Sales And Profit Analysis Using Python

By Himanshu .K

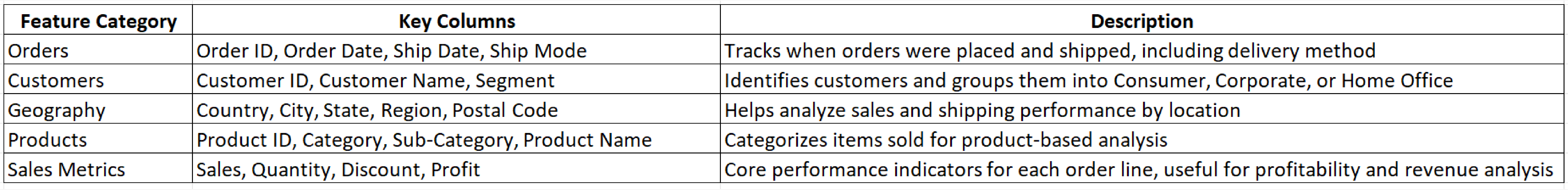
Objective

To analyze sales and profitability trends to uncover opportunities for growth and optimization

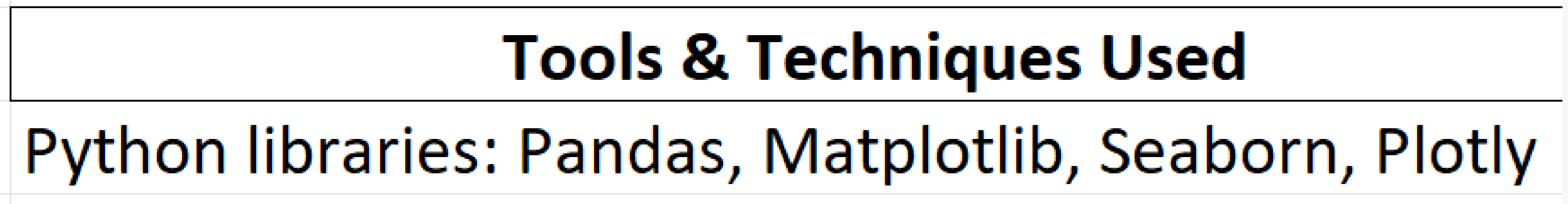
Business Questions

1. Which month generated the highest and lowest sales, and what factors contributed to these trends?
2. Which product category drives the most revenue, and which underperforms—why might this be the case?
3. How do different product sub-categories contribute to overall sales, and are there hidden growth opportunities among them?
4. Which months deliver peak profitability, and how can the business capitalize on this seasonal behavior?
5. Which product categories and sub-categories are most profitable, and are there any that impact overall profitability negatively?
6. Which customer segments are most valuable in terms of sales and profit, and how should marketing and sales strategies adapt accordingly?

Dataset Overview



Feature Category	Key Columns	Description
Orders	Order ID, Order Date, Ship Date, Ship Mode	Tracks when orders were placed and shipped, including delivery method
Customers	Customer ID, Customer Name, Segment	Identifies customers and groups them into Consumer, Corporate, or Home Office
Geography	Country, City, State, Region, Postal Code	Helps analyze sales and shipping performance by location
Products	Product ID, Category, Sub-Category, Product Name	Categorizes items sold for product-based analysis
Sales Metrics	Sales, Quantity, Discount, Profit	Core performance indicators for each order line, useful for profitability and revenue analysis

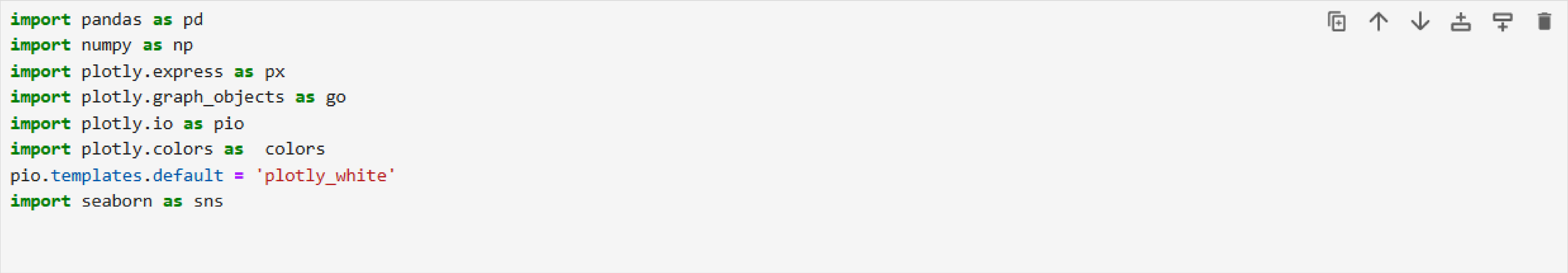


Tools & Techniques Used

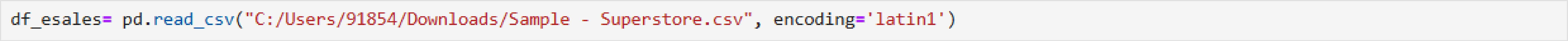
Python libraries: Pandas, Matplotlib, Seaborn, Plotly

Data Cleaning & Preparation

# Importing Libraries into Python

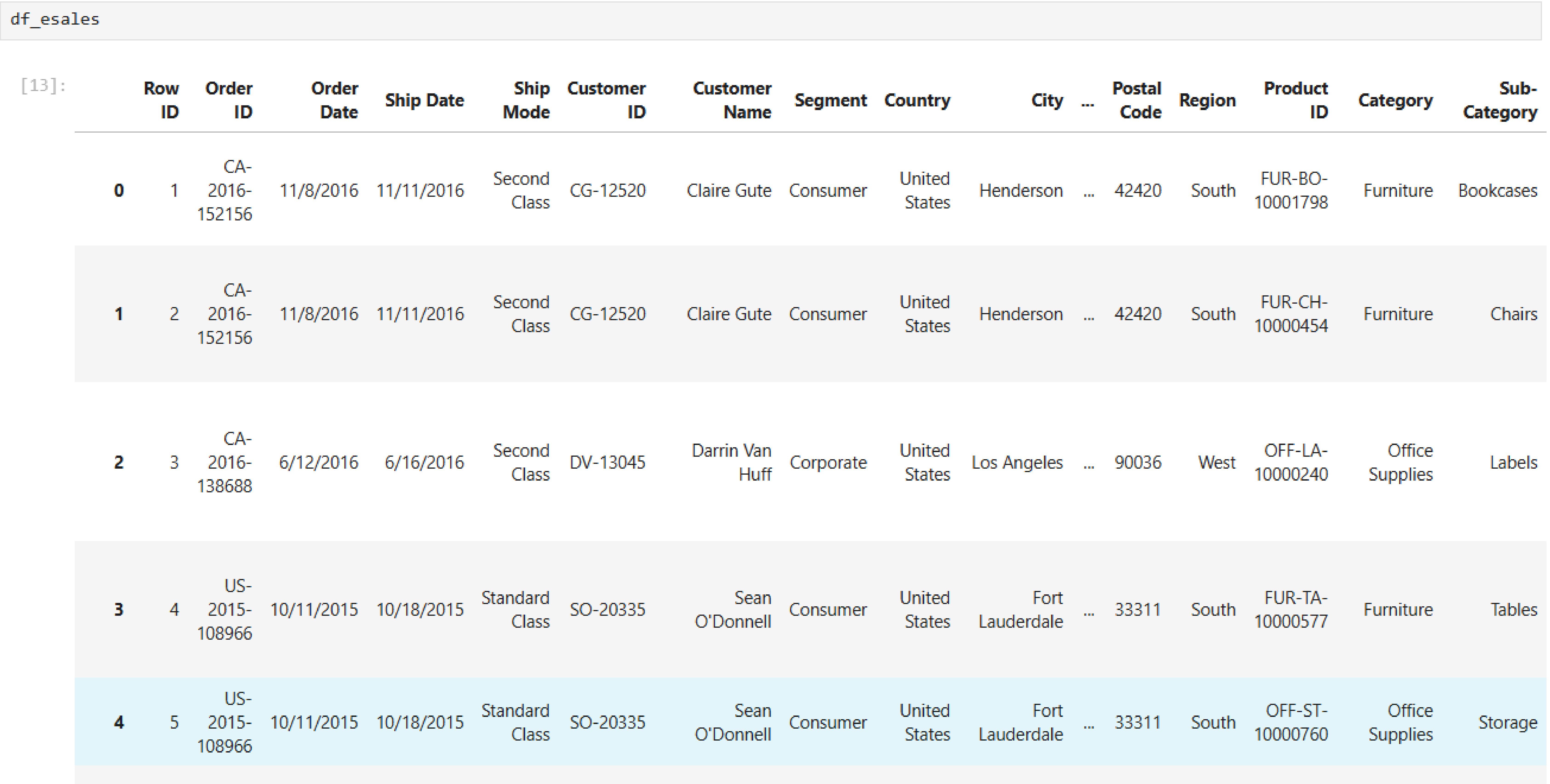


```python
import pandas as pd
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
import plotly.colors as  colors
pio.templates.default = 'plotly_white'
import seaborn as sns
```



```python
df_esales= pd.read_csv("C:/Users/91854/Downloads/Sample - Superstore.csv", encoding='latin1')
```

#Exploring The Dataset

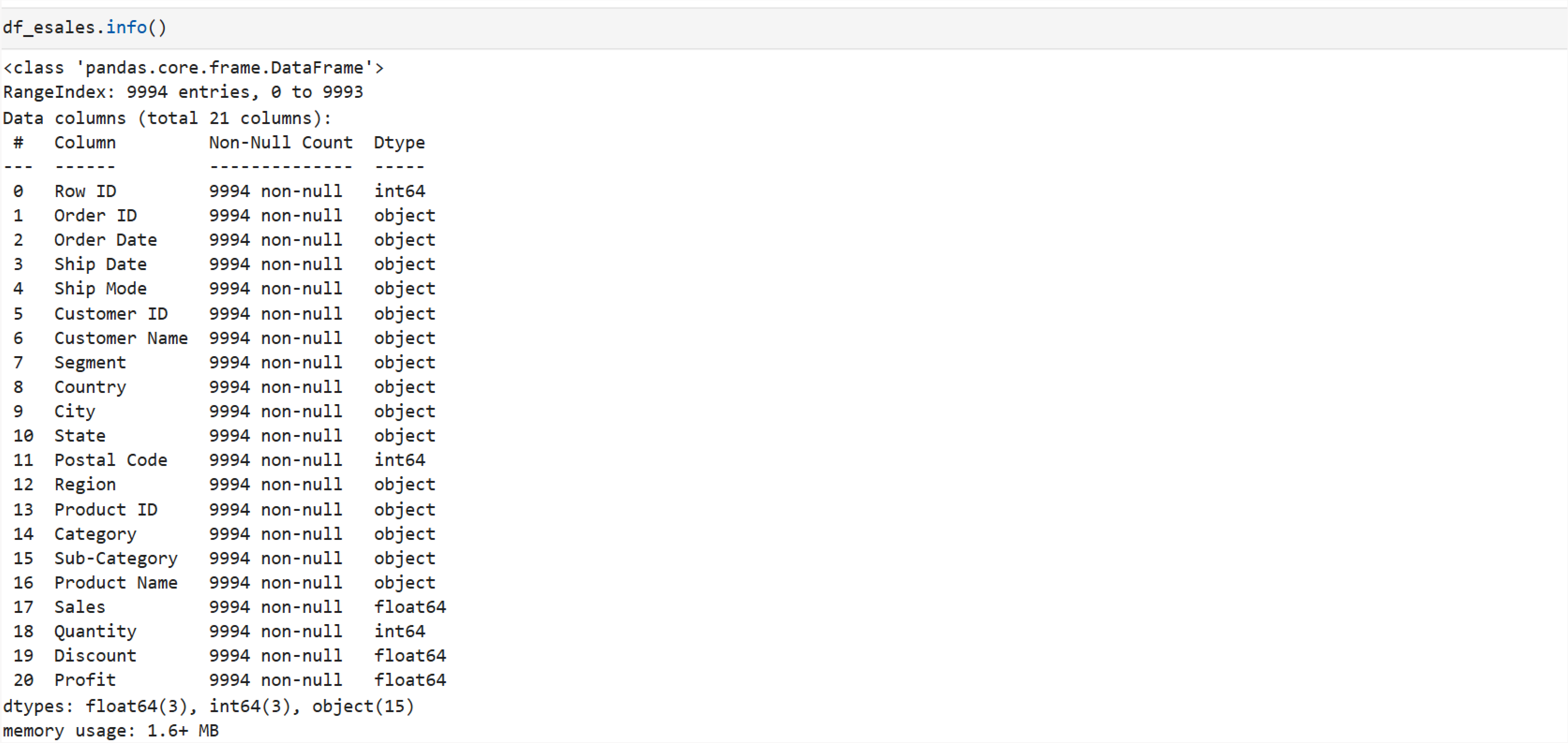


```
df_esales
```

[13]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category	Sub-Category
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-BO-10001798	Furniture	Bookcases
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-CH-10000454	Furniture	Chairs
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036	West	OFF-LA-10000240	Office Supplies	Labels
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	FUR-TA-10000577	Furniture	Tables
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	OFF-ST-10000760	Office Supplies	Storage

DataFrame’s structure, including row count, column names, non-null counts, data types, and memory usage.

