Sales Analysis

November 19, 2024

SALES ANALYSIS

0.1 PROBLEM STATEMENT :

To perform a detailed analysis of a dataset containing sales information and order quantities, we would typically follow a series of steps in the data analysis process. Below is a structured approach to performing the analysis, which includes data cleaning, exploratory data analysis (EDA), and generating insights based on the data.

0.1.1 Objectives:

Importing necessary Libraries/Modules: - Import the modules necessary for Data Manipulation and Visualization.

Loading dataset: - Read the dataset containing sales information.

Task 1 - Exploring the Dataset: - Understand the Structure and various datatypes of the attributes within the dataset.

Task 2 - Missing value analysis: - Identify and analyze missing values in the dataset.‘

Task 3 - Adding data with additional columns
- Adding New Column Based on Existing Data

Task 4 - Bi-varidate analysis
- Conduct bivariate analysis to explore relationships between different variables and answer the following questions

CONCLUSION

0.1.2 IMPORTING LIBRARIES/MODULES

[1]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
from matplotlib.ticker import MultipleLocator, FuncFormatter
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

0.1.3 LOADING THE DATASET

[2]:
```
# Get the list of all CSV files in the directory
files=[file for file in os.listdir("./DirtySalesDate")if file.endswith(".csv")]
```

[3]:
```
files
```

[3]:
```
['Sales_April_2019.csv',
 'Sales_August_2019.csv',
 'Sales_December_2019.csv',
 'Sales_February_2019.csv',
 'Sales_January_2019.csv',
 'Sales_July_2019.csv',
 'Sales_June_2019.csv',
 'Sales_March_2019.csv',
 'Sales_May_2019.csv',
 'Sales_November_2019.csv',
 'Sales_October_2019.csv',
 'Sales_September_2019.csv']
```

We have 12 CSV files, one for each month of the year 2019. Let's combine them into a single CSV file for easier analysis.

[4]:
```
# Create an empty Dataframe to hold the combined data
df=pd.DataFrame()

# Read and concatenate each CSV file
for i in files:
    month_data=pd.read_csv("./DirtySalesDate"+"//"+i)
    df=pd.concat([df,month_data])
```

0.2 Task 1 - Exploring the Dataset:

[5]:
```
df.head()
```

[5]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
1	NaN	NaN	NaN	NaN	NaN	NaN
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001

[6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 186850 entries, 0 to 11685
Data columns (total 6 columns):
 #   Column            Non-Null Count   Dtype 
---  ------            --------------   ----- 
 0   Order ID          186305 non-null  object
 1   Product           186305 non-null  object
 2   Quantity Ordered  186305 non-null  object
 3   Price Each        186305 non-null  object
 4   Order Date        186305 non-null  object
 5   Purchase Address  186305 non-null  object
dtypes: object(6)
memory usage: 10.0+ MB
```

[7]:
```
df.describe()
```

[7]:

	Order ID	Product	Quantity Ordered	Price Each	\
count	186305	186305	186305	186305	
unique	178438	20	10	24	
top	Order ID	USB-C Charging Cable	1	11.95	
freq	355	21903	168552	21903	

	Order Date	Purchase Address
count	186305	186305
unique	142396	140788
top	Order Date	Purchase Address
freq	355	355

0.3 Task 2 - Missing value analysis:

[8]:
```
#  let's check the the count of NaN values for each column.
df.isnull().sum()
```

[8]:
```
Order ID            545
Product             545
Quantity Ordered    545
Price Each          545
Order Date          545
Purchase Address    545
dtype: int64
```

[9]:
```
# filter rows in df where any of the values are NaN
df[df.isnull().any(axis=1)]
```

[9]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
1	NaN	NaN	NaN	NaN	NaN	NaN

```
356        NaN      NaN               NaN         NaN         NaN               NaN
735        NaN      NaN               NaN         NaN         NaN               NaN
1433       NaN      NaN               NaN         NaN         NaN               NaN
1553       NaN      NaN               NaN         NaN         NaN               NaN
…          …        …                 …           …           …                 …
10012      NaN      NaN               NaN         NaN         NaN               NaN
10274      NaN      NaN               NaN         NaN         NaN               NaN
10878      NaN      NaN               NaN         NaN         NaN               NaN
11384      NaN      NaN               NaN         NaN         NaN               NaN
11662      NaN      NaN               NaN         NaN         NaN               NaN

