In [57]:
```
# Import libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
```

In [58]:
```
pd.options.display.max_columns = None
```

In [59]:
```
retail = pd.read_excel(r"C:\Users\USER\Downloads\Dataset\Retail-Supply-Chain-Sales-Dataset.xlsx", sheet_name = 1)
retail.head()
```

Out[59]:

	Row ID	Order ID	Order Date	MonthName	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State
0	1	CA-2016-152156	2016-08-11	Aug	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky
1	2	CA-2016-152156	2016-08-11	Aug	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky
2	3	CA-2016-138688	2016-12-06	Dec	2016-12-06	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California
3	4	US-2015-108966	2015-11-10	Nov	2015-11-10	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida
4	5	US-2015-108966	2015-11-10	Nov	2015-11-10	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida

In [60]:
```
retail[["Sales", "Quantity", "Discount", "Profit"]].describe()
```

Out[60]:

	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000
mean	229.858001	3.789574	0.156203	28.656896
std	623.245101	2.225110	0.206452	234.260108
min	0.444000	1.000000	0.000000	-6599.978000
25%	17.280000	2.000000	0.000000	1.728750
50%	54.490000	3.000000	0.200000	8.666500
75%	209.940000	5.000000	0.200000	29.364000
max	22638.480000	14.000000	0.800000	8399.976000

In [61]:
```
retail.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 24 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   MonthName            9994 non-null   object
 4   Ship Date            9994 non-null   datetime64[ns]
 5   Ship Mode            9994 non-null   object
 6   Customer ID          9994 non-null   object
 7   Customer Name        9994 non-null   object
 8   Segment              9994 non-null   object
 9   Country              9994 non-null   object
 10  City                 9994 non-null   object
 11  State                9994 non-null   object
 12  Postal Code          9994 non-null   int64
 13  Region               9994 non-null   object
 14  Retail Sales People  9994 non-null   object
 15  Product ID           9994 non-null   object
 16  Category             9994 non-null   object
 17  Sub-Category         9994 non-null   object
 18  Product Name         9994 non-null   object
 19  Returned             9994 non-null   object
 20  Sales                9994 non-null   float64
 21  Quantity             9994 non-null   int64
 22  Discount             9994 non-null   float64
 23  Profit               9994 non-null   float64
dtypes: datetime64[ns](2), float64(3), int64(3), object(16)
memory usage: 1.8+ MB
```

In [62]:
```
retail.columns
```

Out[62]:
```
Index(['Row ID', 'Order ID', 'Order Date', 'MonthName', 'Ship Date',
       'Ship Mode', 'Customer ID', 'Customer Name', 'Segment', 'Country',
       'City', 'State', 'Postal Code', 'Region', 'Retail Sales People',
       'Product ID', 'Category', 'Sub-Category', 'Product Name', 'Returned',
       'Sales', 'Quantity', 'Discount', 'Profit'],
      dtype='object')
```

In [63]:
```
retail.nunique()
```

Out[63]:
```
Row ID               9994
Order ID             5009
Order Date           1237
MonthName              12
Ship Date            1215
Ship Mode               4
Customer ID           793
Customer Name         793
Segment                 3
Country                 1
City                  531
State                  49
Postal Code           631
Region                  4
Retail Sales People     4
Product ID           1862
Category                3
Sub-Category           17
Product Name         1850
Returned                2
Sales                5825
Quantity               14
Discount               12
Profit               7287
dtype: int64
```

In [64]:
```
retail.duplicated()
```

Out[64]:
```
0       False
1       False
2       False
3       False
4       False
        ...
9989    False
9990    False
9991    False
9992    False
9993    False
Length: 9994, dtype: bool
```

In [65]:
```
retail.isnull().any()
```

Out[65]:
```
Row ID                False
Order ID              False
Order Date            False
MonthName             False
Ship Date             False
Ship Mode             False
Customer ID           False
Customer Name         False
Segment               False
Country               False
City                  False
State                 False
Postal Code           False
Region                False
Retail Sales People   False
Product ID            False
Category              False
Sub-Category          False
Product Name          False
Returned              False
Sales                 False
Quantity              False
Discount              False
Profit                False
dtype: bool
```

In [66]:
```
retail.isnull().sum()
```

Out[66]:
```
Row ID                0
Order ID              0
Order Date            0
MonthName             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
Retail Sales People   0
Product ID            0
Category              0
Sub-Category          0
Product Name          0
Returned              0
Sales                 0
Quantity              0
Discount              0
Profit                0
dtype: int64
```

In [67]:
```
retail["State"].unique()
```

Out[67]:
```
array(['Kentucky', 'California', 'Florida', 'North Carolina',
       'Washington', 'Texas', 'Wisconsin', 'Utah', 'Nebraska',
       'Pennsylvania', 'Illinois', 'Minnesota', 'Michigan', 'Delaware',
       'Indiana', 'New York', 'Arizona', 'Virginia', 'Tennessee',
       'Alabama', 'South Carolina', 'Oregon', 'Colorado', 'Iowa', 'Ohio',
       'Missouri', 'Oklahoma', 'New Mexico', 'Louisiana', 'Connecticut',
       'New Jersey', 'Massachusetts', 'Georgia', 'Nevada', 'Rhode Island',
       'Mississippi', 'Arkansas', 'Montana', 'New Hampshire', 'Maryland',
       'District of Columbia', 'Kansas', 'Vermont', 'Maine',
       'South Dakota', 'Idaho', 'North Dakota', 'Wyoming',
       'West Virginia'], dtype=object)
```

In [68]:
```
# Extract Day of week, month and year
retail["Order_DOW"] = retail["Order Date"].dt.day_name()
retail["Order_MonthName"] = retail["Order Date"].dt.month_name()
retail["Order_YearName"] = retail["Order Date"].dt.year
```

In [69]:
```
retail["State"].value_counts()
```

Out[69]:
```
State
California              2001
New York                1128
Texas                    985
Pennsylvania             587
Washington               506
Illinois                 492
Ohio                     469
Florida                  383
Michigan                 255
North Carolina           249
Arizona                  224
Virginia                 224
Georgia                  184
Tennessee                183
Colorado                 182
Indiana                  149
Kentucky                 139
Massachusetts            135
New Jersey               130
Oregon                   124
Wisconsin                110
Maryland                 105
Delaware                  96
Minnesota                 89
Connecticut               82
Missouri                  66
Oklahoma                  66
Alabama                   61
Arkansas                  60
Rhode Island              56
Utah                      53
Mississippi               53
South Carolina            42
Louisiana                 42
Nevada                    39
Nebraska                  38
New Mexico                37
Iowa                      30
New Hampshire             27
Kansas                    24
Idaho                     21
Montana                   15
South Dakota              12
Vermont                   11
District of Columbia      10
Maine                      8
North Dakota               7
West Virginia              4
Wyoming                    1
Name: count, dtype: int64
```

In [70]:
```
#  Total Sales, Profit

total_sales = retail["Sales"].sum()
total_profit = retail["Profit"].sum()
avg_sales = retail["Sales"].mean()
avg_profit = retail["Profit"].mean()
min_sales = retail["Sales"].min()
max_sales = retail["Sales"].max()
min_profit = retail["Profit"].min()
max_profit = retail["Profit"].max()
print(f"Total Sales: {total_sales}")
print(f"Total Profit: {total_profit}")
print(f"Average Sales: {avg_sales}")
print(f"Average Profit: {avg_profit}")
print(f"Minimum Sales: {min_sales}")
print(f"Maximum Sales: {max_sales}")
print(f"Minimum Profit: {min_profit}")
print(f"Maximum Profit: {max_profit}")
```

```
Total Sales: 2297200.8603000003
Total Profit: 286397.0217
Average Sales: 229.85800083049833
Average Profit: 28.65689630778467
Minimum Sales: 0.444
Maximum Sales: 22638.48
Minimum Profit: -6599.978
Maximum Profit: 8399.976
```