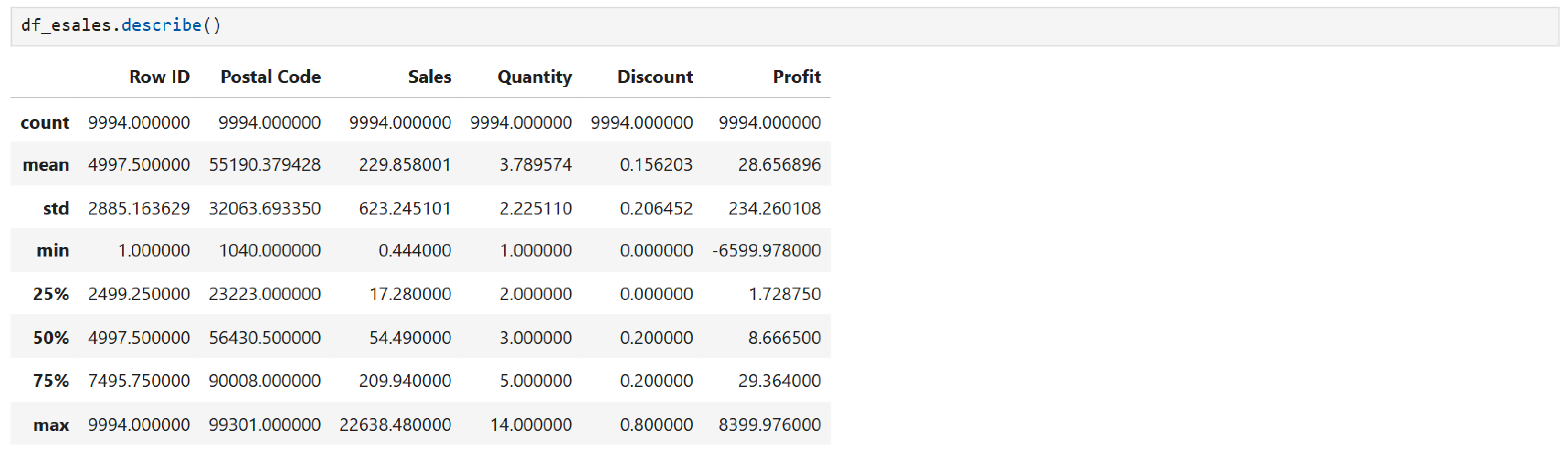


```
df_esales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Row ID         9994 non-null   int64  
 1   Order ID       9994 non-null   object 
 2   Order Date     9994 non-null   object 
 3   Ship Date      9994 non-null   object 
 4   Ship Mode      9994 non-null   object 
 5   Customer ID    9994 non-null   object 
 6   Customer Name  9994 non-null   object 
 7   Segment        9994 non-null   object 
 8   Country        9994 non-null   object 
 9   City           9994 non-null   object 
 10  State          9994 non-null   object 
 11  Postal Code    9994 non-null   int64  
 12  Region         9994 non-null   object 
 13  Product ID     9994 non-null   object 
 14  Category       9994 non-null   object 
 15  Sub-Category   9994 non-null   object 
 16  Product Name   9994 non-null   object 
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64  
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

Your paragraph text

Generating descriptive statistics

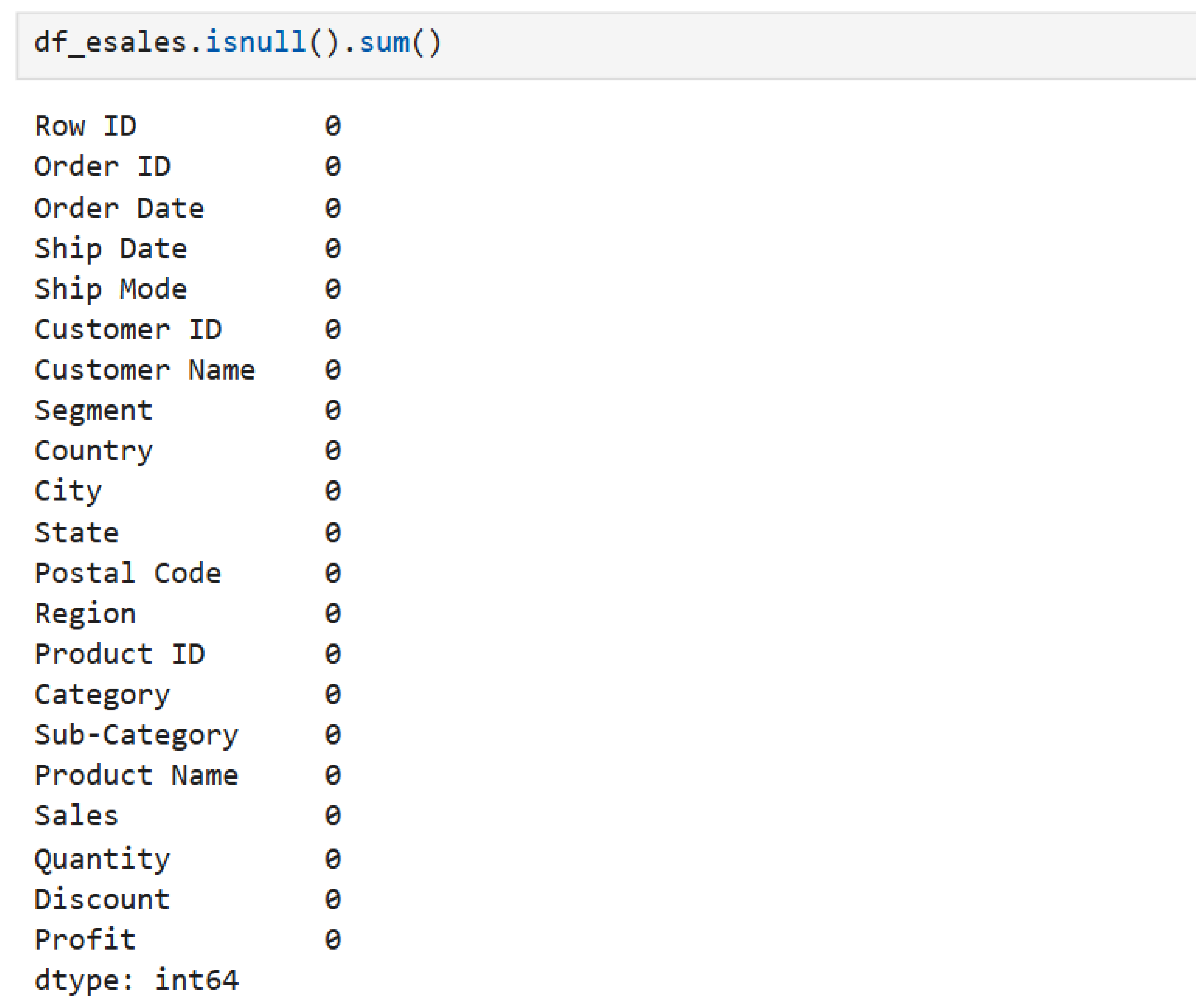


```
df_esales.describe()
```

	Row ID	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	55190.379428	229.858001	3.789574	0.156203	28.656896
std	2885.163629	32063.693350	623.245101	2.225110	0.206452	234.260108
min	1.000000	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	99301.000000	22638.480000	14.000000	0.800000	8399.976000

Summary statistics of numeric columns in a DataFrame, including count, mean, standard deviation, min, max, and percentiles. It helps to understand the data better.

Checking for the null values in the dataset



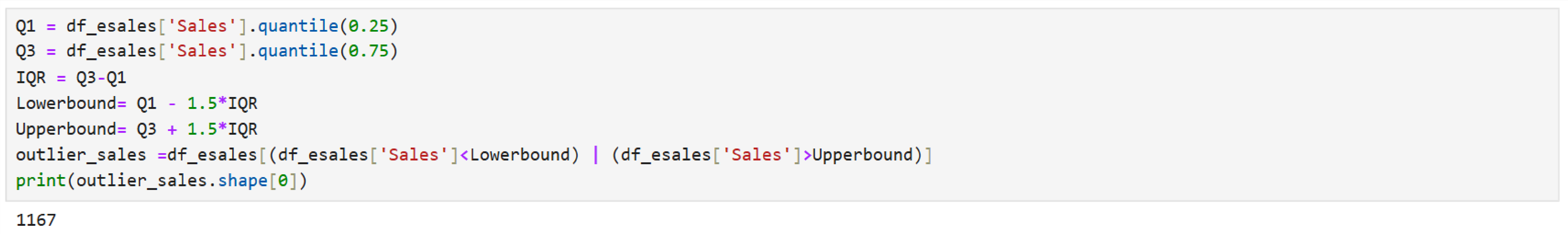
```
df_esales.isnull().sum()
```