[545 rows x 6 columns]
```

[10]:
```
# dropping the rows where every column is NaN
df = df.dropna(how='all')
```

[11]:
```
df.head()
```

[11]:
```
  Order ID                    Product Quantity Ordered Price Each  \
0   176558        USB-C Charging Cable                2      11.95
2   176559  Bose SoundSport Headphones                1      99.99
3   176560                Google Phone                1        600
4   176560            Wired Headphones                1      11.99
5   176561            Wired Headphones                1      11.99

       Order Date                      Purchase Address
0  04/19/19 08:46          917 1st St, Dallas, TX 75001
2  04/07/19 22:30     682 Chestnut St, Boston, MA 02215
3  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001
4  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001
5  04/30/19 09:27     333 8th St, Los Angeles, CA 90001
```

0.3.1 Let's correct the data types of each column

[12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 186305 entries, 0 to 11685
Data columns (total 6 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Order ID          186305 non-null  object
 1   Product           186305 non-null  object
 2   Quantity Ordered  186305 non-null  object
 3   Price Each        186305 non-null  object
 4   Order Date        186305 non-null  object
 5   Purchase Address  186305 non-null  object
dtypes: object(6)
```

```
memory usage: 9.9+ MB
```

[13]:
```
 # Filter out rows where 'Quantity Ordered' column contains 'Quantity Ordered'
df=df[df["Quantity Ordered"]!="Quantity Ordered"]
```

[14]:
```
# changing the data type of "Quantity Ordered" to int
df["Quantity Ordered"]=df["Quantity Ordered"].astype(int)
```

[15]:
```
# converting "Order Date" column into a datetime format.
df["Order Date"]=pd.to_datetime(df["Order Date"])
```

[16]:
```
# changing the data type of "Price Each" to numeric
df['Price Each'] = df['Price Each'].astype(float)
```

[17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 185950 entries, 0 to 11685
Data columns (total 6 columns):
 #   Column            Non-Null Count   Dtype         
---  ------            --------------   -----         
 0   Order ID          185950 non-null  object        
 1   Product           185950 non-null  object        
 2   Quantity Ordered  185950 non-null  int32         
 3   Price Each        185950 non-null  float64       
 4   Order Date        185950 non-null  datetime64[ns]
 5   Purchase Address  185950 non-null  object        
dtypes: datetime64[ns](1), float64(1), int32(1), object(3)
memory usage: 9.2+ MB
```

0.4 Task 3 - Adding data with additional columns

Adding a “Month” column to identify the best month for sales

[18]:
```
df["Month"]=df["Order Date"].dt.month
```

Adding a “Sales” column

[19]:
```
df["Sales"]=df["Quantity Ordered"]*df["Price Each"]
```

Adding a “City” column

[20]:
```
def city(address):
    return address.split(",")[1]
def state(address):
    return address.split(",")[2].split(" ")[1]

df["City"]=df["Purchase Address"].apply(lambda x : f"{city(x)} ({state(x)})")
```

[21]:
```
df.head()
```

[21]:
```
  Order ID                   Product  Quantity Ordered  Price Each  \
0   176558       USB-C Charging Cable                 2       11.95
2   176559  Bose SoundSport Headphones                 1       99.99
3   176560                Google Phone                 1      600.00
4   176560           Wired Headphones                 1       11.99
5   176561           Wired Headphones                 1       11.99

           Order Date                      Purchase Address  Month   Sales  \
0 2019-04-19 08:46:00          917 1st St, Dallas, TX 75001      4   23.90
2 2019-04-07 22:30:00     682 Chestnut St, Boston, MA 02215      4   99.99
3 2019-04-12 14:38:00  669 Spruce St, Los Angeles, CA 90001      4  600.00
4 2019-04-12 14:38:00  669 Spruce St, Los Angeles, CA 90001      4   11.99
5 2019-04-30 09:27:00     333 8th St, Los Angeles, CA 90001      4   11.99

               City
0       Dallas (TX)
2       Boston (MA)
3   Los Angeles (CA)
4   Los Angeles (CA)
5   Los Angeles (CA)
```

Sorting the data according to the order date

[22]:
```
# sorting the data according to the order date
df.sort_values(by="Order Date",inplace=True)
```

[23]:
```
# reseting the index
df.reset_index(drop=True, inplace=True)
```

[24]:
```
df.head()
```

[24]:
```
  Order ID                  Product  Quantity Ordered  Price Each  \
0   147268          Wired Headphones                 1       11.99
1   148041      USB-C Charging Cable                 1       11.95
2   149343  Apple Airpods Headphones                 1      150.00
3   149964    AAA Batteries (4-pack)                 1        2.99
4   149350      USB-C Charging Cable                 2       11.95

