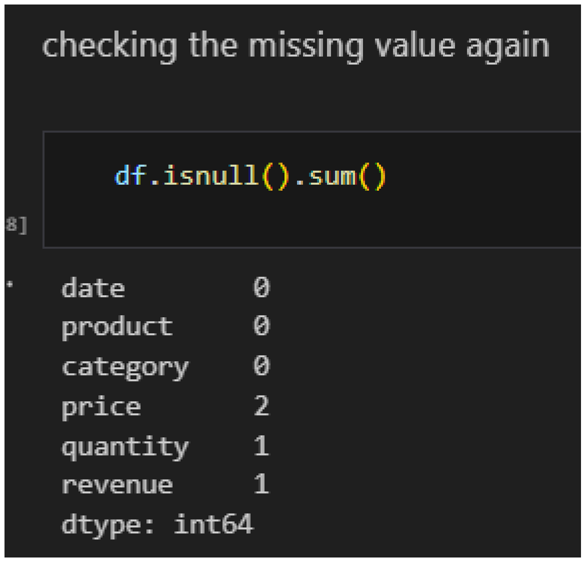
Dropping the missing rows

```
df.dropna()
```

	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
...	...	...	...	...	...	...
364	2022-12-27	Watch	Accessories	150.0	5.0	750.0
365	2022-12-28	Coat	Clothing	100.0	5.0	500.0
366	2022-12-29	Headphones	Electronics	100.0	10.0	1000.0
367	2022-12-30	Smartphone	Electronics	600.0	11.0	6600.0
368	2022-12-31	Hoodie	Clothing	40.0	30.0	1200.0

365 rows × 6 columns

Checking if there is still missing rows

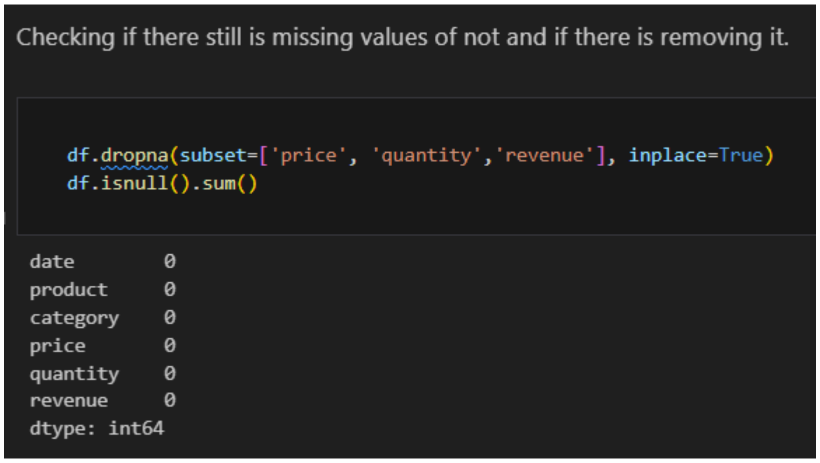


checking the missing value again

```
df.isnull().sum()
```

```
date        0
product     0
category    0
price       2
quantity    1
revenue     1
dtype: int64
```

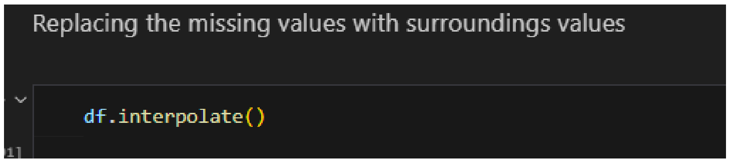
If there is still missing values dropping it and checking again


Checking if there still is missing values of not and if there is removing it.

```
df.dropna(subset=['price', 'quantity','revenue'], inplace=True)
df.isnull().sum()
```

```
date        0
product     0
category    0
price       0
quantity    0
revenue     0
dtype: int64
```