```
Row ID           0
Order ID         0
Order Date       0
Ship Date        0
Ship Mode        0
Customer ID      0
Customer Name    0
Segment          0
Country          0
City             0
State            0
Postal Code      0
Region           0
Product ID       0
Category         0
Sub-Category     0
Product Name     0
Sales            0
Quantity         0
Discount         0
Profit           0
dtype: int64
```

Since there is no null value in the dataset so we don’t need to handle the null values .

Outlier Detection Using IQR Method

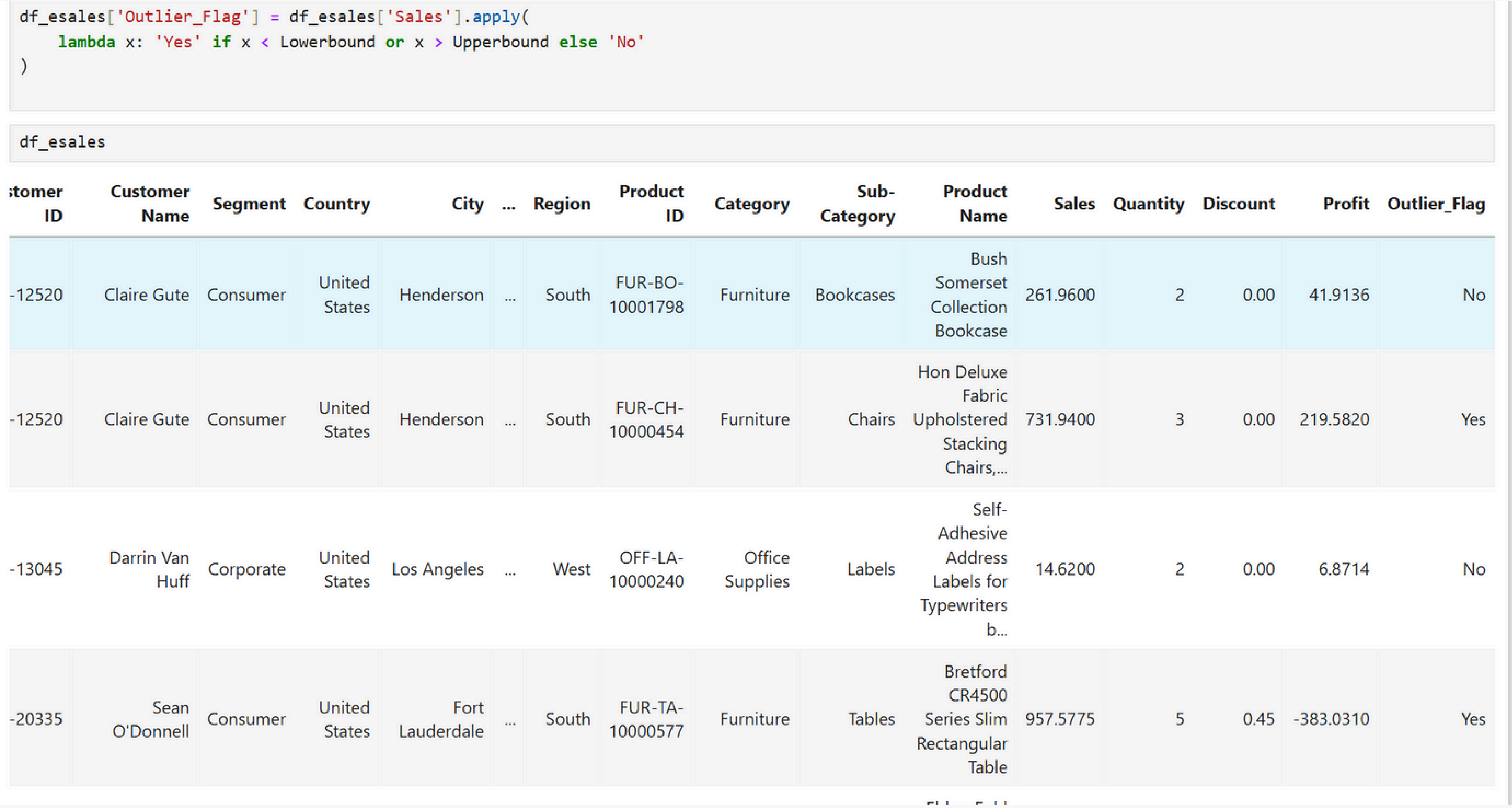
Your paragraph text



```
Q1 = df_esales['Sales'].quantile(0.25)
Q3 = df_esales['Sales'].quantile(0.75)
IQR = Q3-Q1
Lowerbound= Q1 - 1.5*IQR
Upperbound= Q3 + 1.5*IQR
outlier_sales =df_esales[(df_esales['Sales']<Lowerbound) | (df_esales['Sales']>Upperbound)]
print(outlier_sales.shape[0])
```

```
1167
```

Since the no. of outliers is significantly more , I will flag the outliers instead of removing it.

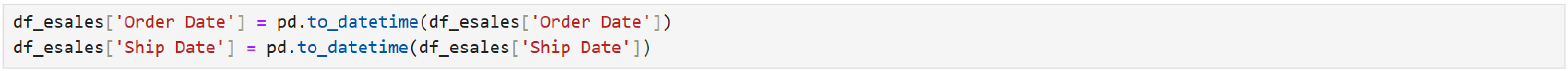


```
df_esales['Outlier_Flag'] = df_esales['Sales'].apply(
    lambda x: 'Yes' if x < Lowerbound or x > Upperbound else 'No'
)
```

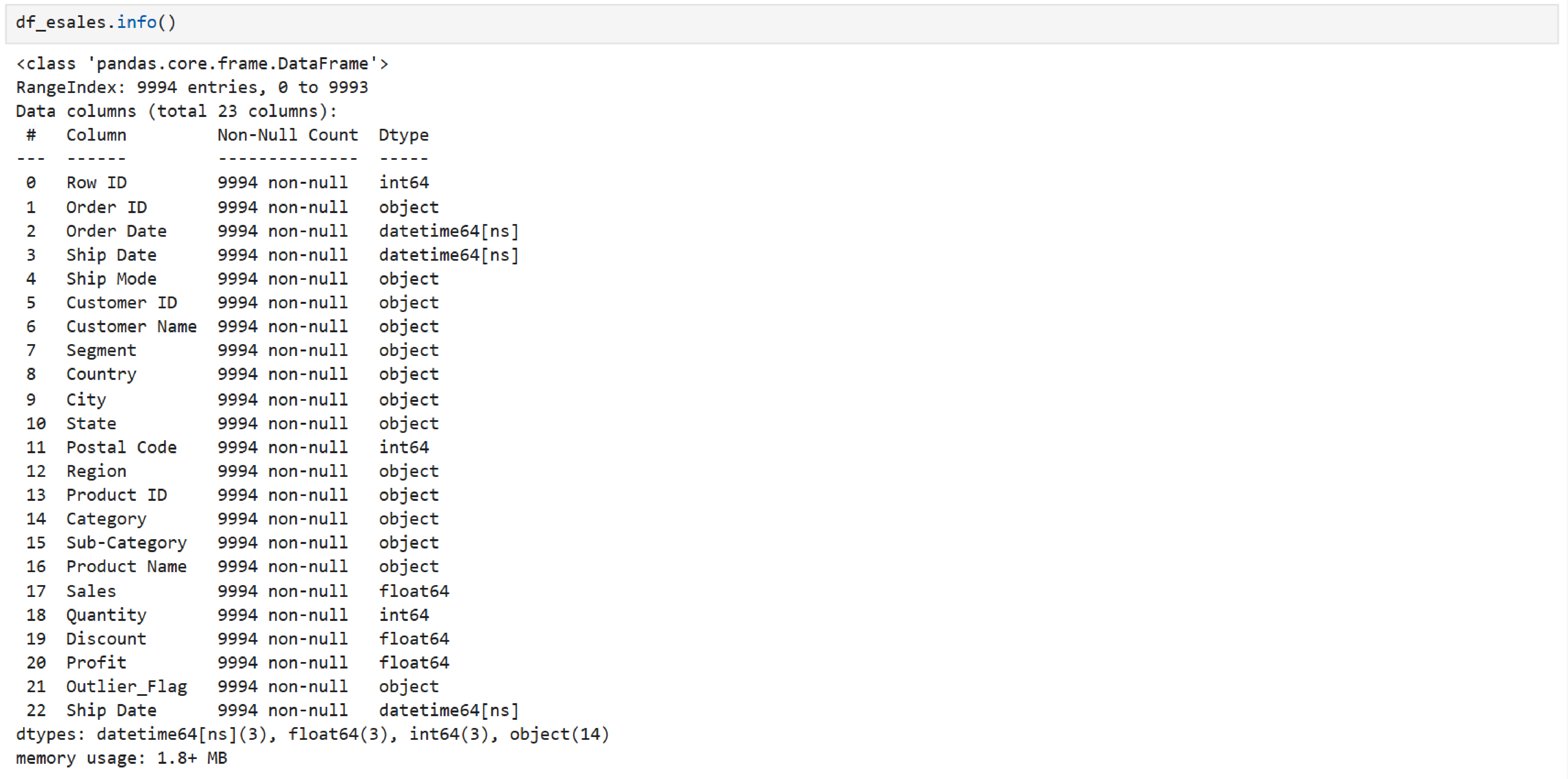
```
df_esales
```

stomer ID	Customer Name	Segment	Country	City	...	Region	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Discount	Profit	Outlier_Flag
-12520	Claire Gute	Consumer	United States	Henderson	...	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.9600	2	0.00	41.9136	No
-12520	Claire Gute	Consumer	United States	Henderson	...	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs,...	731.9400	3	0.00	219.5820	Yes
-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters b...	14.6200	2	0.00	6.8714	No
-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.0310	Yes

Converting Order Date and Ship Date to Datetime Format



```
df_esales['Order Date'] = pd.to_datetime(df_esales['Order Date'])
df_esales['Ship Date'] = pd.to_datetime(df_esales['Ship Date'])
```

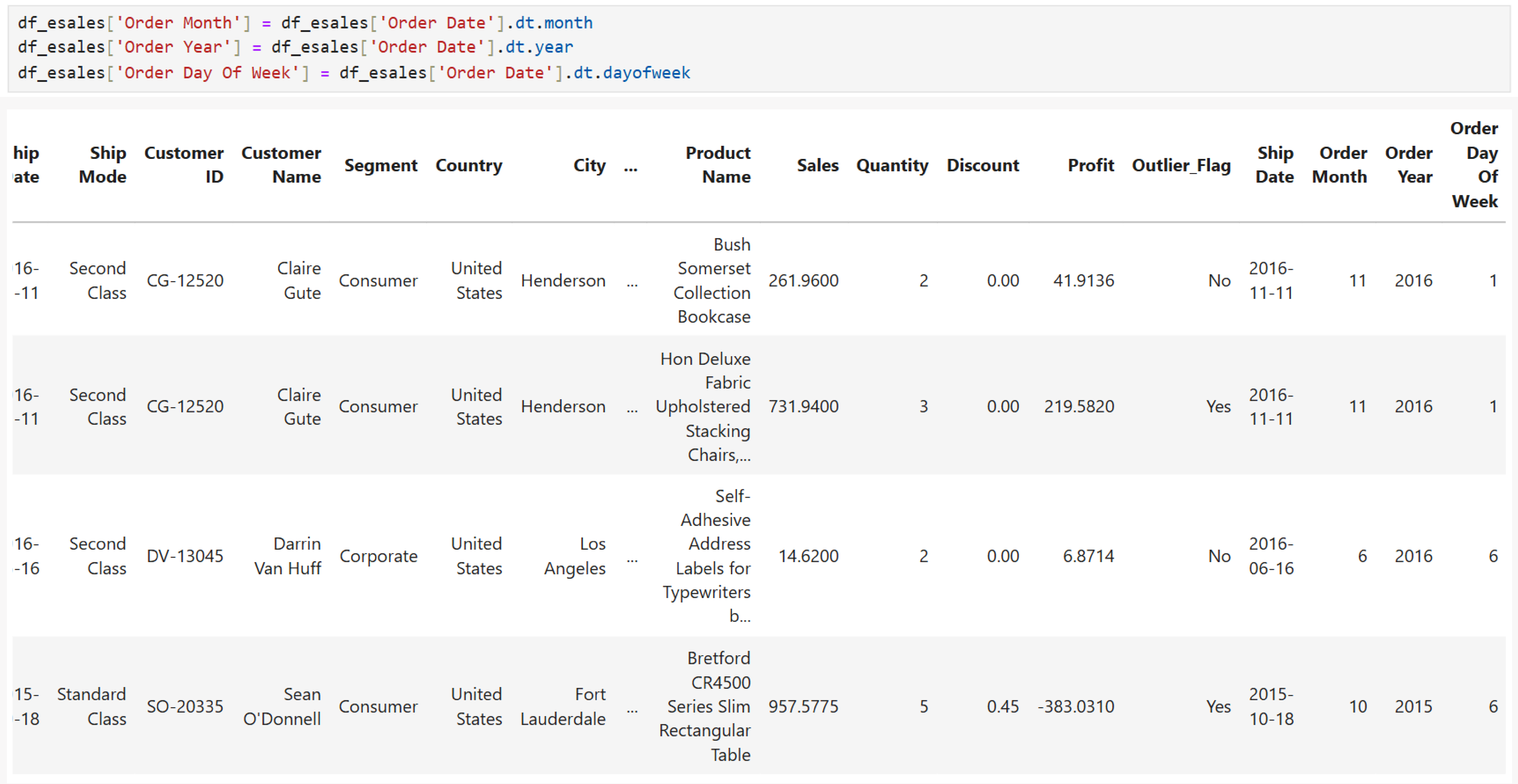