           Order Date                        Purchase Address  Month   Sales  \
0 2019-01-01 03:07:00      9 Lake St, New York City, NY 10001      1   11.99
1 2019-01-01 03:40:00  760 Church St, San Francisco, CA 94016      1   11.95
2 2019-01-01 04:56:00     735 5th St, New York City, NY 10001      1  150.00
3 2019-01-01 05:53:00         75 Jackson St, Dallas, TX 75001      1    2.99
4 2019-01-01 06:03:00           943 2nd St, Atlanta, GA 30301      1   23.90

                 City
0   New York City (NY)
1   San Francisco (CA)
```

```
2   New York City (NY)
3          Dallas (TX)
4         Atlanta (GA)
```

1 Task 4 - Bi-varidate analysis

1.0.1 Question 1: what was the best month for sales ? How much was earned that month

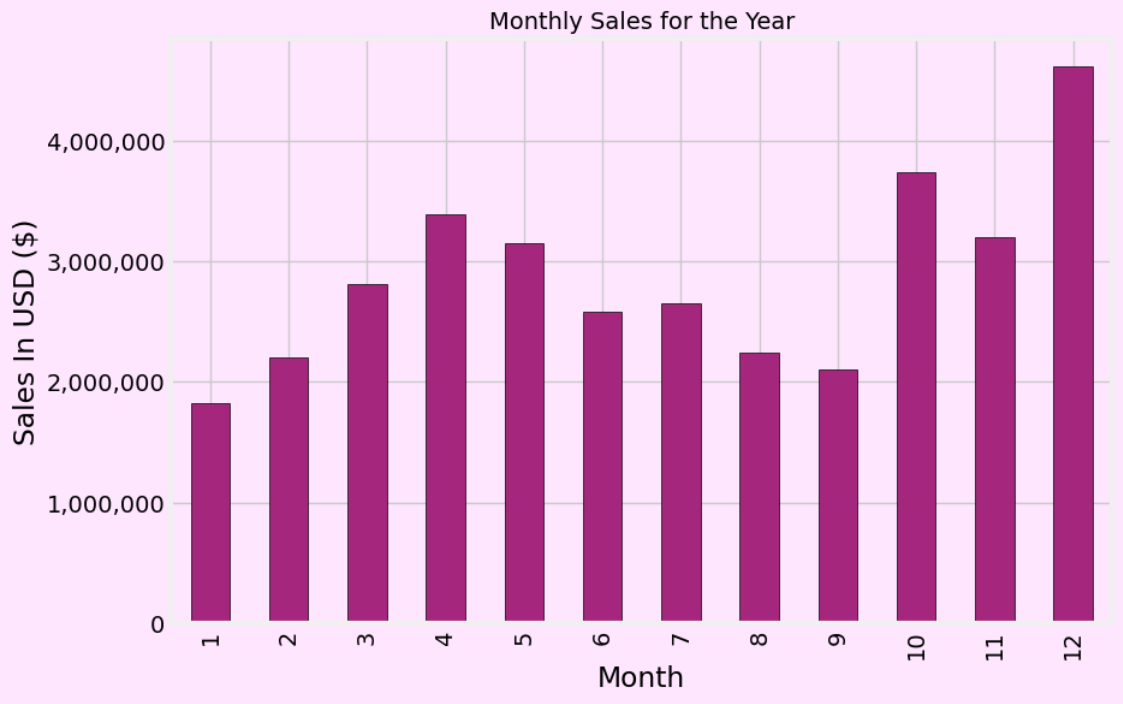
[25]:
```
month_sale=df.groupby(["Month"])["Sales"].sum()
```

[26]:
```
month_sale
```

[26]:
```
Month
1     1822256.73
2     2202022.42
3     2807100.38
4     3390670.24
5     3152606.75
6     2577802.26
7     2647775.76
8     2244467.88
9     2097560.13
10    3736726.88
11    3199603.20
12    4613443.34
Name: Sales, dtype: float64
```

[27]:
```
# set the style to fivethirtyeight
plt.style.use("fivethirtyeight")

# Create a figure and axes
fig, ax = plt.subplots(figsize=(9,6))

# Set the background color
fig.patch.set_facecolor("#ffe6ff")  # Figure background
ax.set_facecolor('#ffe6ff')  #plot area background

month_sale.plot(kind="bar",color="#a5277d",edgecolor="black")

# x label
plt.xlabel("Month")

# ylabel
plt.ylabel("Sales In USD ($)")

# title of the graph
```

```
plt.title("Monthly Sales for the Year", fontsize=14)