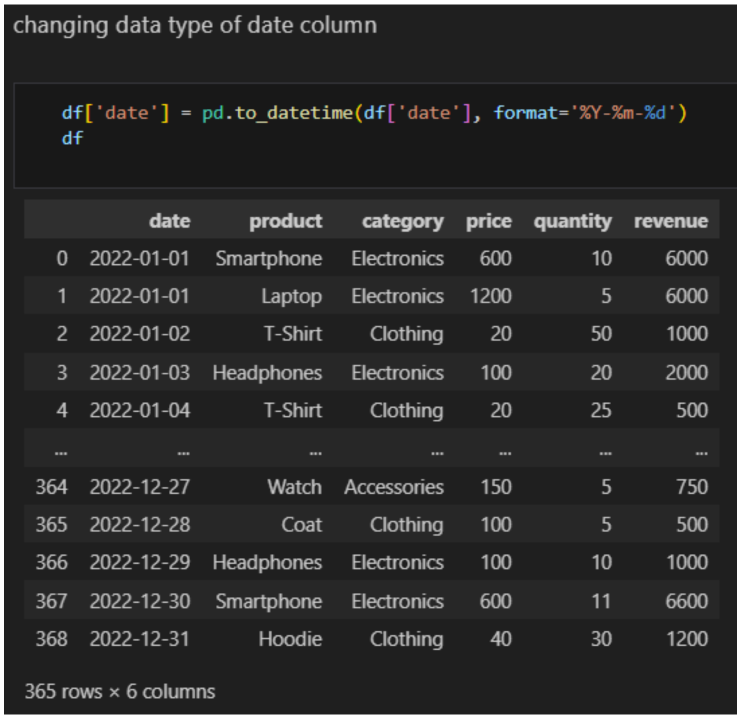
Interpolating the missing values


Replacing the missing values with surroundings values

```
df.interpolate()
```

--Changing the data types of—

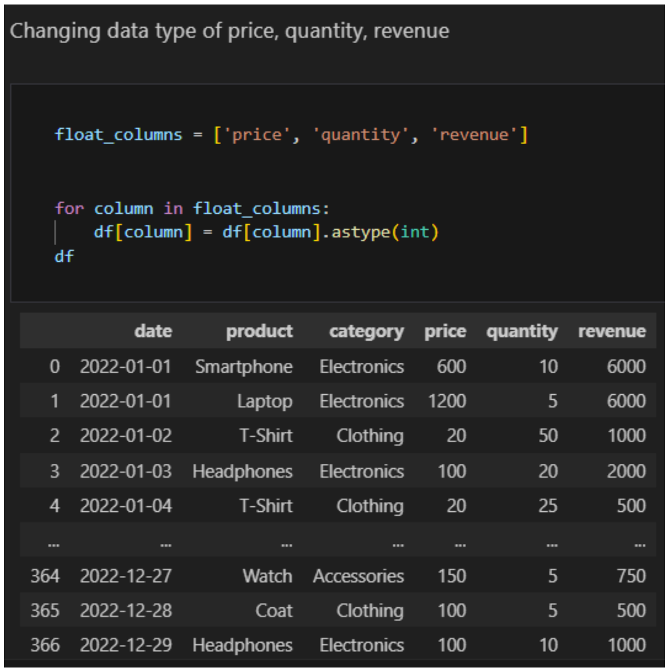
Date


changing data type of date column

```
df['date'] = pd.to_datetime(df['date'], format='%Y-%m-%d')
df
```

```
           date     product     category  price  quantity  revenue
0    2022-01-01  Smartphone  Electronics    600        10     6000
1    2022-01-01      Laptop  Electronics   1200         5     6000
2    2022-01-02     T-Shirt     Clothing     20        50     1000
3    2022-01-03  Headphones  Electronics    100        20     2000
4    2022-01-04     T-Shirt     Clothing     20        25      500
...         ...         ...          ...    ...       ...      ...
364  2022-12-27       Watch  Accessories    150         5      750
365  2022-12-28        Coat     Clothing    100         5      500
366  2022-12-29  Headphones  Electronics    100        10     1000
367  2022-12-30  Smartphone  Electronics    600        11     6600
368  2022-12-31      Hoodie     Clothing     40        30     1200

365 rows × 6 columns
```

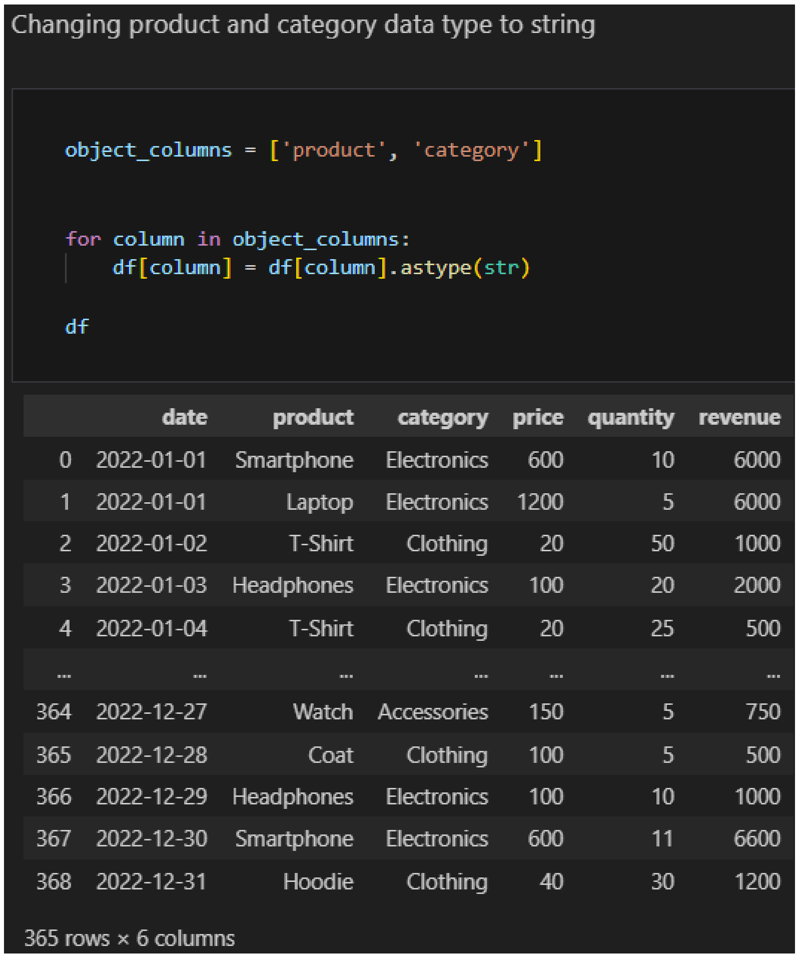
Price, quantity and revenue


Changing data type of price, quantity, revenue

```
float_columns = ['price', 'quantity', 'revenue']


for column in float_columns:
    df[column] = df[column].astype(int)
df
```

```
           date     product     category  price  quantity  revenue
0    2022-01-01  Smartphone  Electronics    600        10     6000
1    2022-01-01      Laptop  Electronics   1200         5     6000
2    2022-01-02     T-Shirt     Clothing     20        50     1000
3    2022-01-03  Headphones  Electronics    100        20     2000
4    2022-01-04     T-Shirt     Clothing     20        25      500
...         ...         ...          ...    ...       ...      ...
364  2022-12-27       Watch  Accessories    150         5      750
365  2022-12-28        Coat     Clothing    100         5      500
366  2022-12-29  Headphones  Electronics    100        10     1000
```

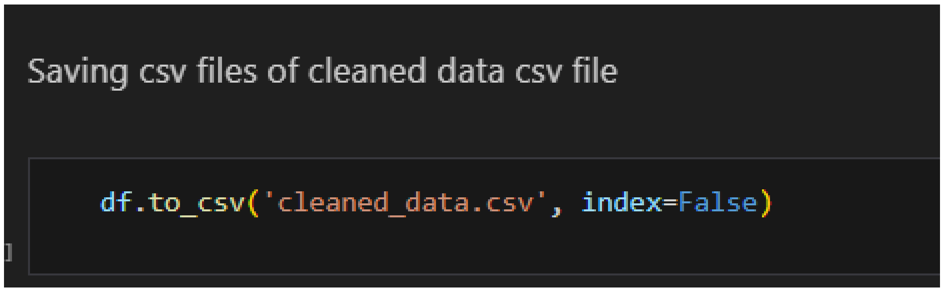
Product, Category


Changing product and category data type to string

```
object_columns = ['product', 'category']


for column in object_columns:
    df[column] = df[column].astype(str)

df
```

```
           date     product     category  price  quantity  revenue
  0  2022-01-01  Smartphone  Electronics    600        10     6000
  1  2022-01-01      Laptop  Electronics   1200         5     6000
  2  2022-01-02     T-Shirt     Clothing     20        50     1000
  3  2022-01-03  Headphones  Electronics    100        20     2000
  4  2022-01-04     T-Shirt     Clothing     20        25      500
...         ...         ...          ...    ...       ...      ...
364  2022-12-27       Watch  Accessories    150         5      750
365  2022-12-28        Coat     Clothing    100         5      500
366  2022-12-29  Headphones  Electronics    100        10     1000
367  2022-12-30  Smartphone  Electronics    600        11     6600
368  2022-12-31      Hoodie     Clothing     40        30     1200

365 rows × 6 columns
```

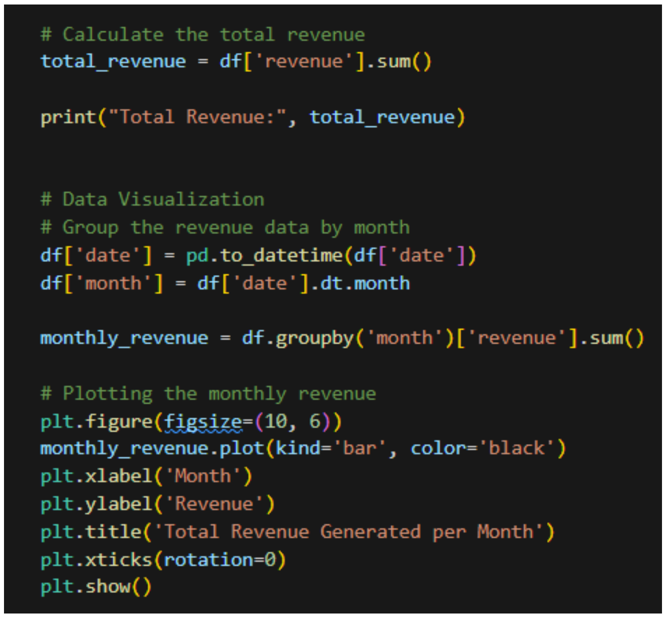
Saving at the end of data preparation cleaned dataframe csv.