```
df_esales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 23 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   datetime64[ns]
 3   Ship Date      9994 non-null   datetime64[ns]
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
 21  Outlier_Flag   9994 non-null   object
 22  Ship Date      9994 non-null   datetime64[ns]
dtypes: datetime64[ns](3), float64(3), int64(3), object(14)
memory usage: 1.8+ MB
```

This was required to create columns for monthly, yearly and weekly observations of the profit and sales.

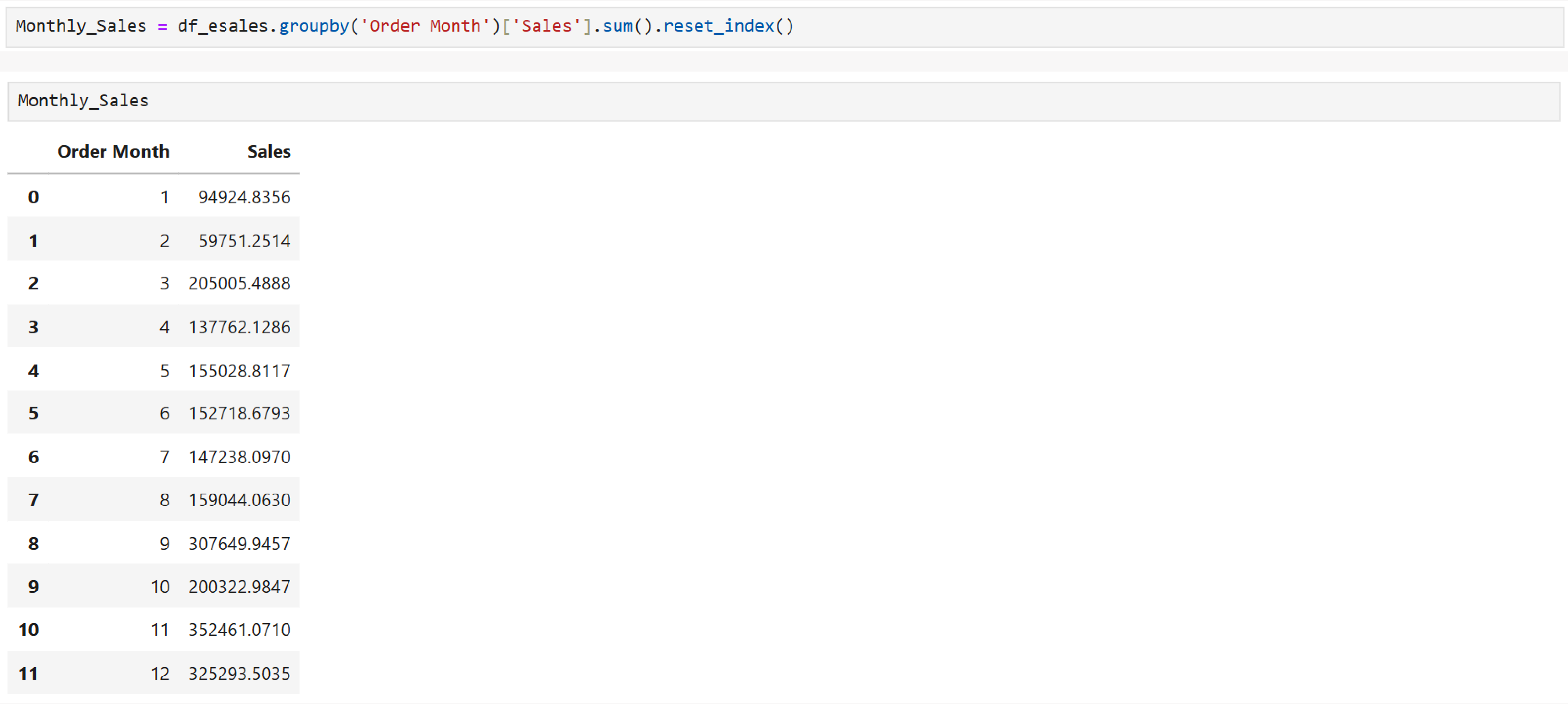
Creating Day Of Week,Month and Year Column for Further Analysis



```
df_esales['Order Month'] = df_esales['Order Date'].dt.month
df_esales['Order Year'] = df_esales['Order Date'].dt.year
df_esales['Order Day Of Week'] = df_esales['Order Date'].dt.dayofweek
```

hip ate	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Product Name	Sales	Quantity	Discount	Profit	Outlier_Flag	Ship Date	Order Month	Order Year	Order Day Of Week
16- -11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	Bush Somerset Collection Bookcase	261.9600	2	0.00	41.9136	No	2016-11-11	11	2016	1
16- -11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	Hon Deluxe Fabric Upholstered Stacking Chairs,...	731.9400	3	0.00	219.5820	Yes	2016-11-11	11	2016	1
16- -16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	Self-Adhesive Address Labels for Typewriters b...	14.6200	2	0.00	6.8714	No	2016-06-16	6	2016	6
15- -18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.0310	Yes	2015-10-18	10	2015	6

Monthly Sales Analysis

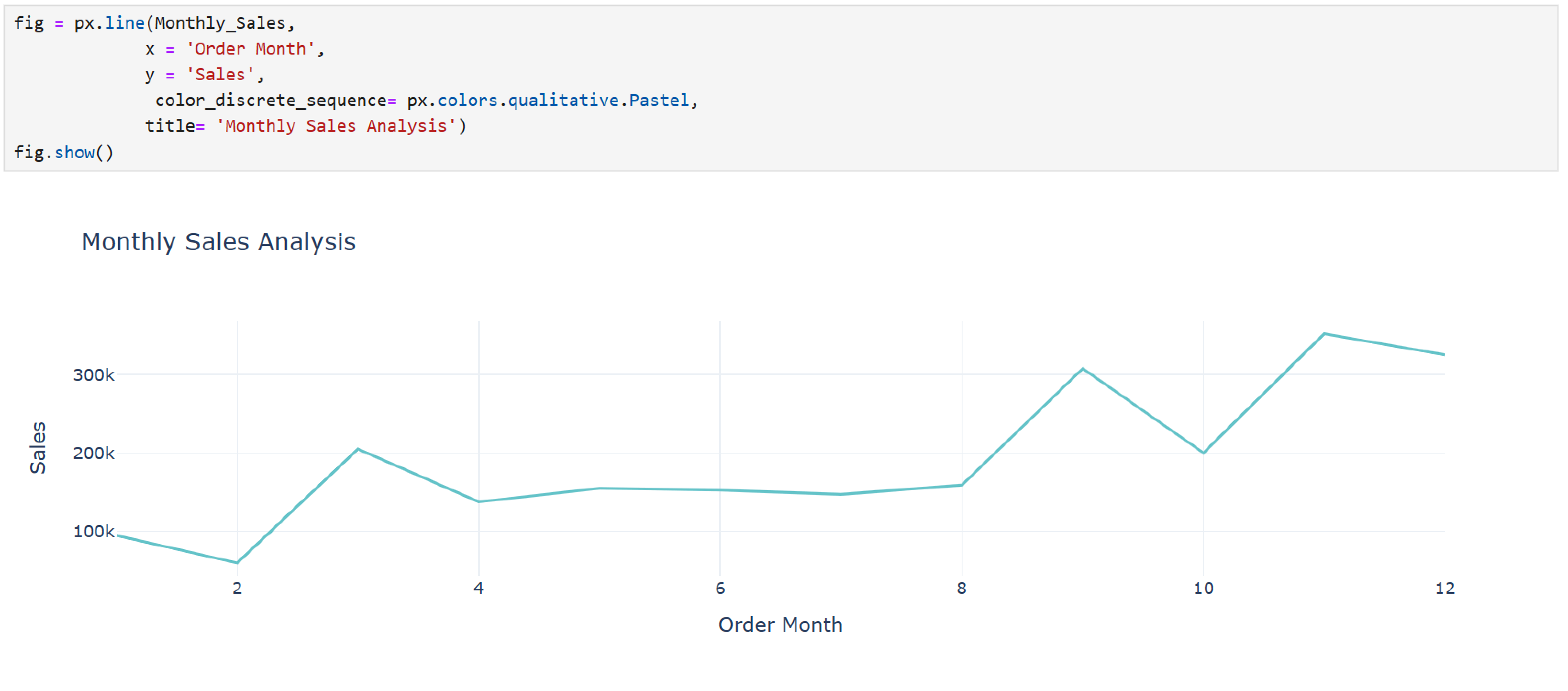


```
Monthly_Sales = df_esales.groupby('Order Month')['Sales'].sum().reset_index()
```

```
Monthly_Sales
```

```
    Order Month        Sales
 0            1   94924.8356
 1            2   59751.2514
 2            3  205005.4888
 3            4  137762.1286
 4            5  155028.8117
 5            6  152718.6793
 6            7  147238.0970
 7            8  159044.0630
 8            9  307649.9457
 9           10  200322.9847
10           11  352461.0710
11           12  325293.5035
```

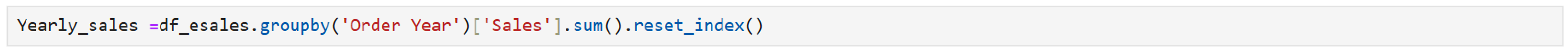
Monthly Sales Analysis Using Line Chart



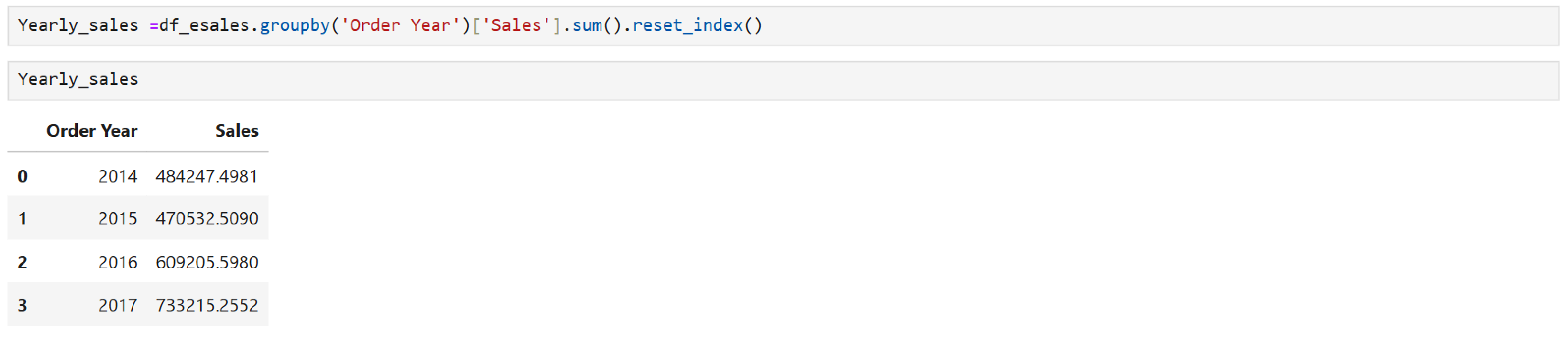
```
fig = px.line(Monthly_Sales,
             x = 'Order Month',
             y = 'Sales',
              color_discrete_sequence= px.colors.qualitative.Pastel,
             title= 'Monthly Sales Analysis')
fig.show()
```

Insight:Sales has significiantly increased throughout the year and has peaked at the end of the year.

Yearly Sales Analysis