# Format y-axis labels with commas
plt.gca().yaxis.set_major_formatter(ticker.FuncFormatter(lambda x, pos: f'{x:,.
↪0f}'))

# show the graph
plt.show()
```



The best month for sales is December (Month 12), with a total of $4,613,443.34 in sales.** December (Month 12) has the highest sales at 4,613,443.34 dollars, nearly doubling the sales of January, which had the lowest sales at 1,822,256.73 dollars. This suggests that sales tend to increase toward the end of the year, likely due to seasonal factors such as the holiday shopping season.

[28]:
```
# sales in different citites in all the 12 month
```

[29]:
```
df.pivot_table(
    values="Sales",             # Column to aggregate
    index="City",               # Rows (index)
    columns="Month",            # Columns
    aggfunc="sum",              # Aggregation function (e.g., sum, mean, etc.)
    fill_value=0                # Optional: Fill missing values with 0 (or any␣
↪value you prefer)
)
```

[29]:
Month	1	2	3	4	5
City					
Atlanta (GA)	149159.54	176470.30	231905.38	284448.91	238853.99
Austin (TX)	88087.06	108787.40	154549.27	172683.59	160635.22
Boston (MA)	201088.49	214808.36	301023.81	353807.11	328803.65
Dallas (TX)	143462.51	186667.99	222376.68	251360.48	268456.49
Los Angeles (CA)	288601.90	342061.66	429929.37	550264.02	499689.21
New York City (NY)	260591.29	305372.26	367262.20	449447.75	436126.40
Portland (ME)	22708.80	29845.49	30516.29	42536.49	57978.76
Portland (OR)	92276.76	119606.37	156691.72	197441.63	173729.25
San Francisco (CA)	435588.33	547072.34	693726.96	812426.19	776679.49
Seattle (WA)	140692.05	171330.25	219118.70	276254.07	211654.29

Month	6	7	8	9	10
City					
Atlanta (GA)	219816.47	211766.47	169267.66	171278.89	306293.01
Austin (TX)	144057.29	150324.93	125713.61	106483.70	203196.12
Boston (MA)	254747.89	291497.14	239275.26	248408.73	367036.39
Dallas (TX)	186885.75	212325.17	179763.46	164212.86	323135.60
Los Angeles (CA)	451531.93	394334.64	345893.50	354075.69	612453.25
New York City (NY)	324148.58	355716.10	303183.46	300563.83	486954.41
Portland (ME)	30025.33	32421.14	35996.60	28759.56	52322.52
Portland (OR)	139562.02	143994.49	116881.14	103811.88	201778.34
San Francisco (CA)	613173.48	642881.76	538778.10	463595.72	866700.98
Seattle (WA)	213853.52	212513.92	189715.09	156369.27	316856.26

Month	11	12
City		
Atlanta (GA)	275338.70	360899.26
Austin (TX)	171286.47	233777.09
Boston (MA)	351546.02	509599.16
Dallas (TX)	248609.56	380718.85
Los Angeles (CA)	499690.79	684044.84
New York City (NY)	428180.32	646770.83
Portland (ME)	34681.22	51966.07
Portland (OR)	173210.70	251748.04
San Francisco (CA)	764979.29	1106601.27
Seattle (WA)	252080.13	387317.93

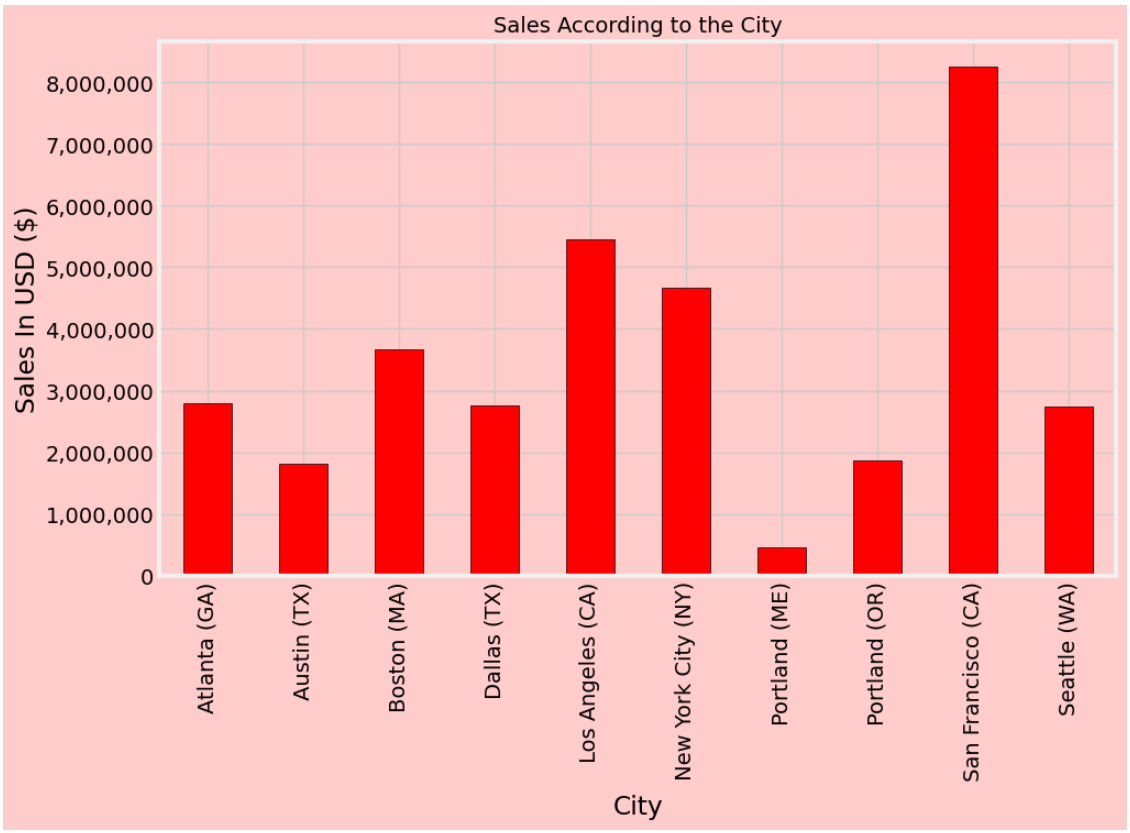
1.0.2 Question 2: what city has the highest number of sales?

[30]:
```
result=df.groupby(["City"])["Sales"].sum()
result
```

[30]:
```
City
Atlanta (GA)          2795498.58
Austin (TX)           1819581.75
```