Saving csv files of cleaned data csv file

```
df.to_csv('cleaned_data.csv', index=False)
```

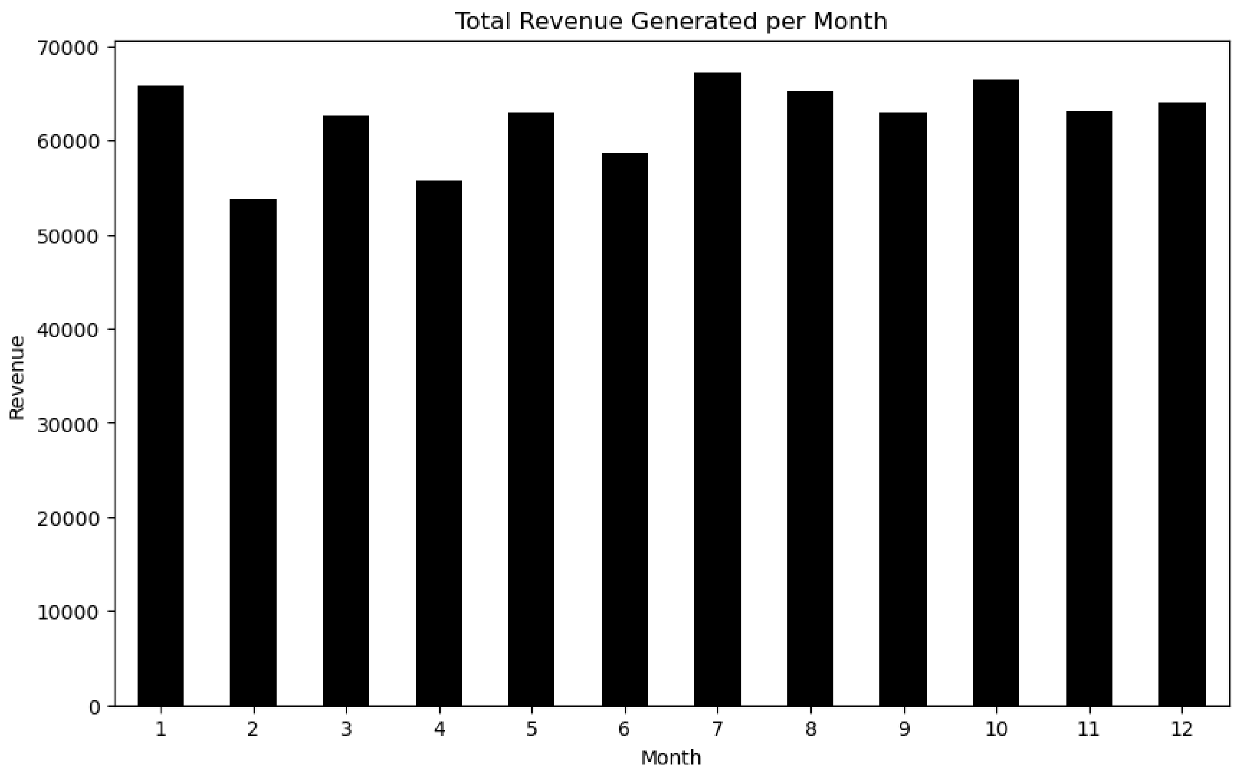
Operations / Answer of all 7 question with visulation.

1. What was the total revenue generated by the company over the course of the year?

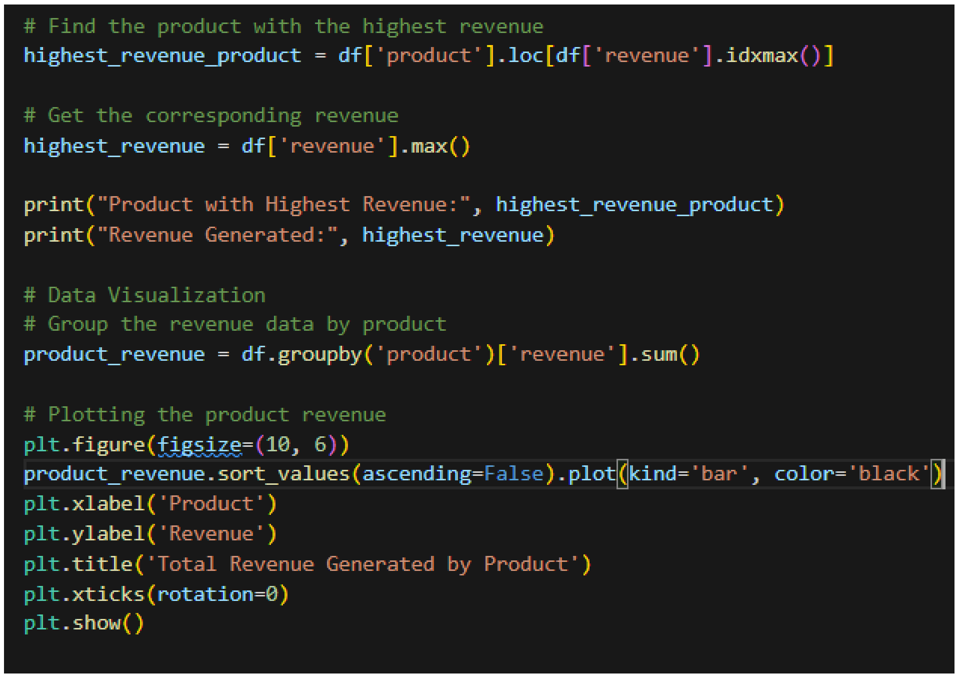


```python
# Calculate the total revenue
total_revenue = df['revenue'].sum()

print("Total Revenue:", total_revenue)


# Data Visualization
# Group the revenue data by month
df['date'] = pd.to_datetime(df['date'])
df['month'] = df['date'].dt.month

monthly_revenue = df.groupby('month')['revenue'].sum()

# Plotting the monthly revenue
plt.figure(figsize=(10, 6))
monthly_revenue.plot(kind='bar', color='black')
plt.xlabel('Month')
plt.ylabel('Revenue')
plt.title('Total Revenue Generated per Month')
plt.xticks(rotation=0)
plt.show()
```