```python
Yearly_sales =df_esales.groupby('Order Year')['Sales'].sum().reset_index()
```

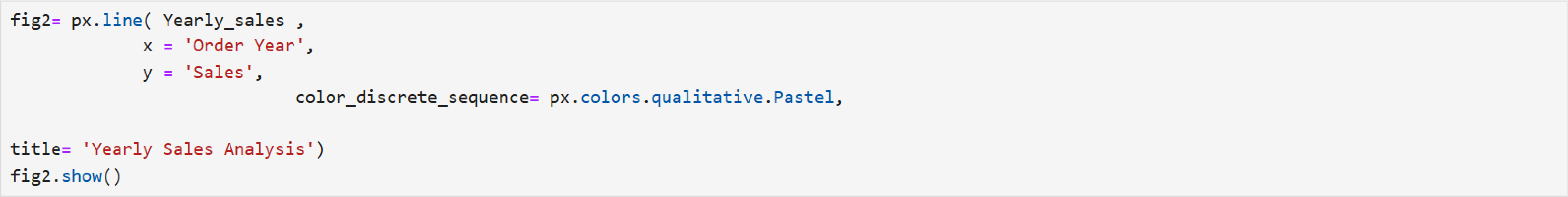


```python
Yearly_sales =df_esales.groupby('Order Year')['Sales'].sum().reset_index()
```

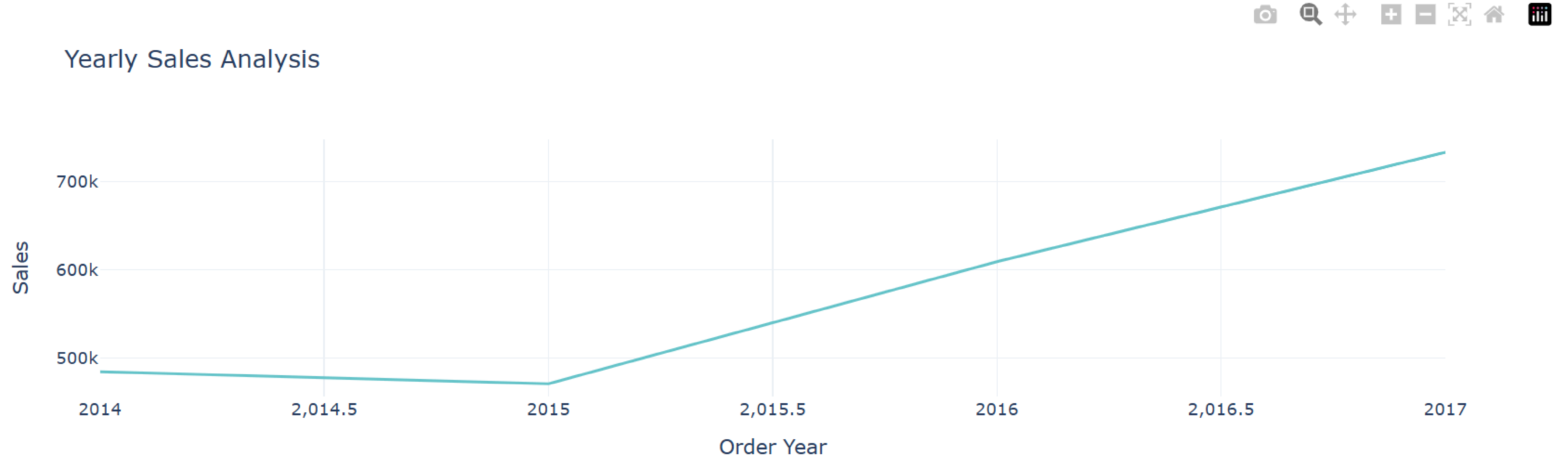
```python
Yearly_sales
```

```
   Order Year        Sales
0        2014  484247.4981
1        2015  470532.5090
2        2016  609205.5980
3        2017  733215.2552
```

Yearly Sales Analysis Using Line Chart

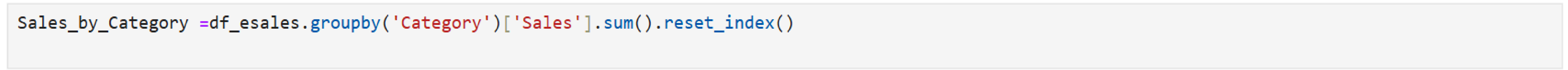


```
fig2= px.line( Yearly_sales ,
              x = 'Order Year',
              y = 'Sales',
                              color_discrete_sequence= px.colors.qualitative.Pastel,

title= 'Yearly Sales Analysis')
fig2.show()
```

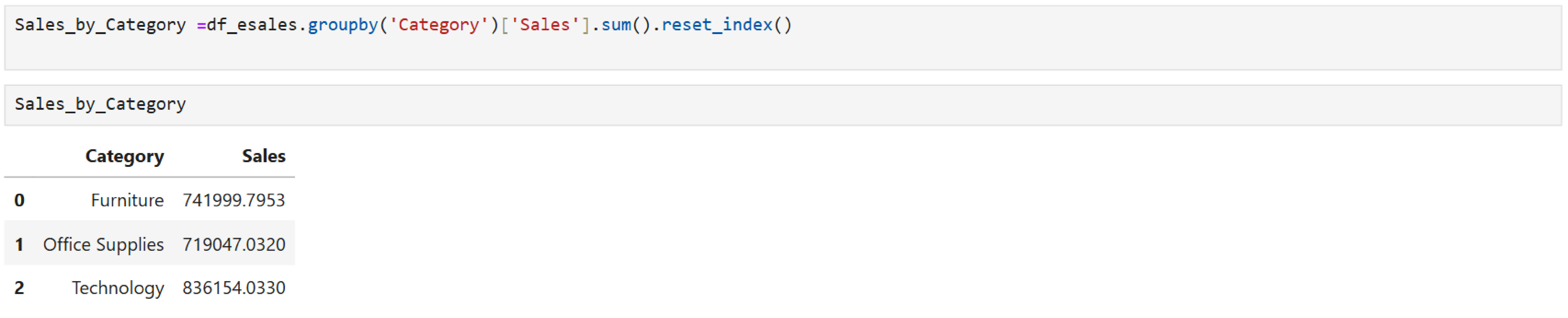


Insight: Sales has increased through years and marks the continuous growth of the business. .

Sales Analysis by Category



```
Sales_by_Category =df_esales.groupby('Category')['Sales'].sum().reset_index()
```

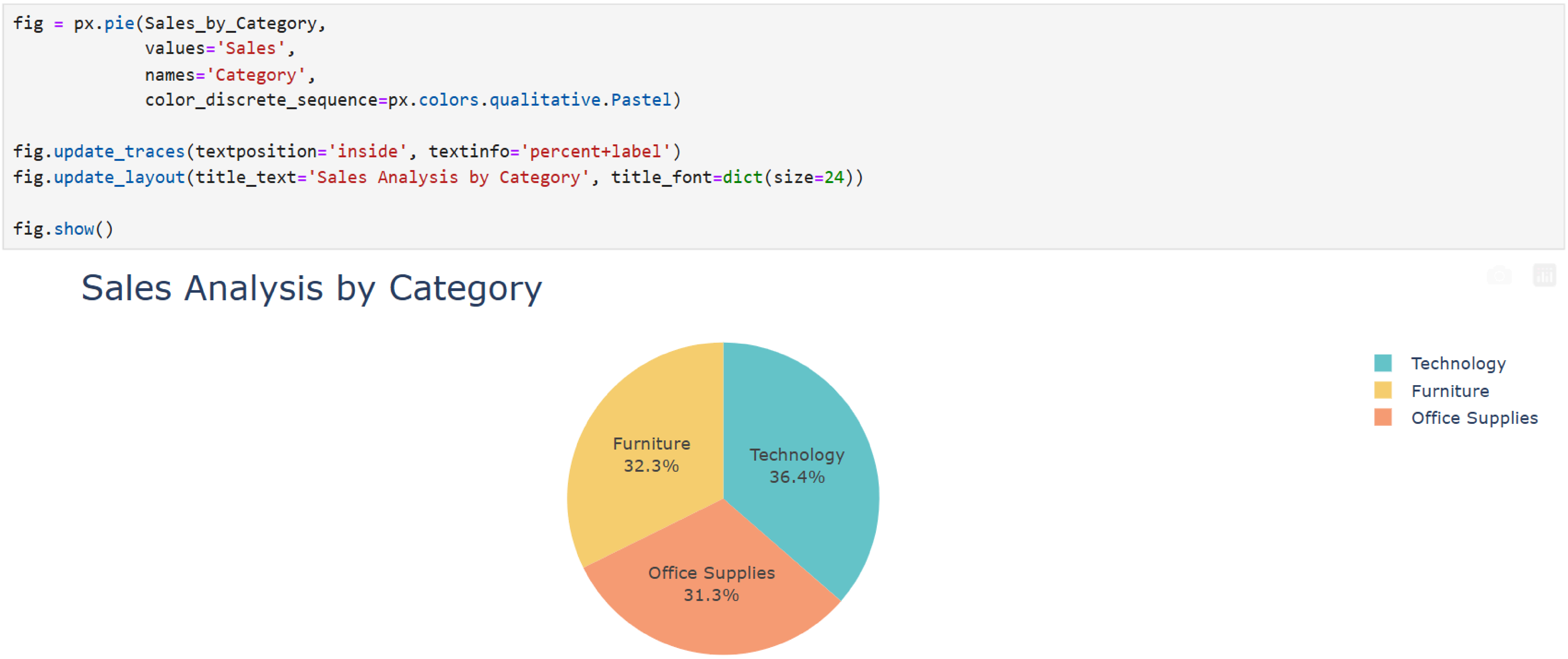


```
Sales_by_Category =df_esales.groupby('Category')['Sales'].sum().reset_index()
```