Customer Analysis

In [71]:
```
# Total Sales and count of sales by segment

grouped_segment = retail.groupby("Segment")["Sales"].agg(sum = "sum", count = "count").sort_values(by = "count", ascending = F
grouped_segment
```

Out[71]:

Segment	sum	count
Consumer	1.161401e+06	5191
Corporate	7.061464e+05	3020
Home Office	4.296531e+05	1783

In [72]:
```
# Top Customers

retail.groupby("Customer Name")["Sales"].sum().nlargest(15)
```

Out[72]:
```
Customer Name
Sean Miller           25043.050
Tamara Chand          19052.218
Raymond Buch          15117.339
Tom Ashbrook          14595.620
Adrian Barton         14473.571
Ken Lonsdale          14175.229
Sanjit Chand          14142.334
Hunter Lopez          12873.298
Sanjit Engle          12209.438
Christopher Conant    12129.072
Todd Sumrall          11891.751
Greg Tran             11820.120
Becky Martin          11789.630
Seth Vernon           11470.950
Caroline Jumper       11164.974
Name: Sales, dtype: float64
```

In [73]:
```
retail.columns
```

Out[73]:
```
Index(['Row ID', 'Order ID', 'Order Date', 'MonthName', 'Ship Date',
       'Ship Mode', 'Customer ID', 'Customer Name', 'Segment', 'Country',
       'City', 'State', 'Postal Code', 'Region', 'Retail Sales People',
       'Product ID', 'Category', 'Sub-Category', 'Product Name', 'Returned',
       'Sales', 'Quantity', 'Discount', 'Profit', 'Order_DOW',
       'Order_MonthName', 'Order_YearName'],
      dtype='object')
```

In [74]:
```
retail.groupby("Order_YearName")["Sales"].sum()
```

Out[74]:
```
Order_YearName
2014    484247.4981
2015    470532.5090
2016    609205.5980
2017    733215.2552
Name: Sales, dtype: float64
```

In [75]:
```
# Top Customers in 2016
filtered_year = retail[retail["Order_YearName"] == 2016].reset_index()
total_2016_sales = filtered_year.groupby("Segment")["Sales"].sum()
top_10_customers_2016 = filtered_year.groupby("Customer Name")["Sales"].sum().nlargest(10)
print("__________ 2016 Buyers Segments _____________")
print(total_2016_sales)
print("__________ Top 10 Customers in 2016 _____________")
print(top_10_customers_2016)
```

```
__________ 2016 Buyers Segments _____________
Segment
Consumer       296863.8992
Corporate      207106.3618
Home Office    105235.3370
Name: Sales, dtype: float64
__________ Top 10 Customers in 2016 _____________
Customer Name
Tamara Chand          18344.052
Christopher Conant    11901.184
Adrian Barton         10403.865
Sanjit Engle           9879.220
Bill Shonely           9199.780
Edward Hooks           7937.512
Joe Elijah             7587.971
Laura Armstrong        6451.222
Karen Daniels          6317.162
Jonathan Doherty       6181.108
Name: Sales, dtype: float64
```

In [76]:
```
retail.groupby("Segment")["Sales"].sum()
```

Out[76]:
```
Segment
Consumer       1.161401e+06
Corporate      7.061464e+05
Home Office    4.296531e+05
Name: Sales, dtype: float64
```

In [77]:
```
# Profit made by segment

retail.groupby("Segment")["Profit"].sum()
```

Out[77]:
```
Segment
Consumer       134119.2092
Corporate       91979.1340
Home Office     60298.6785
Name: Profit, dtype: float64
```

In [78]:
```
# Quantity bought by segment
retail.groupby("Segment")["Quantity"].sum()
```

Out[78]:
```
Segment
Consumer       19521
Corporate      11608
Home Office     6744
Name: Quantity, dtype: int64
```

In [79]:
```
retail["Segment"].value_counts()
```

Out[79]:
```
Segment
Consumer       5191
Corporate      3020
Home Office    1783
Name: count, dtype: int64
```

In [80]:
```
retail.groupby("Product Name")["Sales"].agg(sum = "sum", count = "count", mean = "mean").sort_values(by = "count", ascending =
```

Out[80]:

Product Name	sum	count	mean
Staple envelope	1686.812	48	35.141917
Staples	755.470	46	16.423261
Easy-staple paper	2504.192	46	54.438957
Avery Non-Stick Binders	217.316	20	10.865800
Staples in misc. colors	478.812	19	25.200632
...	...	...	...
Avaya IP Phone 1140E VoIP phone	1394.950	1	1394.950000
Avery 484	9.216	1	9.216000
Plantronics Single Ear Headset	29.925	1	29.925000
Zebra GK420t Direct Thermal/Thermal Transfer Printer	703.710	1	703.710000
Xiaomi Mi3	2279.960	1	2279.960000

1850 rows × 3 columns

In [81]:
```
# Total sales and total profit. Count of sales and mean of sales and profit

retail.groupby("Category")[["Sales", "Profit"]].agg({
    "Sales" : ["sum", "count", "mean"],
    "Profit" : ["sum", "mean"]}).sort_values(by = ("Sales", "sum"), ascending = False)
```

Out[81]:

	Sales			Profit	
	sum	count	mean	sum	mean
Category					
Technology	836154.0330	1847	452.709276	145454.9481	78.752002
Furniture	741999.7953	2121	349.834887	18451.2728	8.699327
Office Supplies	719047.0320	6026	119.324101	122490.8008	20.327050

In [82]:
```
retail.head()
```

Out[82]:

	Row ID	Order ID	Order Date	MonthName	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State
0	1	CA-2016-152156	2016-08-11	Aug	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky
1	2	CA-2016-152156	2016-08-11	Aug	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky
2	3	CA-2016-138688	2016-12-06	Dec	2016-12-06	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California
3	4	US-2015-108966	2015-11-10	Nov	2015-11-10	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida
4	5	US-2015-108966	2015-11-10	Nov	2015-11-10	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida

In [83]:
```
retail.groupby(["Order_DOW", "Category"])[["Sales", "Profit"]].sum()
```

Out[83]:

Order_DOW	Category	Sales	Profit
Friday	**Furniture**	120899.3643	3824.9747
	Office Supplies	134530.0930	23924.3926
	Technology	149802.6980	20326.4988
Monday	**Furniture**	131494.0757	5641.9928
	Office Supplies	139214.5620	30897.7474
	Technology	120570.9020	16746.6630
Saturday	**Furniture**	85010.7562	1736.8025
	Office Supplies	108909.4700	15520.7406
	Technology	115873.4360	22401.5244
Sunday	**Furniture**	125872.4151	1387.1105
	Office Supplies	108833.6830	11881.5902
	Technology	137011.7030	30858.8447
Thursday	**Furniture**	119825.4634	817.7114
	Office Supplies	91075.9420	14553.9845
	Technology	93857.0940	23505.6445
Tuesday	**Furniture**	99920.8788	2551.4516
	Office Supplies	94230.0940	19981.8932
	Technology	129702.0530	10810.3468
Wednesday	**Furniture**	58976.8418	2491.2293
	Office Supplies	42253.1880	5730.4523
	Technology	89336.1470	20805.4259