Visualization

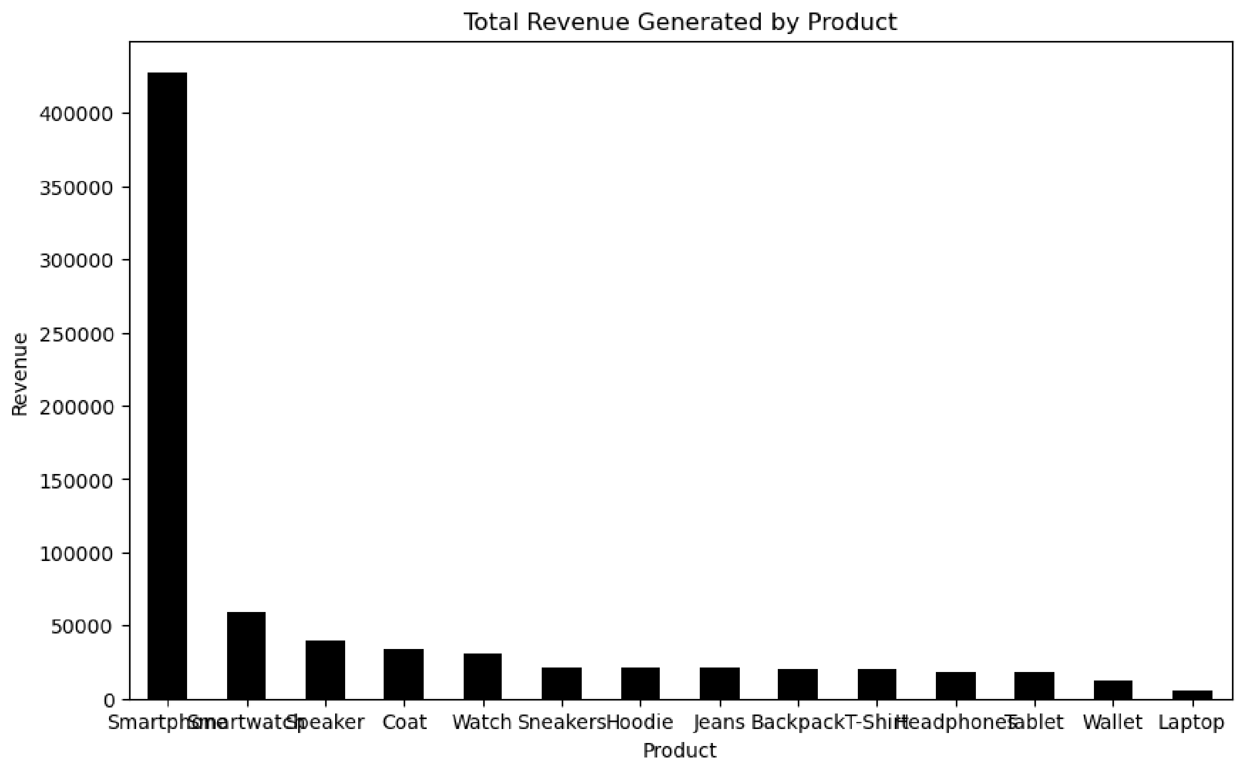


2. Which product had the highest revenue? How much revenue did it generate?

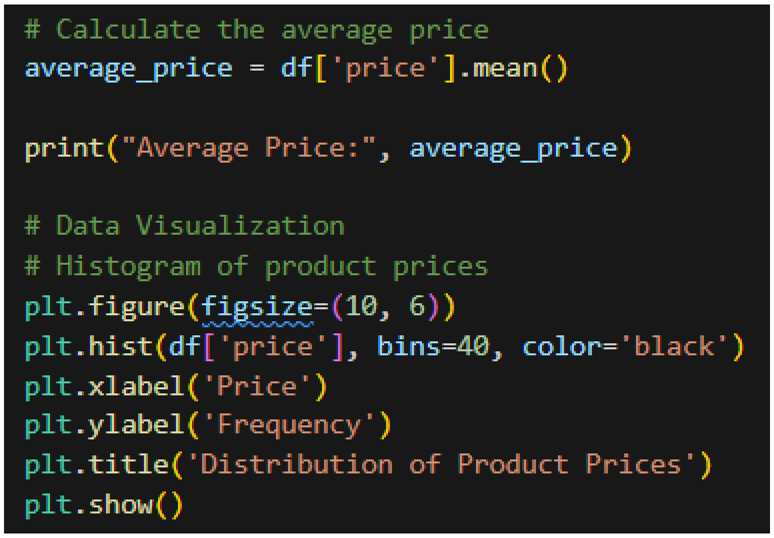


```python
# Find the product with the highest revenue
highest_revenue_product = df['product'].loc[df['revenue'].idxmax()]

# Get the corresponding revenue
highest_revenue = df['revenue'].max()

print("Product with Highest Revenue:", highest_revenue_product)
print("Revenue Generated:", highest_revenue)

# Data Visualization
# Group the revenue data by product
product_revenue = df.groupby('product')['revenue'].sum()

# Plotting the product revenue
plt.figure(figsize=(10, 6))
product_revenue.sort_values(ascending=False).plot(kind='bar', color='black')
plt.xlabel('Product')
plt.ylabel('Revenue')
plt.title('Total Revenue Generated by Product')
plt.xticks(rotation=0)
plt.show()
```