```
Sales_by_Category
```

	Category	Sales
0	Furniture	741999.7953
1	Office Supplies	719047.0320
2	Technology	836154.0330

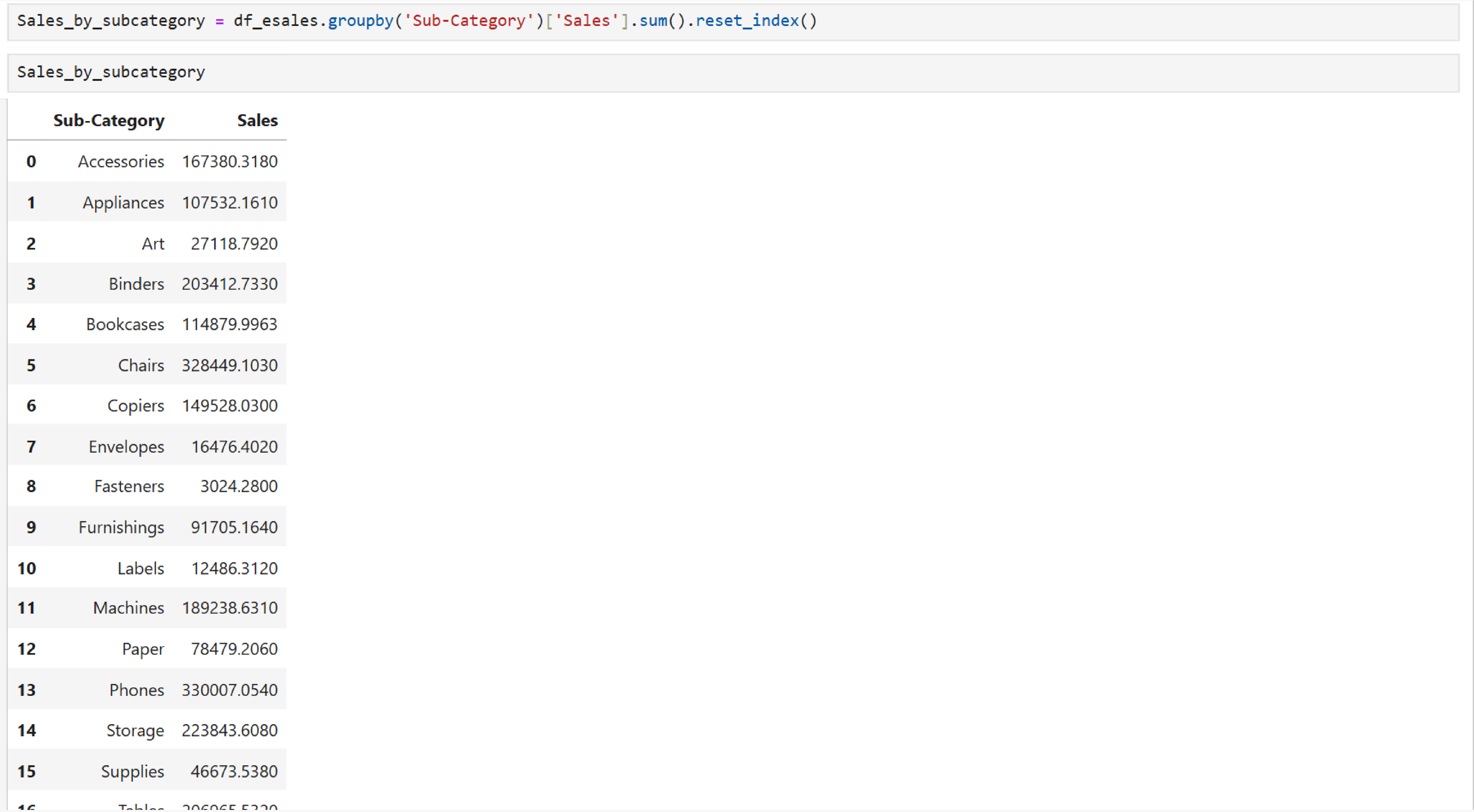
Sales Analysis by Category Using Pie-Chart



```
fig = px.pie(Sales_by_Category,
             values='Sales',
             names='Category',
             color_discrete_sequence=px.colors.qualitative.Pastel)

fig.update_traces(textposition='inside', textinfo='percent+label')
fig.update_layout(title_text='Sales Analysis by Category', title_font=dict(size=24))

fig.show()
```

Insight: The pie chart shows that Technology leads in sales at 36.4%, slightly ahead of Furniture (32.3%) and Office Supplies (31.3%), indicating a relatively balanced distribution across categories..

Sales_by_Sub-Category

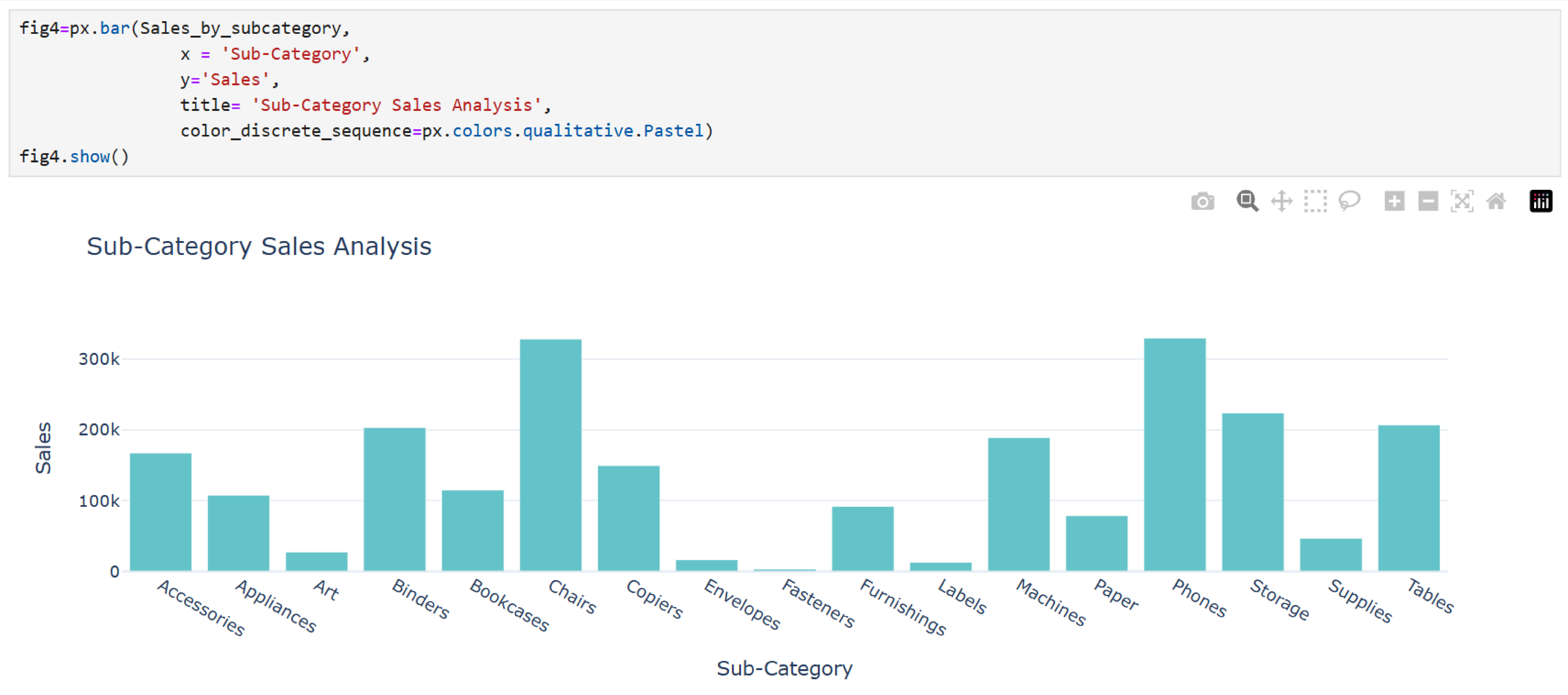


```
Sales_by_subcategory = df_esales.groupby('Sub-Category')['Sales'].sum().reset_index()
```

```
Sales_by_subcategory
```

	Sub-Category	Sales
0	Accessories	167380.3180
1	Appliances	107532.1610
2	Art	27118.7920
3	Binders	203412.7330
4	Bookcases	114879.9963
5	Chairs	328449.1030
6	Copiers	149528.0300
7	Envelopes	16476.4020
8	Fasteners	3024.2800
9	Furnishings	91705.1640
10	Labels	12486.3120
11	Machines	189238.6310
12	Paper	78479.2060
13	Phones	330007.0540
14	Storage	223843.6080
15	Supplies	46673.5380
16	Tables	206965.5320

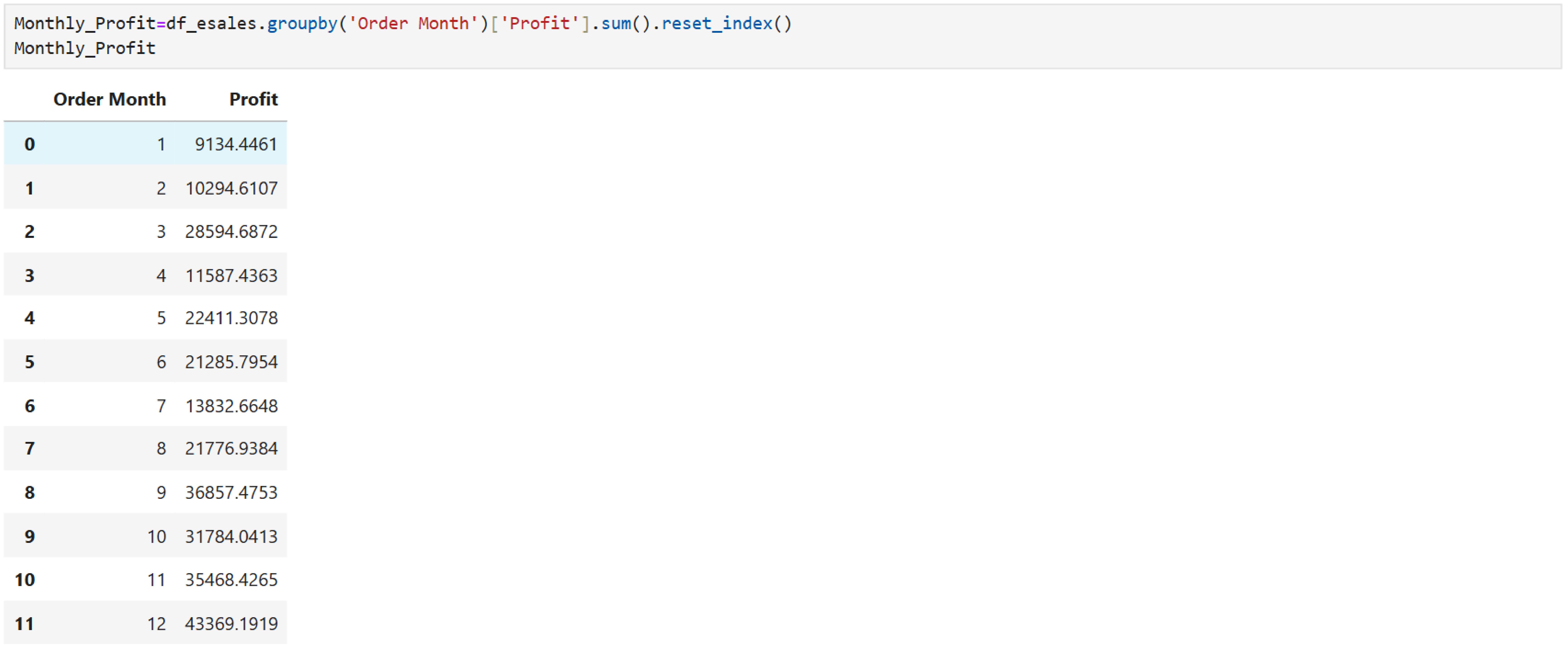
Sales Analysis By Sub- Category Using Bar Graph