In [84]:
```
# Daily sales and profit by category

retail.groupby(["Order_DOW", "Category"])[["Sales", "Profit"]].sum().unstack()
```

Out[84]:

	Sales			Profit		
Category	**Furniture**	**Office Supplies**	**Technology**	**Furniture**	**Office Supplies**	**Technology**
Order_DOW						
Friday	120899.3643	134530.093	149802.698	3824.9747	23924.3926	20326.4988
Monday	131494.0757	139214.562	120570.902	5641.9928	30897.7474	16746.6630
Saturday	85010.7562	108909.470	115873.436	1736.8025	15520.7406	22401.5244
Sunday	125872.4151	108833.683	137011.703	1387.1105	11881.5902	30858.8447
Thursday	119825.4634	91075.942	93857.094	817.7114	14553.9845	23505.6445
Tuesday	99920.8788	94230.094	129702.053	2551.4516	19981.8932	10810.3468
Wednesday	58976.8418	42253.188	89336.147	2491.2293	5730.4523	20805.4259

In [85]:
```
sales_subCategory = retail.groupby("Sub-Category")["Sales"].sum().sort_values(ascending = False)
total_sales = retail["Sales"].sum()
percentSalesSubCat = (sales_subCategory / total_sales) * 100
result_df = pd.DataFrame({
    "Sub-Category": sales_subCategory.index,
    "Total Sales": sales_subCategory.values,
    "Percentage of Total (%)": percentSalesSubCat.values
}).reset_index(drop=True)
```

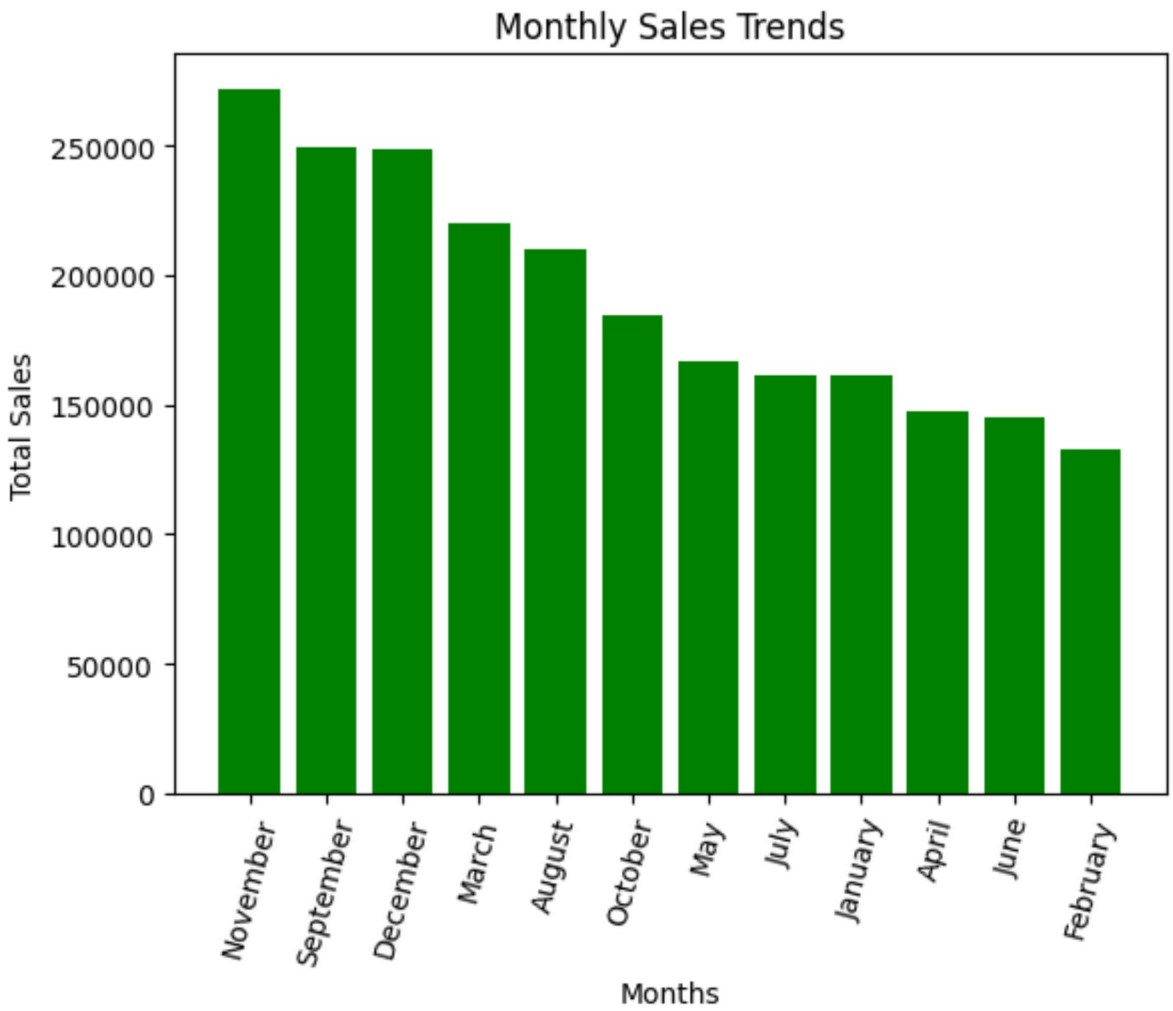
```
print(result_df)
```

```
   Sub-Category  Total Sales  Percentage of Total (%)
0        Phones  330007.0540                14.365616
1        Chairs  328449.1030                14.297796
2       Storage  223843.6080                 9.744190
3        Tables  206965.5320                 9.009466
4       Binders  203412.7330                 8.854808
5      Machines  189238.6310                 8.237792
6   Accessories  167380.3180                 7.286273
7       Copiers  149528.0300                 6.509140
8     Bookcases  114879.9963                 5.000869
9    Appliances  107532.1610                 4.681008
10  Furnishings   91705.1640                 3.992039
11        Paper   78479.2060                 3.416297
12     Supplies   46673.5380                 2.031757
13          Art   27118.7920                 1.180515
14    Envelopes   16476.4020                 0.717238
15       Labels   12486.3120                 0.543545
16    Fasteners    3024.2800                 0.131651
```

In [86]:
```
# Monthly Sales

monthlySales = round(retail.groupby("Order_MonthName")["Sales"].sum().sort_values(ascending = False), 2)
print(monthlySales)
```