Visualization

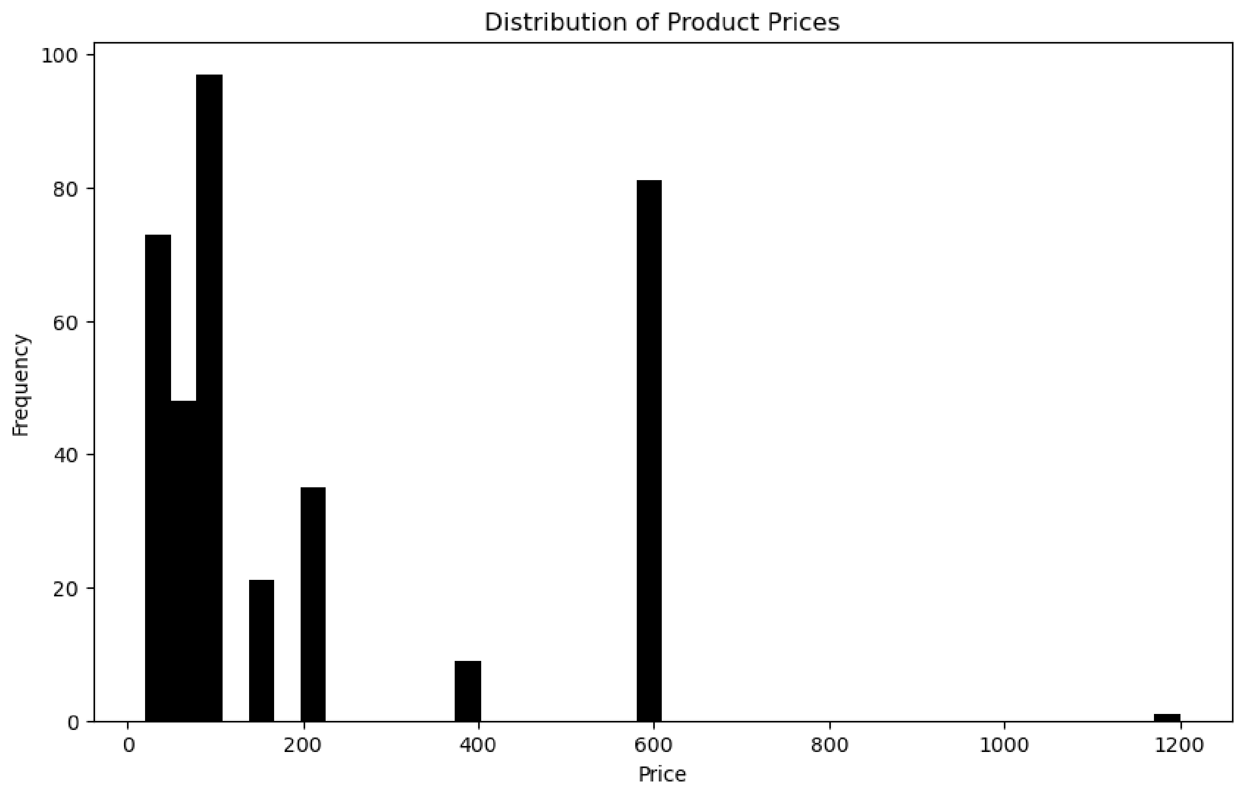


3. What was the average price of a product sold by the company?

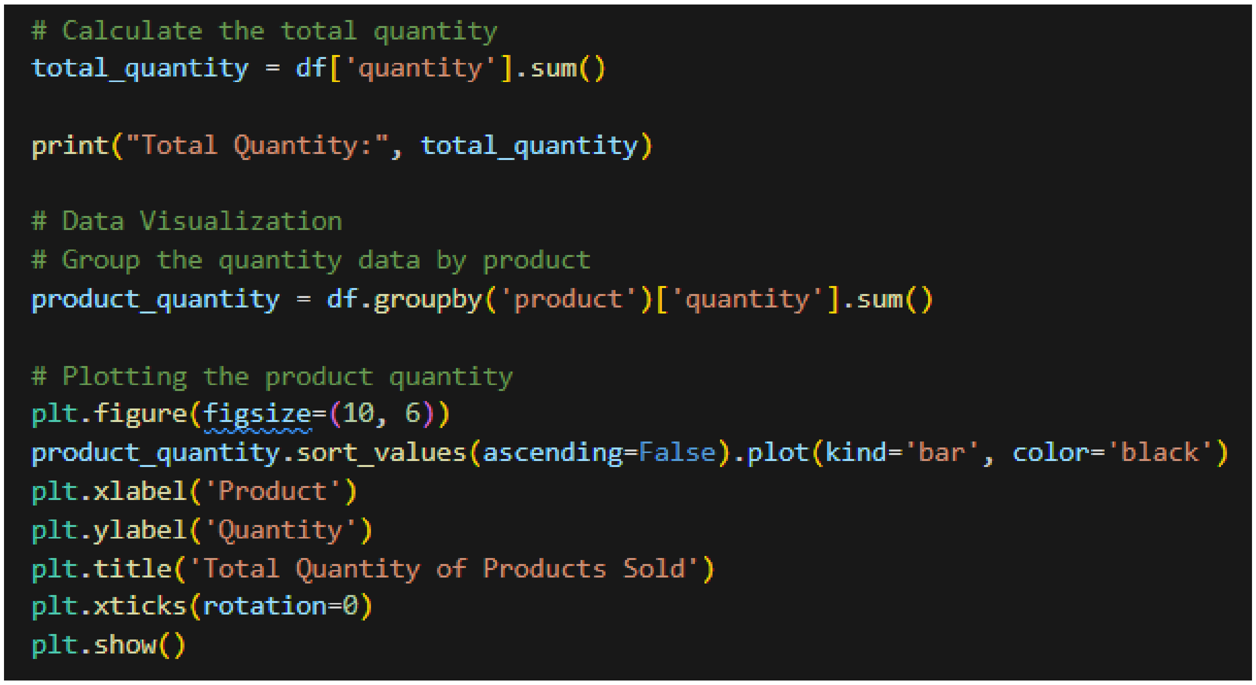


```python
# Calculate the average price
average_price = df['price'].mean()

print("Average Price:", average_price)

# Data Visualization
# Histogram of product prices
plt.figure(figsize=(10, 6))
plt.hist(df['price'], bins=40, color='black')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.title('Distribution of Product Prices')
plt.show()
```