```
fig4=px.bar(Sales_by_subcategory,
                x = 'Sub-Category',
                y='Sales',
                title= 'Sub-Category Sales Analysis',
                color_discrete_sequence=px.colors.qualitative.Pastel)
fig4.show()
```

Insight: Chairs and Paper dominate sub-category sales, while Fasteners and Art lag far behind, signaling priority and underperformance areas in product focus. .

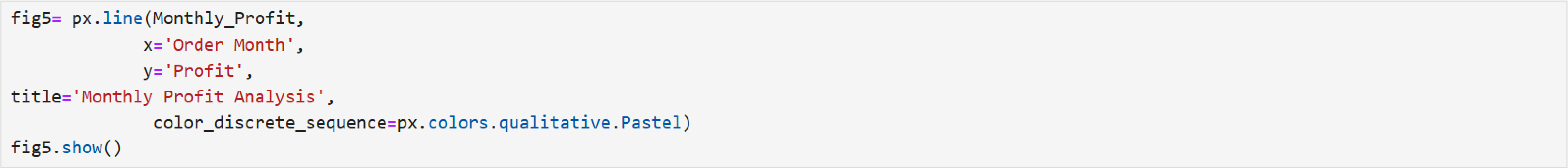
Monthly Profit



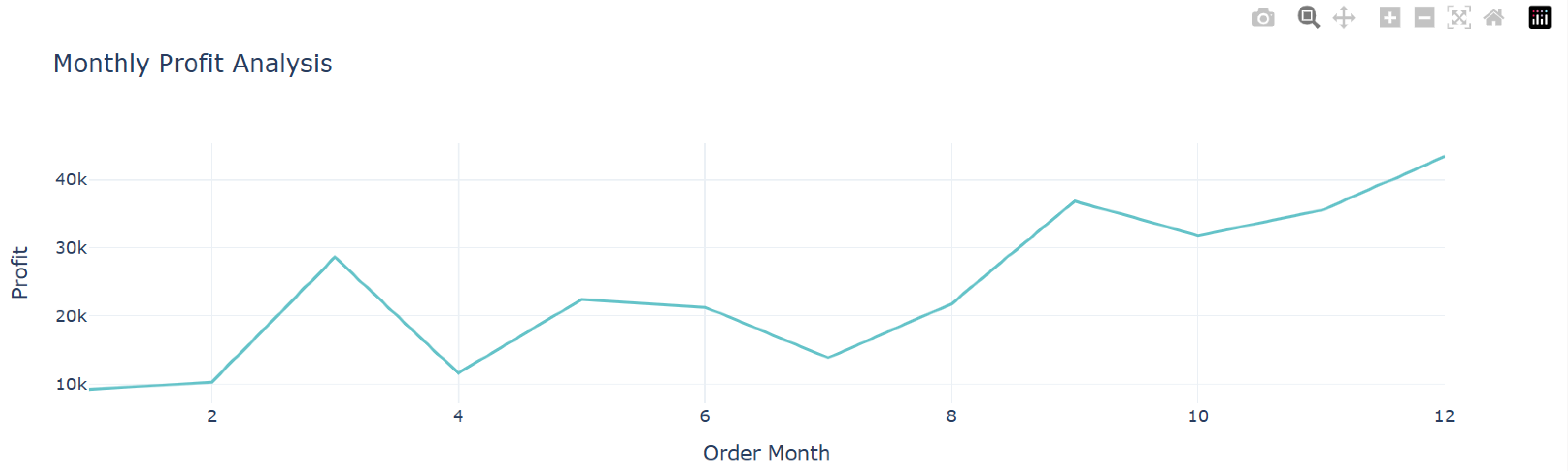
```
Monthly_Profit=df_esales.groupby('Order Month')['Profit'].sum().reset_index()
Monthly_Profit
```

	Order Month	Profit
0	1	9134.4461
1	2	10294.6107
2	3	28594.6872
3	4	11587.4363
4	5	22411.3078
5	6	21285.7954
6	7	13832.6648
7	8	21776.9384
8	9	36857.4753
9	10	31784.0413
10	11	35468.4265
11	12	43369.1919

Monthly Profit Analysis Using Line Chart

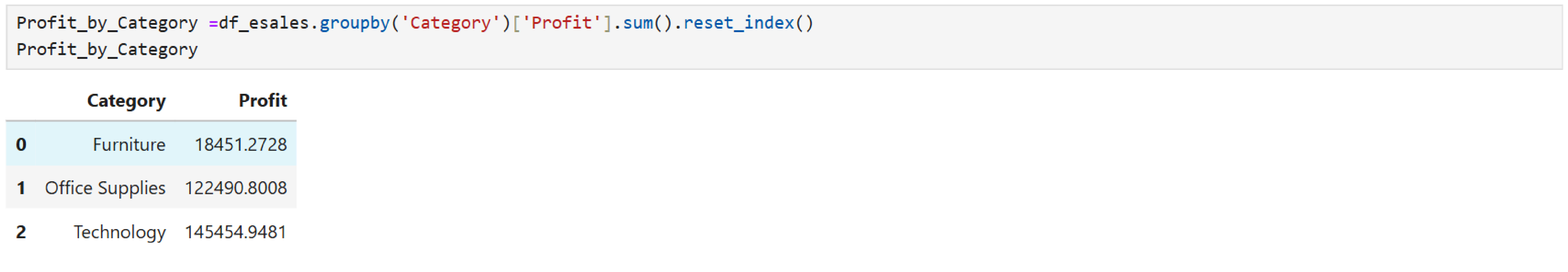


```
fig5= px.line(Monthly_Profit,
             x='Order Month',
             y='Profit',
title='Monthly Profit Analysis',
              color_discrete_sequence=px.colors.qualitative.Pastel)
fig5.show()
```



Insight: Profit has increased in the beginning of the year but took a downward trend during the mid-year then again took an upward trend.

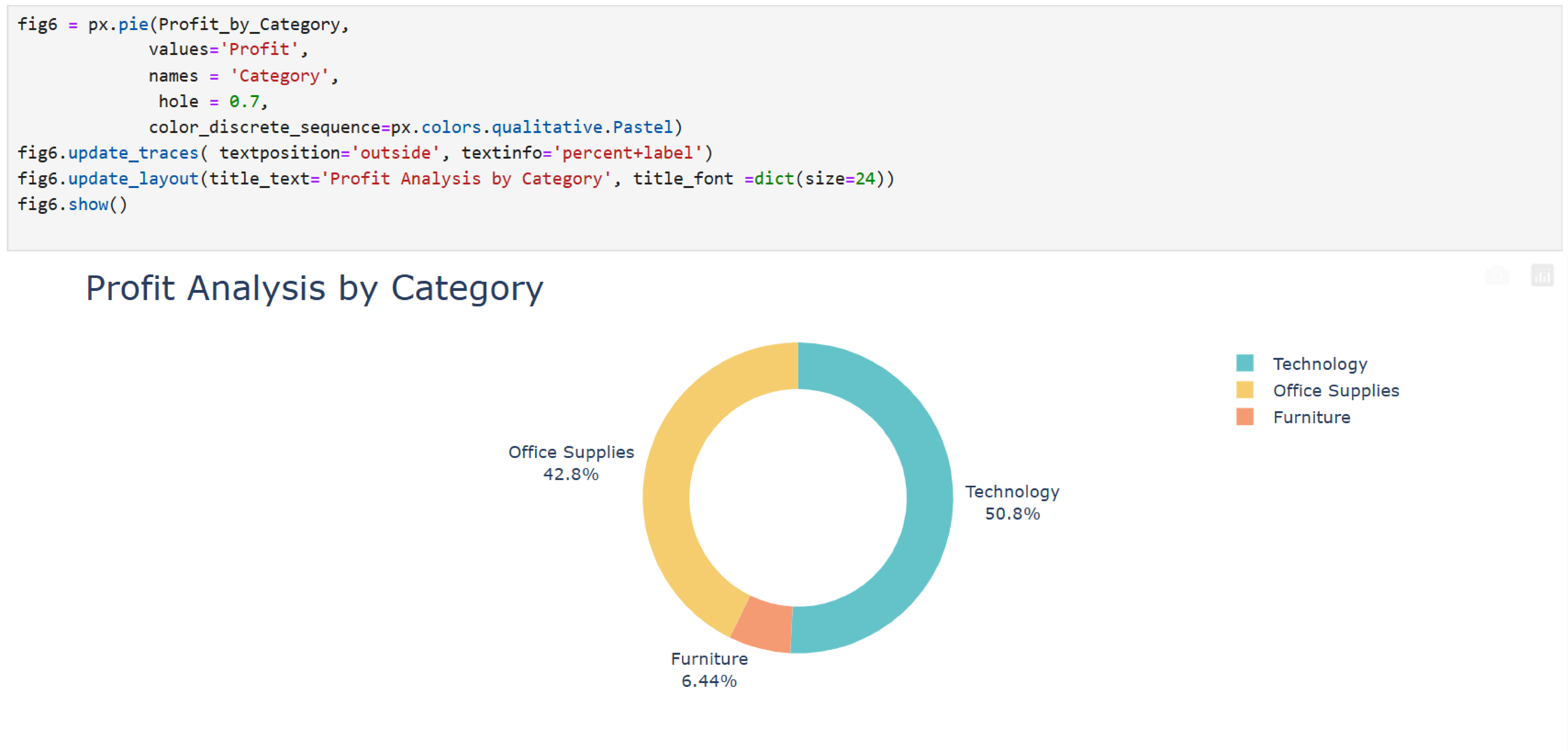
Profit By Category



```
Profit_by_Category =df_esales.groupby('Category')['Profit'].sum().reset_index()
Profit_by_Category
```

	Category	Profit
0	Furniture	18451.2728
1	Office Supplies	122490.8008
2	Technology	145454.9481

Profit Analysis By Category Using Donut Chart

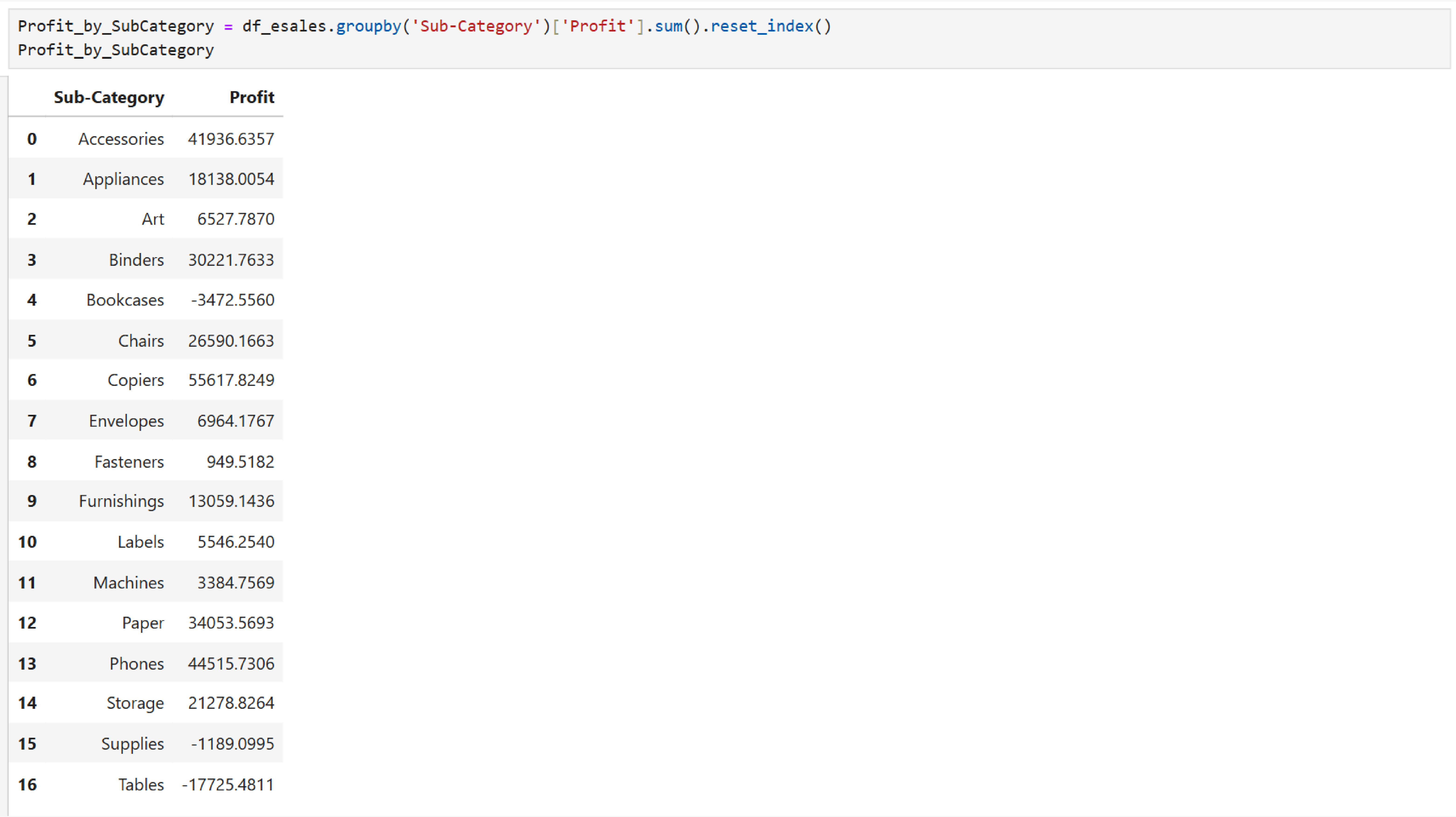