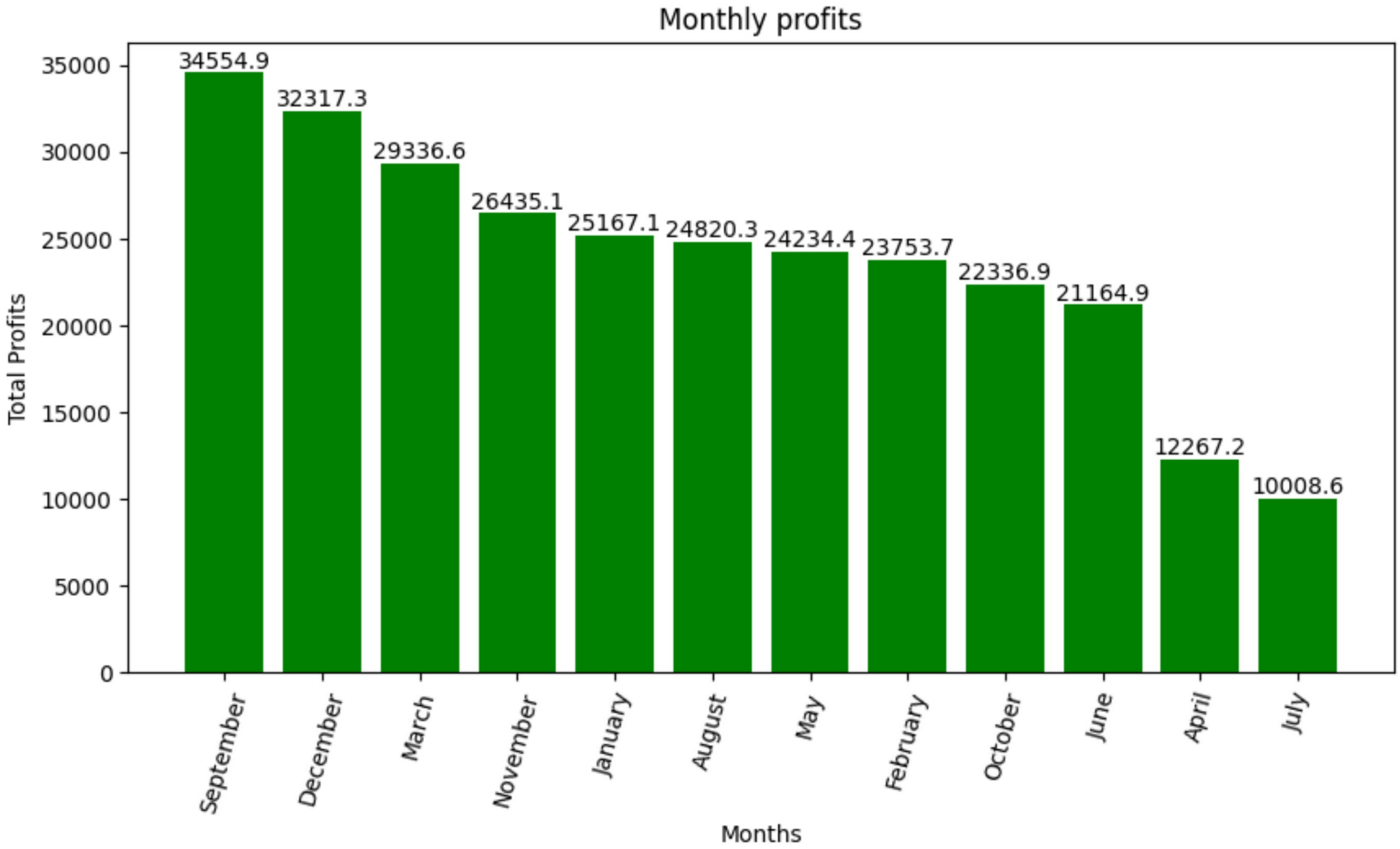
```
Order_MonthName
November     271693.75
September    248989.30
December     248765.33
March        220064.65
August       209964.37
October      184356.33
May          166420.32
July         161227.10
January      161083.59
April        147031.26
June         144883.50
February     132721.36
Name: Sales, dtype: float64
```

In [87]:
```
monthlySales = retail.groupby("Order_MonthName")["Sales"].sum().sort_values(ascending = False).round(2).reset_index()
monthlySales.columns = ["Order_MonthName", "Sales"]

plt.bar(monthlySales["Order_MonthName"], monthlySales["Sales"], color = "Green")
plt.title("Monthly Sales Trends")
plt.xlabel("Months")
plt.ylabel("Total Sales")
plt.xticks(rotation = 75)
plt.show()
```



In [88]:
```
# Montly Profit

profitMonthlyView = retail.groupby("Order_MonthName")["Profit"].sum().sort_values(ascending = False).round(2)
print(profitMonthlyView)

print("_____________ Visualization _____________")

monthlyProfit = retail.groupby("Order_MonthName")["Profit"].sum().sort_values(ascending = False).round(2).reset_index()
monthlyProfit.columns = ["Order_MonthName", "Profit"]

fig, ax = plt.subplots(figsize = (10,5))
bars = ax.bar(monthlyProfit["Order_MonthName"], monthlyProfit["Profit"], color = "green")
ax.bar_label(bars)
plt.title("Monthly profits")
plt.xlabel("Months")
plt.ylabel("Total Profits")
plt.xticks(rotation = 75)
plt.show()
```

```
Order_MonthName
September    34554.88
December     32317.30
March        29336.59
November     26435.08
January      25167.10
August       24820.34
May          24234.41
February     23753.65
October      22336.93
June         21164.90
April        12267.20
July         10008.65
Name: Profit, dtype: float64
_____________ Visualization _____________
```