Visualization

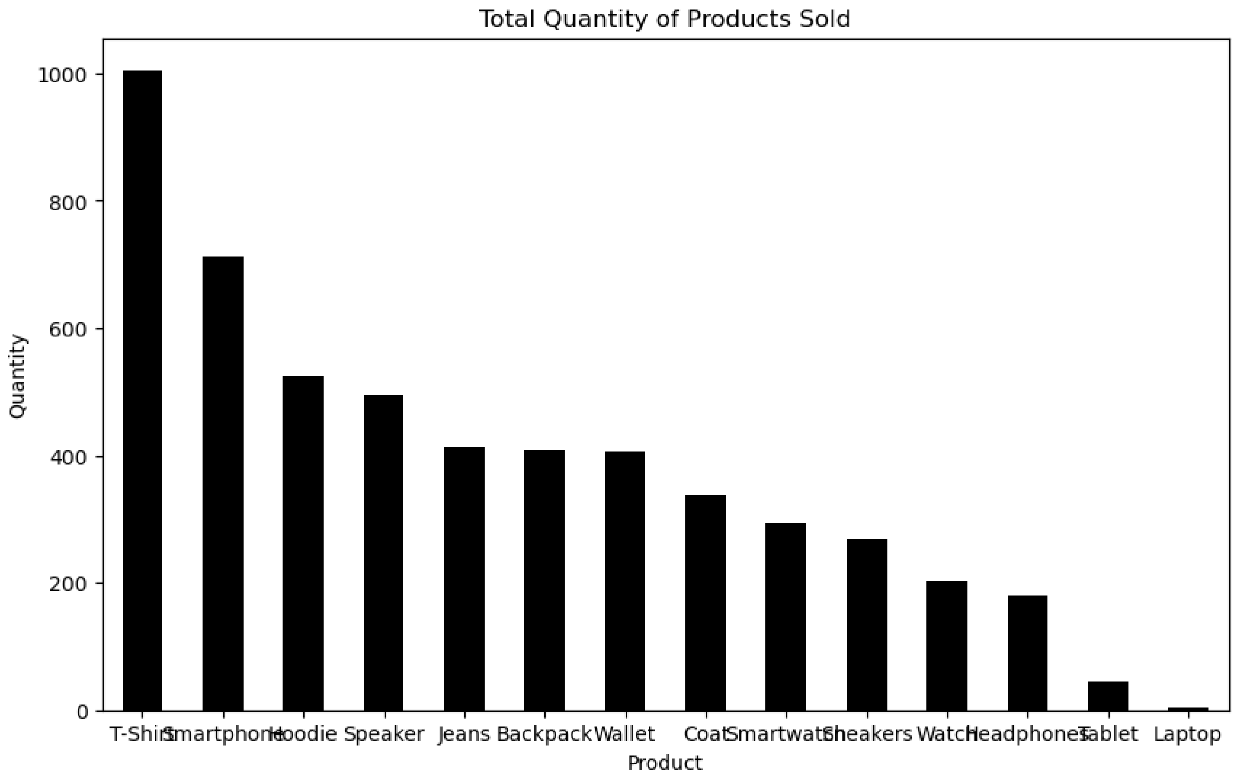


4. What was the total quantity of products sold by the company?

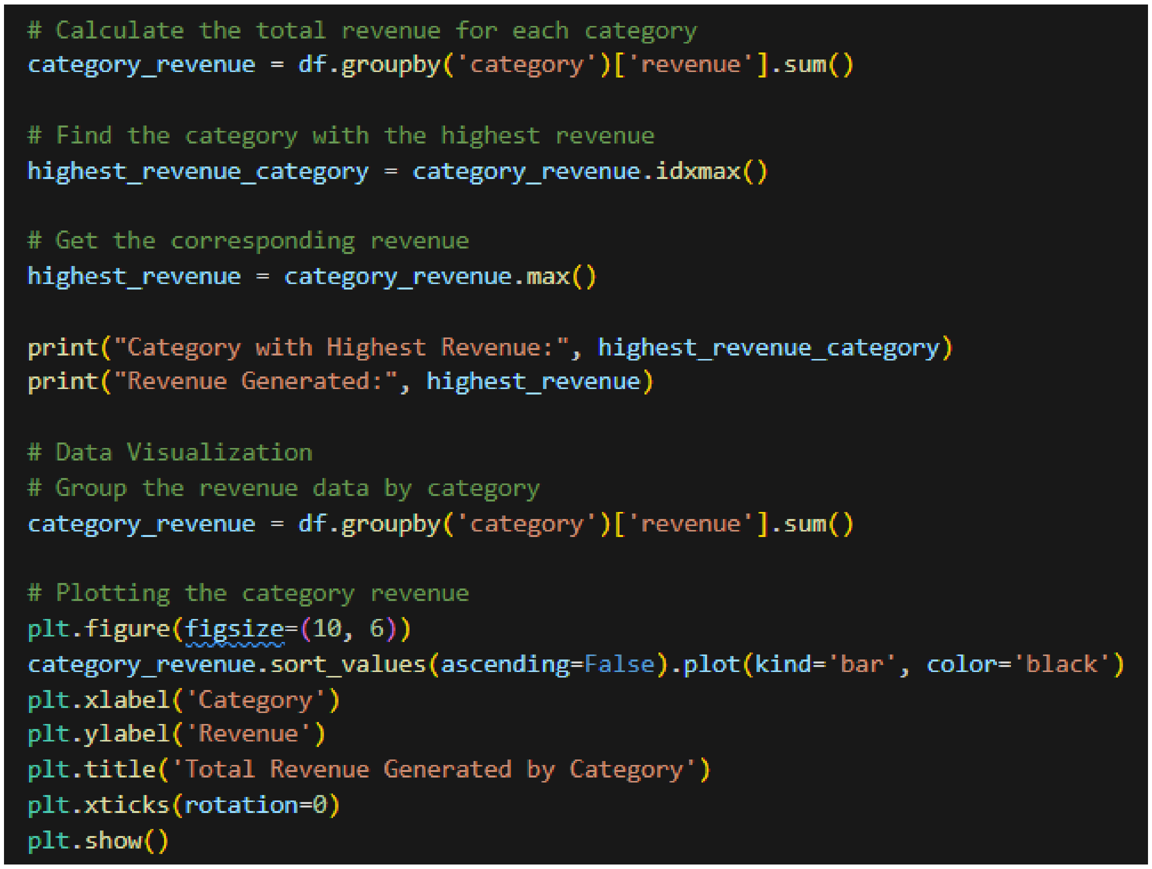


```python
# Calculate the total quantity
total_quantity = df['quantity'].sum()

print("Total Quantity:", total_quantity)

# Data Visualization
# Group the quantity data by product
product_quantity = df.groupby('product')['quantity'].sum()

# Plotting the product quantity
plt.figure(figsize=(10, 6))
product_quantity.sort_values(ascending=False).plot(kind='bar', color='black')
plt.xlabel('Product')
plt.ylabel('Quantity')
plt.title('Total Quantity of Products Sold')
plt.xticks(rotation=0)
plt.show()
```