```
fig6 = px.pie(Profit_by_Category,
             values='Profit',
             names = 'Category',
              hole = 0.7,
             color_discrete_sequence=px.colors.qualitative.Pastel)
fig6.update_traces( textposition='outside', textinfo='percent+label')
fig6.update_layout(title_text='Profit Analysis by Category', title_font =dict(size=24))
fig6.show()
```

Profit Analysis by Category

Insight: Office Supplies and technology has contributed the most in the profit while Furniture has shown the least contribution.

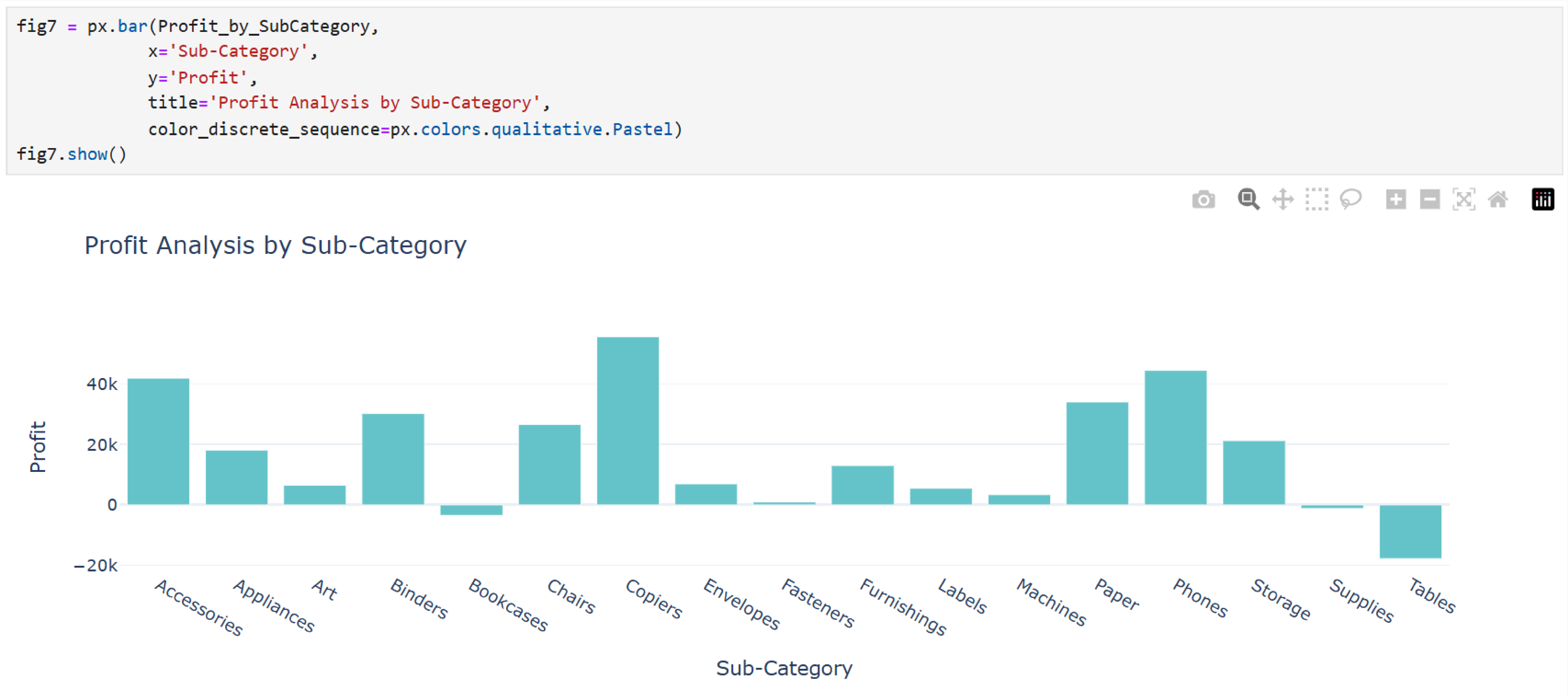
Profit By Sub-Category



```
Profit_by_SubCategory = df_esales.groupby('Sub-Category')['Profit'].sum().reset_index()
Profit_by_SubCategory
```

	Sub-Category	Profit
0	Accessories	41936.6357
1	Appliances	18138.0054
2	Art	6527.7870
3	Binders	30221.7633
4	Bookcases	-3472.5560
5	Chairs	26590.1663
6	Copiers	55617.8249
7	Envelopes	6964.1767
8	Fasteners	949.5182
9	Furnishings	13059.1436
10	Labels	5546.2540
11	Machines	3384.7569
12	Paper	34053.5693
13	Phones	44515.7306
14	Storage	21278.8264
15	Supplies	-1189.0995
16	Tables	-17725.4811

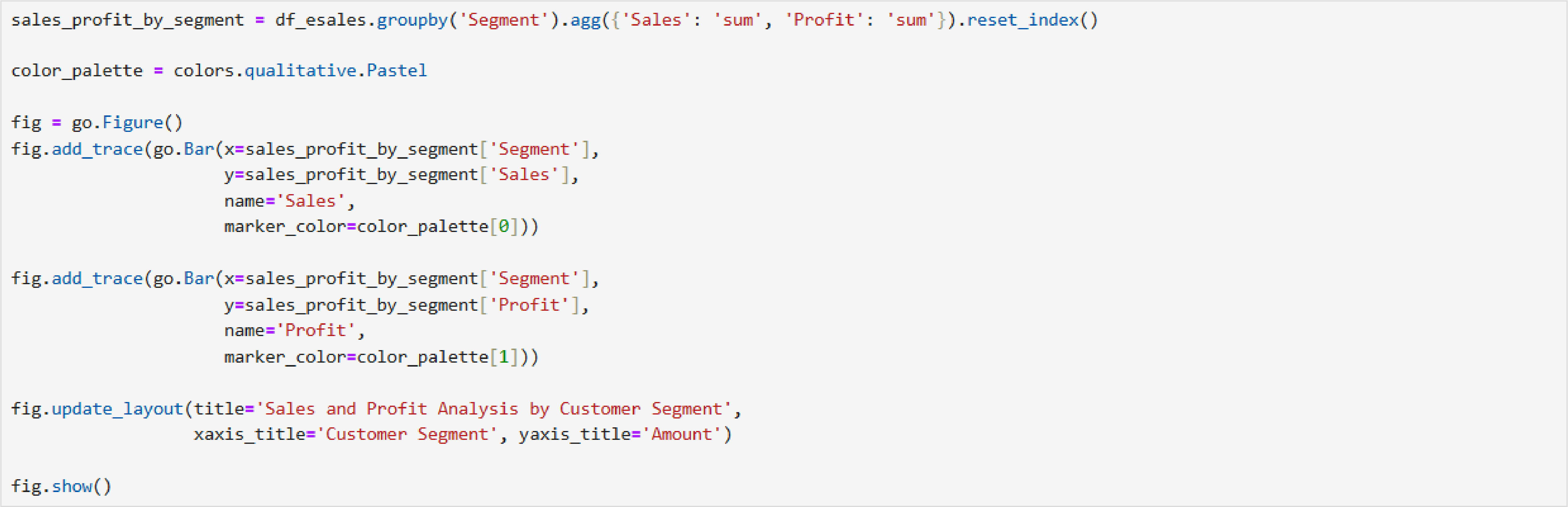
Profit Analysis By Sub-Category



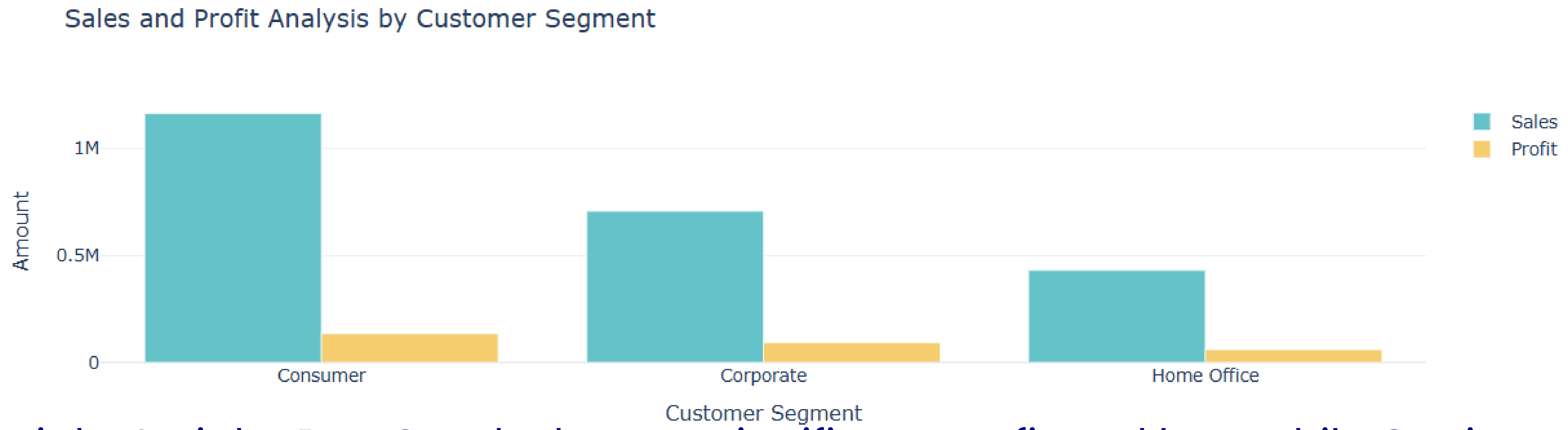
```
fig7 = px.bar(Profit_by_SubCategory,
             x='Sub-Category',
             y='Profit',
             title='Profit Analysis by Sub-Category',
             color_discrete_sequence=px.colors.qualitative.Pastel)
fig7.show()
```

Insight: Bar-Graph shows a significant profit and loss, while Copiers and Phones emerge as the most profitable sub-categories while other sub-categories need a check.

Sales and Profit Analysis by Customer Segment



```
sales_profit_by_segment = df_esales.groupby('Segment').agg({'Sales': 'sum', 'Profit': 'sum'}).reset_index()

color_palette = colors.qualitative.Pastel

fig = go.Figure()
fig.add_trace(go.Bar(x=sales_profit_by_segment['Segment'],
                     y=sales_profit_by_segment['Sales'],
                     name='Sales',
                     marker_color=color_palette[0]))

fig.add_trace(go.Bar(x=sales_profit_by_segment['Segment'],
                     y=sales_profit_by_segment['Profit'],
                     name='Profit',
                     marker_color=color_palette[1]))

fig.update_layout(title='Sales and Profit Analysis by Customer Segment',
                  xaxis_title='Customer Segment', yaxis_title='Amount')

fig.show()
```



Insight: Insight: Bar-Graph shows a significant profit and loss, while Copiers and Phones emerge as the most profitable sub-categories while other sub-categories need a check.

.

Course Of Action

1. Boost inventory and marketing during Q4 to leverage peak sales.
2. Focus on high-performing categories like Technology; improve or reposition low performers.
3. Promote top sub-categories (e.g., Chairs, Paper); reassess underperforming ones (e.g., Fasteners).
4. Stabilize mid-year profits by managing discounts and controlling operational costs.
5. Prioritize high-margin products (e.g., Copiers, Phones); minimize losses in low-profit items.
6. Target high-value customer segments with tailored campaigns and loyalty programs.

Thank You

Project by Himanshu.K