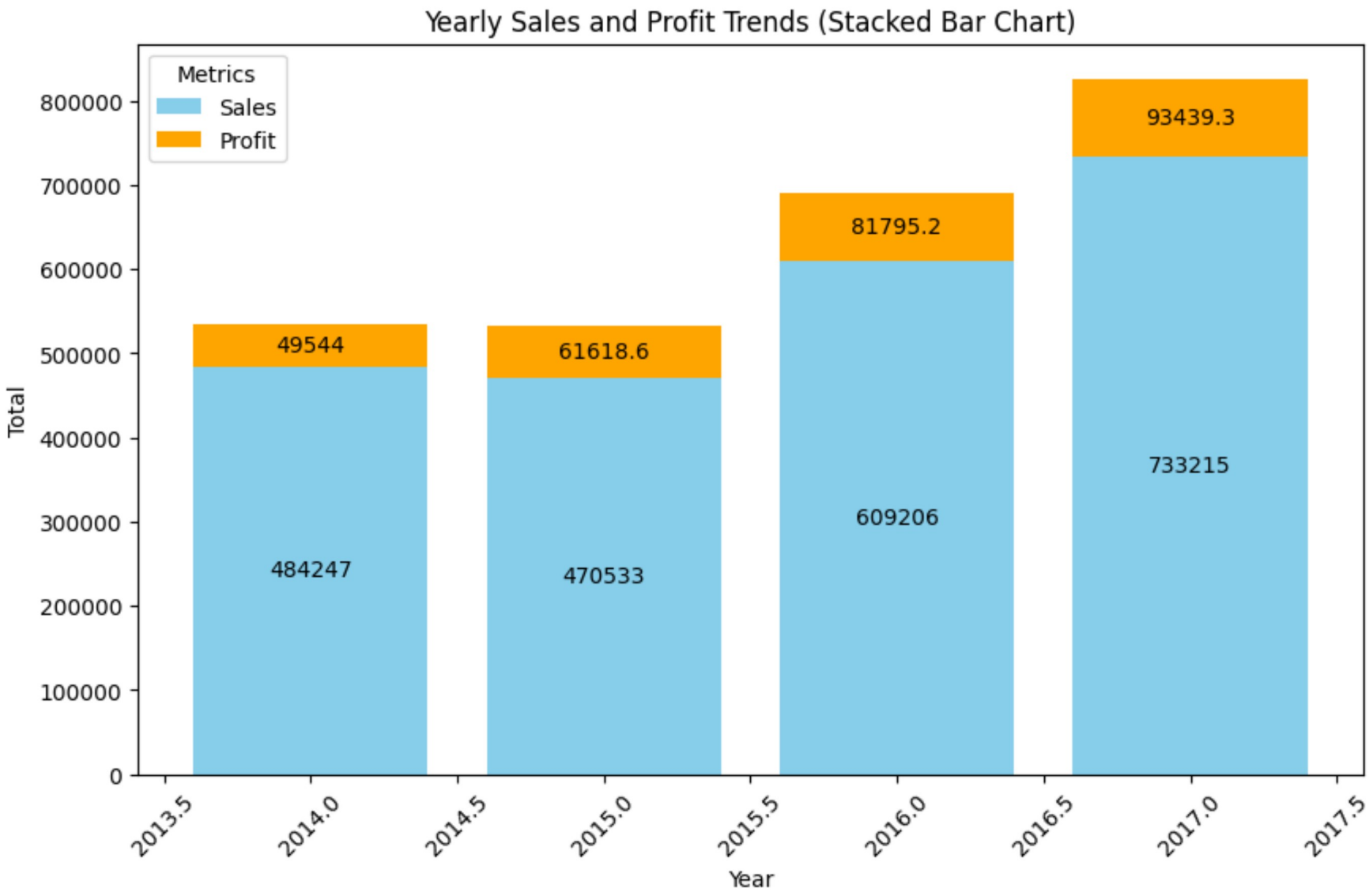


In [89]:
```
# Yearly Sales

retail.groupby("Order_YearName")[["Sales", "Profit"]].sum()
```

Out[89]:

Order_YearName	Sales	Profit
2014	484247.4981	49543.9741
2015	470532.5090	61618.6037
2016	609205.5980	81795.1743
2017	733215.2552	93439.2696

In [90]:
```
# yearly sales and profit stacked barchart

yearlySales = retail.groupby("Order_YearName")[["Sales", "Profit"]].sum().reset_index()
yearlySales.columns = ["Order_YearName", "Sales", "Profit"]

fig, ax = plt.subplots(figsize=(10, 6))
bars1 = ax.bar(yearlySales["Order_YearName"], yearlySales["Sales"], label="Sales", color="skyblue")
bars2 = ax.bar(yearlySales["Order_YearName"], yearlySales["Profit"], bottom=yearlySales["Sales"], label="Profit", color="orang

ax.bar_label(bars1, label_type = "center")
ax.bar_label(bars2, label_type = "center")

plt.title("Yearly Sales and Profit Trends (Stacked Bar Chart)")
plt.xlabel("Year")
plt.ylabel("Total")
plt.legend(title = "Metrics")
plt.xticks(rotation = 45)
plt.show()
```



Profit

In [91]:
```
# Category with most profit
catProfit = retail.groupby("Category")["Profit"].sum().sort_values(ascending = False)
catProfit
```

Out[91]:
```
Category
Technology         145454.9481
Office Supplies    122490.8008
Furniture           18451.2728
Name: Profit, dtype: float64
```

In [92]:
```
# Profit by sub-category

subCatProfit = retail.groupby("Sub-Category")["Profit"].sum().sort_values(ascending = False)
subCatProfit
```

Out[92]:
```
Sub-Category
Copiers        55617.8249
Phones         44515.7306
Accessories    41936.6357
Paper          34053.5693
Binders        30221.7633
Chairs         26590.1663
Storage        21278.8264
Appliances     18138.0054
Furnishings    13059.1436
Envelopes       6964.1767
Art             6527.7870
Labels          5546.2540
Machines        3384.7569
Fasteners        949.5182
Supplies       -1189.0995
Bookcases      -3472.5560
Tables        -17725.4811
Name: Profit, dtype: float64
```

In [93]:
```
# Shape of zero profit and loss

row_columns_no_low_profit = retail[retail["Profit"] <= 0].shape
print(row_columns_no_low_profit)

# Zero profit and loss
zero_loss = retail[retail["Profit"] <= 0][["Sub-Category", "Profit"]]
zero_loss
```

```
(1936, 27)
```

Out[93]:

	Sub-Category	Profit
3	Tables	-383.0310
14	Appliances	-123.8580
15	Binders	-3.8160
23	Chairs	-1.0196
27	Bookcases	-1665.0522
...	...	...
9921	Binders	-4.5936
9931	Bookcases	-40.1960
9937	Tables	-1.7772
9962	Bookcases	-67.6704
9977	Fasteners	0.0000

1936 rows × 2 columns

In [94]:
```
# Discount effects on Profit

discount_profit = retail.groupby("Discount")["Profit"].mean().sort_values()
discount_profit
```

Out[94]:
```
Discount
0.50   -310.703456
0.45   -226.646464
0.40   -111.927429
0.80   -101.796797
0.70    -95.874060
0.32    -88.560656
0.30    -45.679636
0.60    -43.077212
0.20     24.702572
0.15     27.288298
0.00     66.900292
0.10     96.055074
Name: Profit, dtype: float64
```

Shipping Analysis

In [95]:
```
# Shipping Time
retail["Ship Time"] = (retail["Ship Date"] - retail["Order Date"]).dt.days
average_shipping_time = retail.groupby("Ship Mode")["Ship Time"].mean().sort_values(ascending = False)
average_shipping_time
```

Out[95]:
```
Ship Mode
Standard Class    41.850871
Second Class      30.559383
First Class       23.537711
Same Day           0.880295
Name: Ship Time, dtype: float64
```

Product Performance

In [96]:
```
# Top Most returned products

retail[retail["Returned"] == "Yes"].groupby("Product Name")["Returned"].count().nlargest(20)
```

Out[96]:
```
Product Name
KI Adjustable-Height Table                                  4
Staple envelope                                             4
Advantus Push Pins                                          3
Avery Durable Binders                                       3
Fellowes Black Plastic Comb Bindings                        3
Fellowes Strictly Business Drawer File, Letter/Legal Size   3
Ibico Standard Transparent Covers                           3
Longer-Life Soft White Bulbs                                3
Novimex Turbo Task Chair                                    3
Plantronics Encore H101 Dual Earpieces Headset              3
Prang Drawing Pencil Set                                    3
Stanley Bostitch Contemporary Electric Pencil Sharpeners    3
Staple holder                                               3
Staple remover                                              3
Staple-based wall hangings                                  3
Tenex B1-RE Series Chair Mats for Low Pile Carpets          3
Wilson Jones Clip & Carry Folder Binder Tool for Ring Binders, Clear  3
Wilson Jones Easy Flow II Sheet Lifters                     3
Wilson Jones Hanging View Binder, White, 1"                 3
Wirebound Service Call Books, 5 1/2" x 4"                   3
Name: Returned, dtype: int64
```

In [97]:
```
# Loss caused by return

loss_due_returns = retail[retail["Returned"] == "Yes"]["Profit"].sum()
loss_due_returns
```

Out[97]:
```
np.float64(23232.3615)
```

Discount and Sales

In [98]:
```
discount_sales = retail.groupby("Discount")["Sales"].sum()
print(discount_sales)

discount_profit = retail.groupby("Discount")["Profit"].sum()
print(discount_profit)
```

```
Discount
0.00    1.087908e+06
0.10    5.436935e+04
0.15    2.755852e+04
0.20    7.645944e+05
0.30    1.032267e+05
0.32    1.449346e+04
0.40    1.164178e+05
0.45    5.484974e+03
0.50    5.891854e+04
0.60    6.644700e+03
0.70    4.062028e+04
0.80    1.696376e+04
Name: Sales, dtype: float64
Discount
0.00    320987.6032
0.10      9029.1770
0.15      1418.9915
0.20     90337.3060
0.30    -10369.2774
0.32     -2391.1377
0.40    -23057.0504
0.45     -2493.1111
0.50    -20506.4281
0.60     -5944.6552
0.70    -40075.3569
0.80    -30539.0392
Name: Profit, dtype: float64
```