Visualization

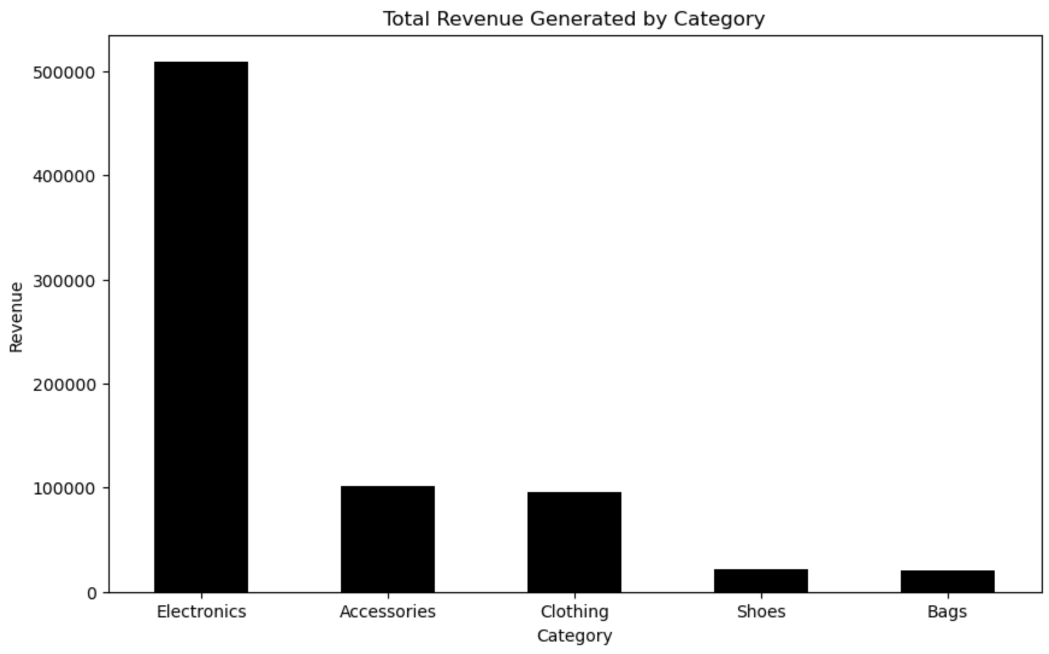


5. Which category had the highest revenue? How much revenue did it generate?

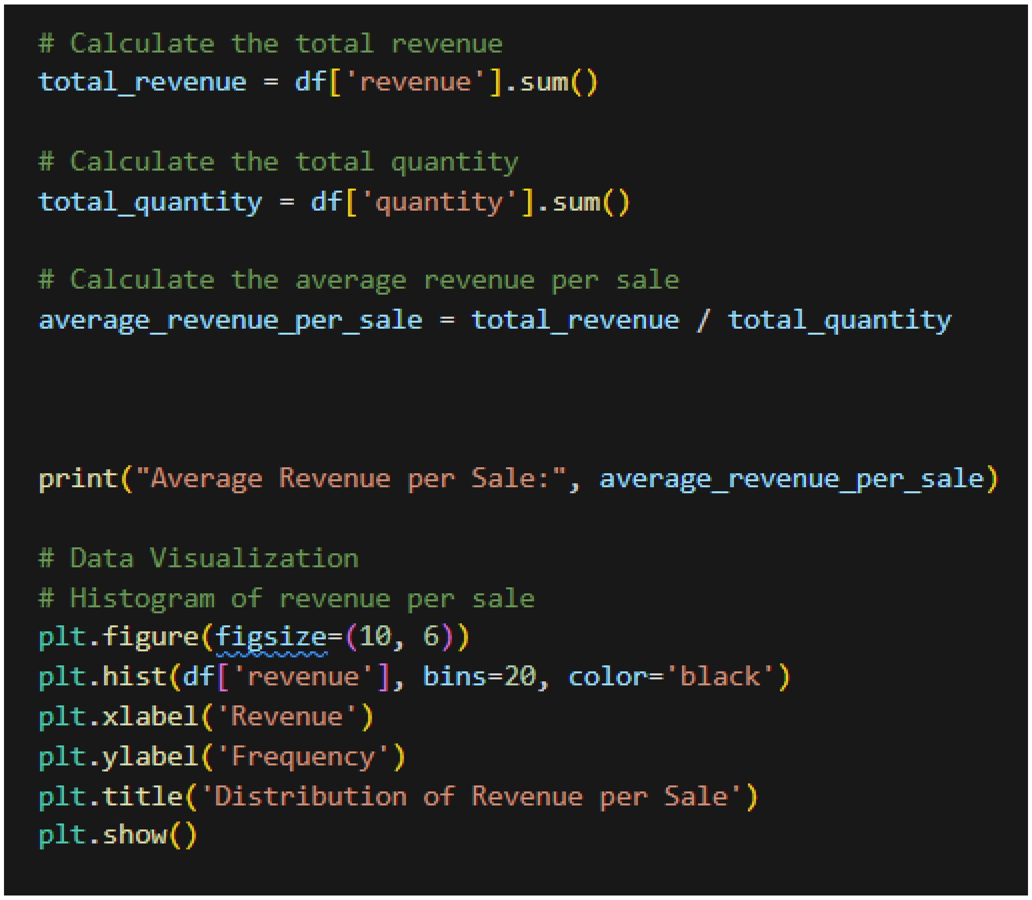


```python
# Calculate the total revenue for each category
category_revenue = df.groupby('category')['revenue'].sum()

# Find the category with the highest revenue
highest_revenue_category = category_revenue.idxmax()

# Get the corresponding revenue
highest_revenue = category_revenue.max()

print("Category with Highest Revenue:", highest_revenue_category)
print("Revenue Generated:", highest_revenue)

# Data Visualization
# Group the revenue data by category
category_revenue = df.groupby('category')['revenue'].sum()

# Plotting the category revenue
plt.figure(figsize=(10, 6))
category_revenue.sort_values(ascending=False).plot(kind='bar', color='black')
plt.xlabel('Category')
plt.ylabel('Revenue')
plt.title('Total Revenue Generated by Category')
plt.xticks(rotation=0)
plt.show()
```

Visualization

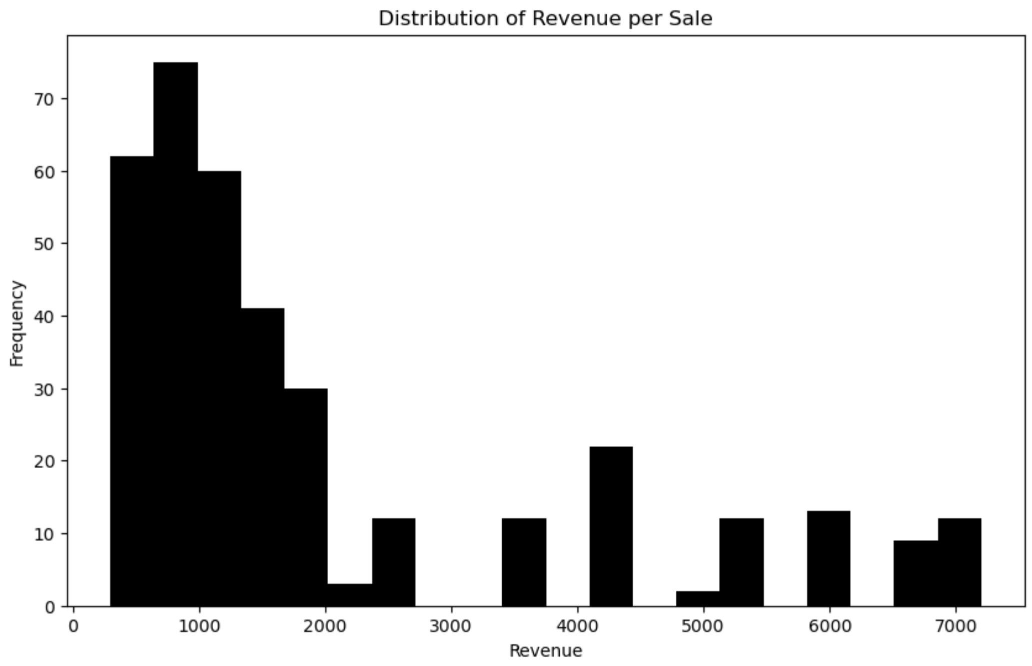


6. What was the average revenue per sale?

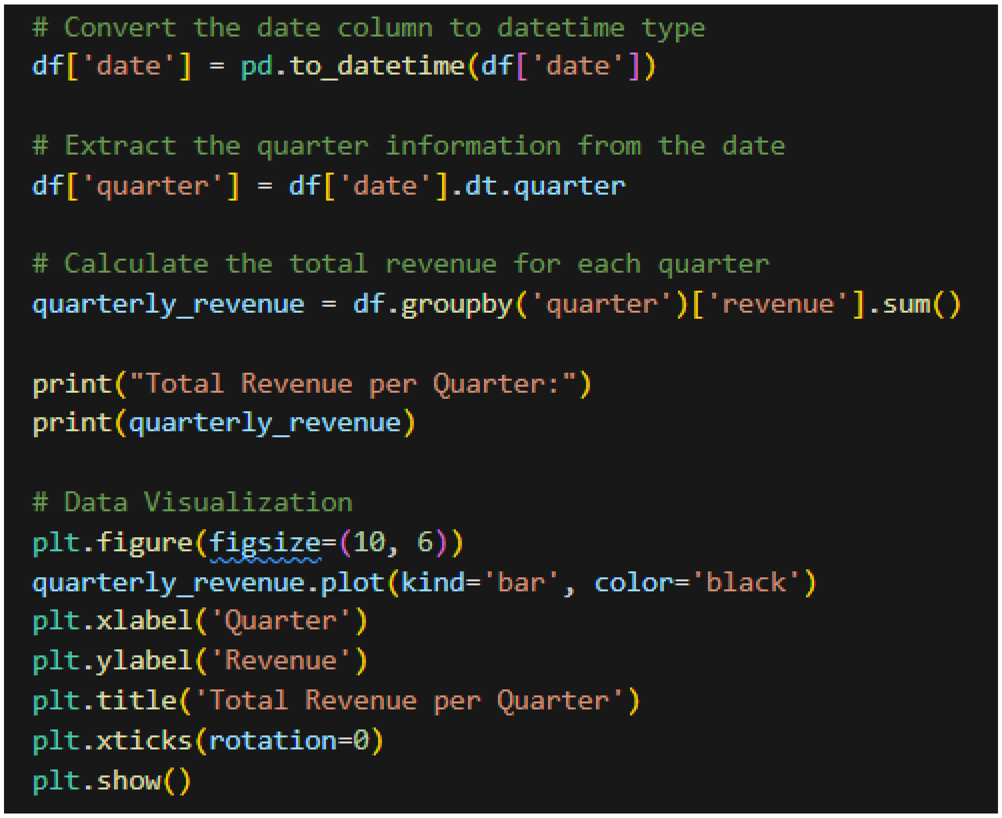


```
# Calculate the total revenue
total_revenue = df['revenue'].sum()

# Calculate the total quantity
total_quantity = df['quantity'].sum()

# Calculate the average revenue per sale
average_revenue_per_sale = total_revenue / total_quantity



print("Average Revenue per Sale:", average_revenue_per_sale)

# Data Visualization
# Histogram of revenue per sale
plt.figure(figsize=(10, 6))
plt.hist(df['revenue'], bins=20, color='black')
plt.xlabel('Revenue')
plt.ylabel('Frequency')
plt.title('Distribution of Revenue per Sale')
plt.show()
```

Visualization

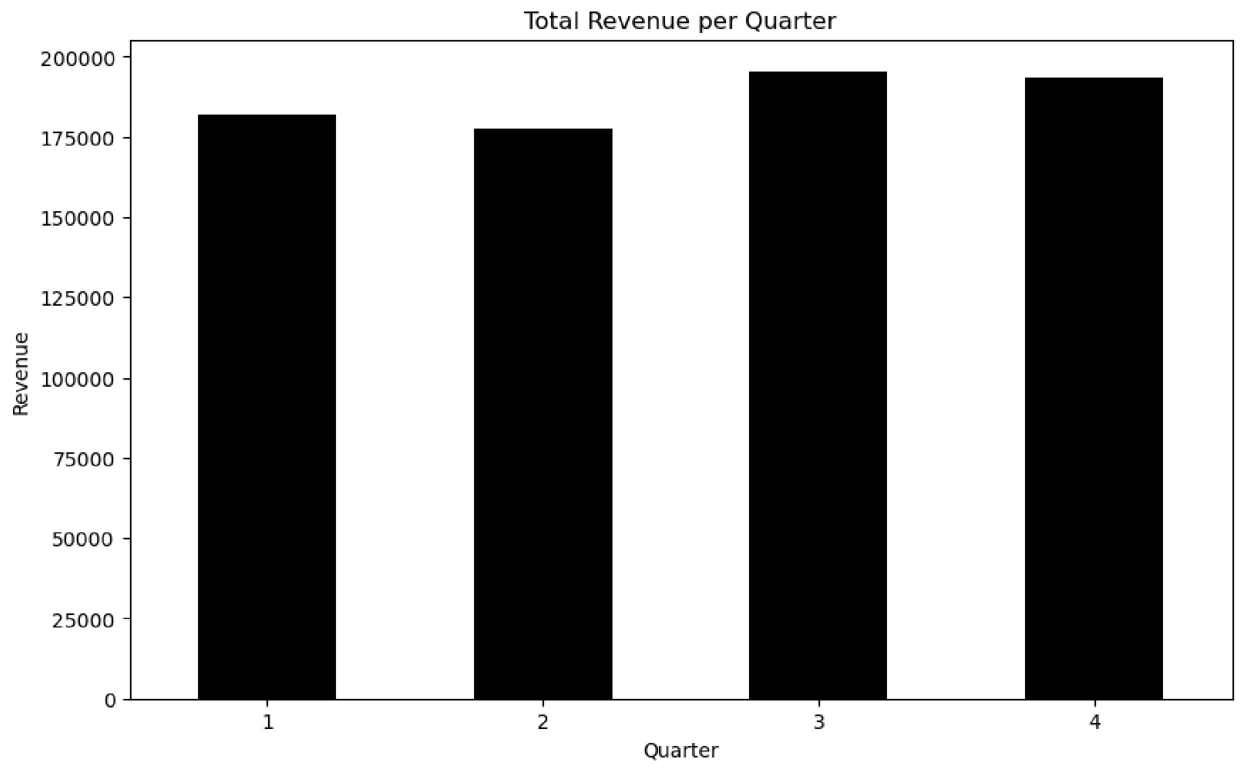


7. What was the total revenue generated in each quarter of the year? (i.e. Q1, Q2, Q3, Q4)



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

# Extract the quarter information from the date
df['quarter'] = df['date'].dt.quarter

# Calculate the total revenue for each quarter
quarterly_revenue = df.groupby('quarter')['revenue'].sum()

print("Total Revenue per Quarter:")
print(quarterly_revenue)

# Data Visualization
plt.figure(figsize=(10, 6))
quarterly_revenue.plot(kind='bar', color='black')
plt.xlabel('Quarter')
plt.ylabel('Revenue')
plt.title('Total Revenue per Quarter')
plt.xticks(rotation=0)
plt.show()
```

Visualization


Total Revenue per Quarter
Revenue
200000
175000
150000
125000
100000
75000
50000
25000
0
1
2
3
4
Quarter