States & City

In [99]:
```
# States

retail.groupby("State")[["Sales", "Profit"]].sum().sort_values(by = "Sales", ascending = False)
```

Out[99]:

State	Sales	Profit
California	457687.6315	76381.3871
New York	310876.2710	74038.5486
Texas	170188.0458	-25729.3563
Washington	138641.2700	33402.6517
Pennsylvania	116511.9140	-15559.9603
Florida	89473.7080	-3399.3017
Illinois	80166.1010	-12607.8870
Ohio	78258.1360	-16971.3766
Michigan	76269.6140	24463.1876
Virginia	70636.7200	18597.9504
North Carolina	55603.1640	-7490.9122
Indiana	53555.3600	18382.9363
Georgia	49095.8400	16250.0433
Kentucky	36591.7500	11199.6966
New Jersey	35764.3120	9772.9138
Arizona	35282.0010	-3427.9246
Wisconsin	32114.6100	8401.8004
Colorado	32108.1180	-6527.8579
Tennessee	30661.8730	-5341.6936
Minnesota	29863.1500	10823.1874
Massachusetts	28634.4340	6785.5016
Delaware	27451.0690	9977.3748
Maryland	23705.5230	7031.1788
Rhode Island	22627.9560	7285.6293
Missouri	22205.1500	6436.2105
Oklahoma	19683.3900	4853.9560
Alabama	19510.6400	5786.8253
Oregon	17431.1500	-1190.4705
Nevada	16729.1020	3316.7659
Connecticut	13384.3570	3511.4918
Arkansas	11678.1300	4008.6871
Utah	11220.0560	2546.5335
Mississippi	10771.3400	3172.9762
Louisiana	9217.0300	2196.1023
Vermont	8929.3700	2244.9783
South Carolina	8481.7100	1769.0566
Nebraska	7464.9300	2037.0942
New Hampshire	7292.5240	1706.5028
Montana	5589.3520	1833.3285
New Mexico	4783.5220	1157.1161
Iowa	4579.7600	1183.8119
Idaho	4382.4860	826.7231
Kansas	2914.3100	836.4435
District of Columbia	2865.0200	1059.5893
Wyoming	1603.1360	100.1960
South Dakota	1315.5600	394.8283
Maine	1270.5300	454.4862

State	Sales	Profit
West Virginia	1209.8240	185.9216
North Dakota	919.9100	230.1497

In [100…
```
# sales by city

retail.groupby("City")[["Sales", "Profit"]].sum().sort_values(by = "Sales", ascending = False)
```

Out[100…

City	Sales	Profit
New York City	256368.161	62036.9837
Los Angeles	175851.341	30440.7579
Seattle	119540.742	29156.0967
San Francisco	112669.092	17507.3854
Philadelphia	109077.013	-13837.7674
...	...	...
Ormond Beach	2.808	-1.9656
Pensacola	2.214	-1.4760
Jupiter	2.064	0.1548
Elyria	1.824	-1.3984
Abilene	1.392	-3.7584

531 rows × 2 columns

In [101…
```
# Profits by region

retail.groupby("Region")[["Sales", "Profit"]].sum().sort_values(by = "Sales", ascending = False)
```

Out[101…

Region	Sales	Profit
West	725457.8245	108418.4489
East	678781.2400	91522.7800
Central	501239.8908	39706.3625
South	391721.9050	46749.4303

In [102…
```
# Top Selling Products

top_15_selling = retail.groupby("Product Name")["Quantity"].sum().nlargest(15)
top_15_selling
```

Out[102…
```
Product Name
Staples                                                 215
Staple envelope                                         170
Easy-staple paper                                       150
Staples in misc. colors                                  86
KI Adjustable-Height Table                               74
Avery Non-Stick Binders                                  71
Storex Dura Pro Binders                                  71
GBC Premium Transparent Covers with Diagonal Lined Pattern  67
Situations Contoured Folding Chairs, 4/Set               64
Staple-based wall hangings                               62
Chromcraft Round Conference Tables                       61
Eldon Wave Desk Accessories                              61
Staple remover                                           61
Global Wood Trimmed Manager's Task Chair, Khaki          59
Wilson Jones Turn Tabs Binder Tool for Ring Binders      59
Name: Quantity, dtype: int64
```

Return on Investment

In [103…
```
# ROI

retail["roi"] = retail["Profit"] / retail["Sales"]
roi_by_category = retail.groupby("Category")["roi"].mean()
```

```
# ROI in percentage

roi_by_category
```

Out[103…
```
Category
Furniture          0.038784
Office Supplies    0.138030
Technology         0.156138
Name: roi, dtype: float64
```

In [104…
```
# ROI by region

regional_roi = retail.groupby("Region")["roi"].mean()

print("___________ Regional ROI ___________")
print(regional_roi)
```

```
___________ Regional ROI ___________
Region
Central   -0.104073
East       0.167227
South      0.163519
West       0.219487
Name: roi, dtype: float64
```

Sales Person

In [105…
```
retail["Retail Sales People"].unique()
```

Out[105…
```
array(['Cassandra Brandow', 'Anna Andreadi', 'Kelly Williams',
       'Chuck Magee'], dtype=object)
```

In [106…
```
retail["Retail Sales People"].nunique()
```

Out[106… 4

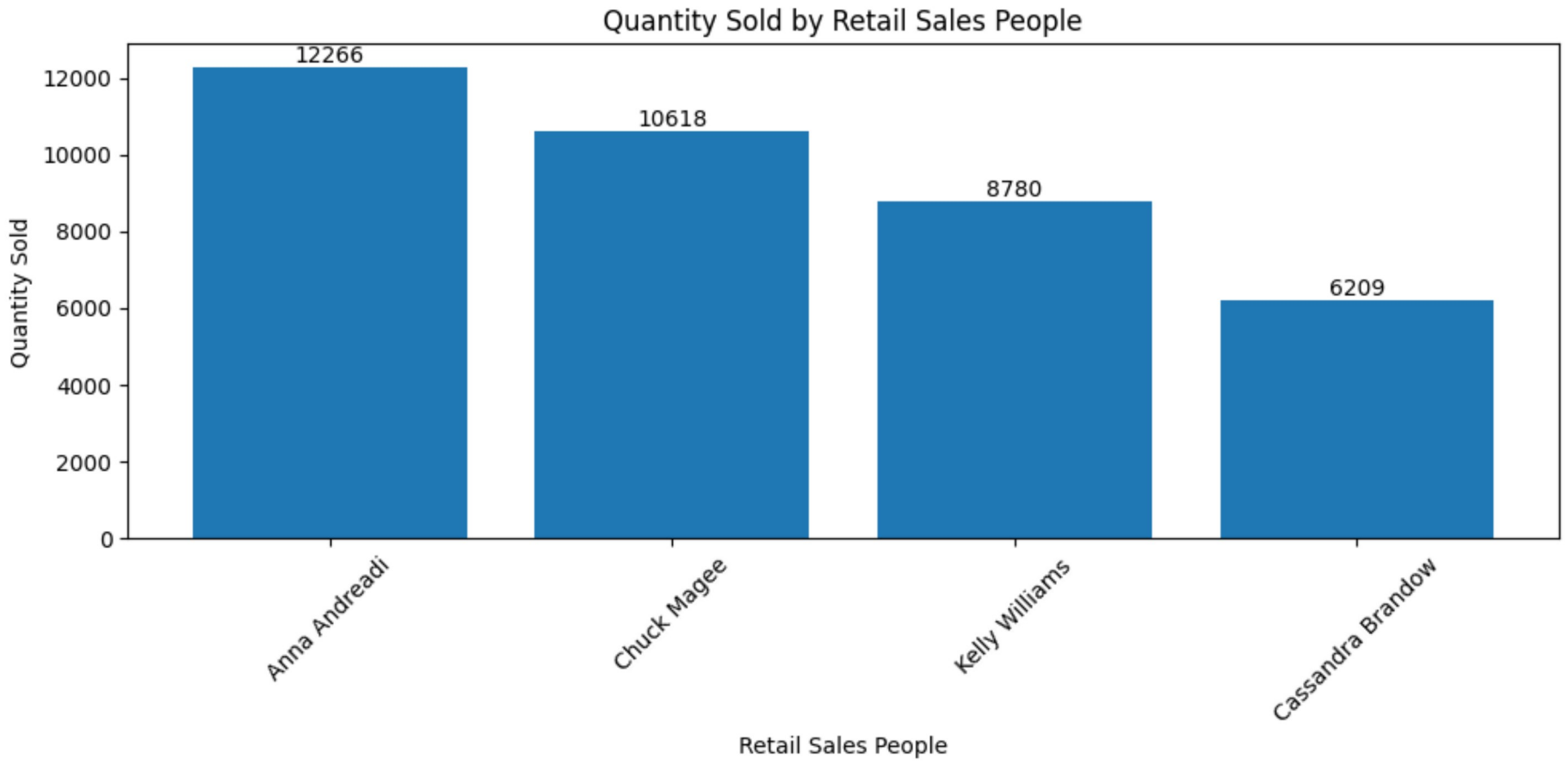
In [107…
```
retail["Retail Sales People"].value_counts().sort_values(ascending = False)
```

Out[107…
```
Retail Sales People
Anna Andreadi        3203
Chuck Magee          2848
Kelly Williams       2323
Cassandra Brandow    1620
Name: count, dtype: int64
```

In [108…
```
retail.groupby("Retail Sales People")["Quantity"].sum().sort_values(ascending = False)
```

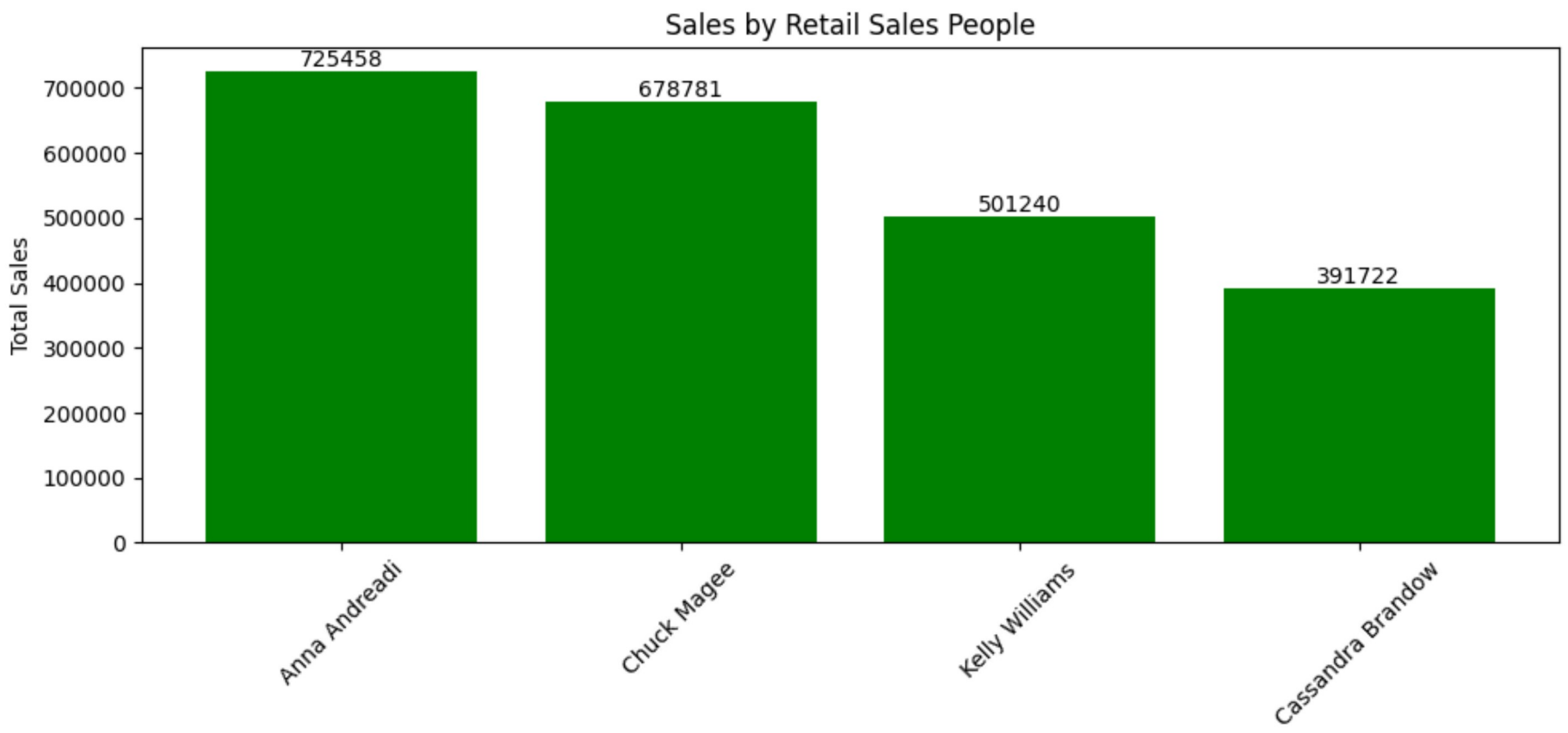
Out[108…
```
Retail Sales People
Anna Andreadi        12266
Chuck Magee          10618
Kelly Williams        8780
Cassandra Brandow     6209
Name: Quantity, dtype: int64
```

In [109…
```
salesByPersons = (retail.groupby("Retail Sales People")["Quantity"].sum().sort_values(ascending=False).reset_index())
salesByPersons.columns = ["Retail Sales People", "Quantity"]

fig, ax = plt.subplots(figsize = (10,5))
bars = ax.bar(salesByPersons["Retail Sales People"], salesByPersons["Quantity"])
ax.bar_label(bars)
plt.xticks(rotation=45)
plt.title("Quantity Sold by Retail Sales People")
plt.xlabel("Retail Sales People")
plt.ylabel("Quantity Sold")
plt.tight_layout()
plt.show()
```



In [110…

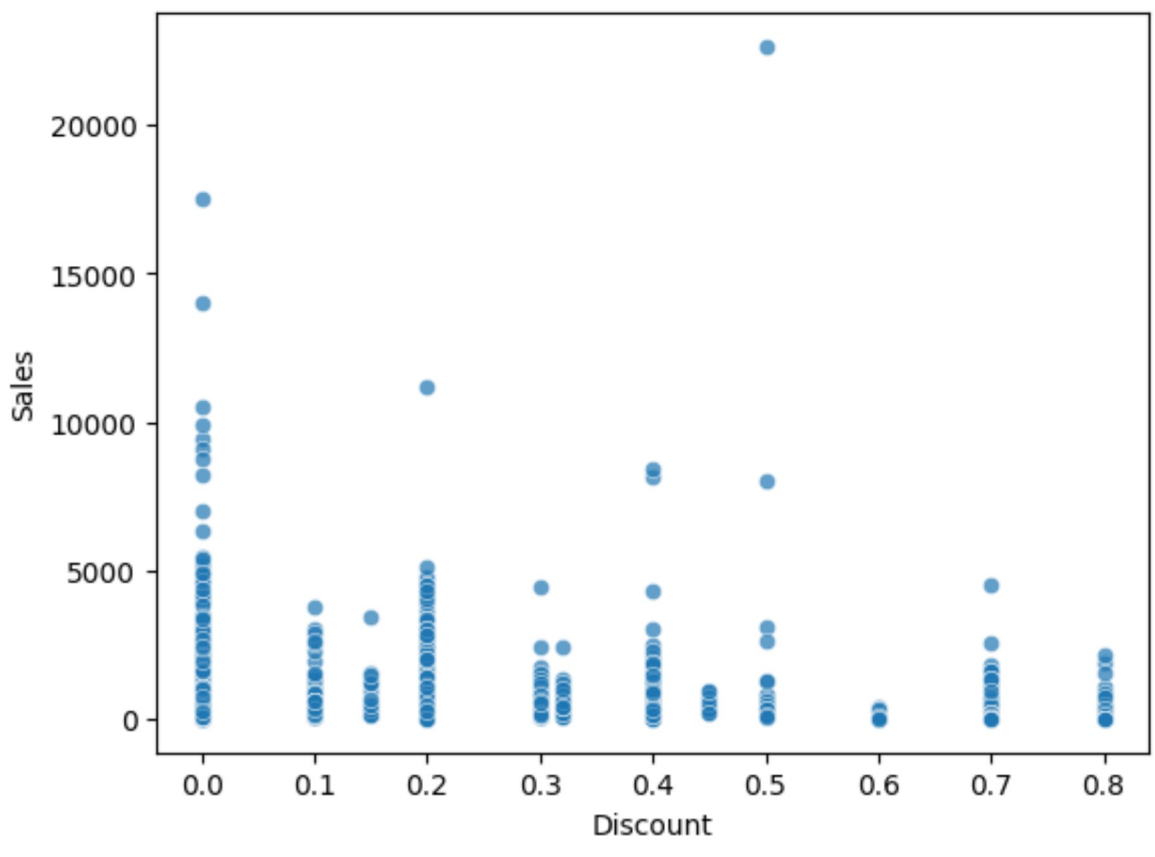
```
salesByPersons = (retail.groupby("Retail Sales People")["Sales"].sum().sort_values(ascending=False).reset_index())
salesByPersons.columns = ["Retail Sales People", "Sales"]

fig, ax = plt.subplots(figsize = (10,5))
bars = ax.bar(salesByPersons["Retail Sales People"], salesByPersons["Sales"], color = "green")
ax.bar_label(bars)
plt.xticks(rotation=45)
plt.title("Sales by Retail Sales People")
plt.xlabel("Retail Sales People")
plt.ylabel("Total Sales")
plt.tight_layout()
plt.show()
```



In [111…

```
salesByPersons = (retail.groupby("Retail Sales People")["Profit"].sum().sort_values(ascending=False).reset_index())
salesByPersons.columns = ["Retail Sales People", "Profit"]

fig, ax = plt.subplots(figsize = (10,5))
bars = ax.bar(salesByPersons["Retail Sales People"], salesByPersons["Profit"])
ax.bar_label(bars)
plt.xticks(rotation=45)
plt.title("Profit Made by Retail Sales People")
plt.xlabel("Retail Sales People")
plt.ylabel("Profit")
plt.tight_layout()
plt.show()
```

In [112…
```
sns.scatterplot(data = retail, y = "Sales", x = "Discount", alpha = 0.7)
plt.show()
```



In [113…
```
retail.groupby("Retail Sales People")[["Sales", "Profit"]].sum().sort_values(by = "Sales", ascending = False)
```

Out[113…

Retail Sales People	Sales	Profit
Anna Andreadi	725457.8245	108418.4489
Chuck Magee	678781.2400	91522.7800
Kelly Williams	501239.8908	39706.3625
Cassandra Brandow	391721.9050	46749.4303

In [114…
```
print(retail.columns)
```

```
Index(['Row ID', 'Order ID', 'Order Date', 'MonthName', 'Ship Date',
       'Ship Mode', 'Customer ID', 'Customer Name', 'Segment', 'Country',
       'City', 'State', 'Postal Code', 'Region', 'Retail Sales People',
       'Product ID', 'Category', 'Sub-Category', 'Product Name', 'Returned',
       'Sales', 'Quantity', 'Discount', 'Profit', 'Order_DOW',
       'Order_MonthName', 'Order_YearName', 'Ship Time', 'roi'],
      dtype='object')
```

In [115…
```
retail.head()
```

Out[115...

	Row ID	Order ID	Order Date	MonthName	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State
0	1	CA-2016-152156	2016-08-11	Aug	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky
1	2	CA-2016-152156	2016-08-11	Aug	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky
2	3	CA-2016-138688	2016-12-06	Dec	2016-12-06	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California
3	4	US-2015-108966	2015-11-10	Nov	2015-11-10	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida
4	5	US-2015-108966	2015-11-10	Nov	2015-11-10	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida

In [116...
```
retail.groupby("State")["Sales"].sum()
```

Out[116...
```
State
Alabama                 19510.6400
Arizona                 35282.0010
Arkansas                11678.1300
California             457687.6315
Colorado                32108.1180
Connecticut             13384.3570
Delaware                27451.0690
District of Columbia     2865.0200
Florida                 89473.7080
Georgia                 49095.8400
Idaho                    4382.4860
Illinois                80166.1010
Indiana                 53555.3600
Iowa                     4579.7600
Kansas                   2914.3100
Kentucky                36591.7500
Louisiana                9217.0300
Maine                    1270.5300
Maryland                23705.5230
Massachusetts           28634.4340
Michigan                76269.6140
Minnesota               29863.1500
Mississippi             10771.3400
Missouri                22205.1500
Montana                  5589.3520
Nebraska                 7464.9300
Nevada                  16729.1020
New Hampshire            7292.5240
New Jersey              35764.3120
New Mexico               4783.5220
New York               310876.2710
North Carolina          55603.1640
North Dakota              919.9100
Ohio                    78258.1360
Oklahoma                19683.3900
Oregon                  17431.1500
Pennsylvania           116511.9140
Rhode Island            22627.9560
South Carolina           8481.7100
South Dakota             1315.5600
Tennessee               30661.8730
Texas                  170188.0458
Utah                    11220.0560
Vermont                  8929.3700
Virginia                70636.7200
Washington             138641.2700
West Virginia            1209.8240
Wisconsin               32114.6100
Wyoming                  1603.1360
Name: Sales, dtype: float64
```