```
Boston (MA)           3661642.01
Dallas (TX)           2767975.40
Los Angeles (CA)      5452570.80
New York City (NY)    4664317.43
Portland (ME)          449758.27
Portland (OR)         1870732.34
San Francisco (CA)    8262203.91
Seattle (WA)          2747755.48
Name: Sales, dtype: float64
```

[31]:
```python
# Apply fivethirtyeight style
plt.style.use("fivethirtyeight")

# Create a figure and axes
fig, ax = plt.subplots(figsize=(10,6))

# Set the background colors
fig.patch.set_facecolor('#ffcccc')  # Set figure background color
ax.set_facecolor('#ffcccc')             # Set plot area background color

# Plot the data as a bar chart
result.plot(kind="bar", ax=ax, color='#ff0000', edgecolor='black')

# Customize labels and title
plt.ylabel("Sales In USD ($)")
plt.xlabel("City")
plt.title("Sales According to the City", fontsize=14)

# Format y-axis labels with commas
ax.yaxis.set_major_formatter(ticker.FuncFormatter(lambda x, pos: f'{x:,.0f}'))

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



San Francisco (CA) leads the cities in total sales revenue, with over 8.26 million dollars, while Portland (ME) shows the lowest sales at just under 450K, highlighting significant regional difference in sales performance.

[32]:
```
df.pivot_table(
    values="Sales",              # Column to aggregate
    index="Product",                # Rows (index)
    columns="City",             # Columns
    aggfunc="sum",               # Aggregation function (e.g., sum, mean, etc.)
)
```

[32]:
```
City                        Atlanta (GA)  Austin (TX)  Boston (MA)  \
Product
20in Monitor                    37616.58     25297.70     43336.06
27in 4K Gaming Monitor         192265.07    124016.82    263243.25
27in FHD Monitor                88194.12     53996.40    119542.03
34in Ultrawide Monitor         183155.18    124636.72    254973.29
AA Batteries (4-pack)            8421.12      5468.16     11581.44
AAA Batteries (4-pack)           7053.41      4987.32     10348.39
Apple Airpods Headphones       189900.00    133050.00    247950.00
Bose SoundSport Headphones     108389.16     70692.93    141585.84
```

Flatscreen TV	122100.00	72600.00	166200.00
Google Phone	270600.00	164400.00	355800.00
LG Dryer	35400.00	33000.00	35400.00
LG Washing Machine	31200.00	15600.00	43200.00
Lightning Charging Cable	28091.05	19539.65	37240.45
Macbook Pro Laptop	644300.00	426700.00	814300.00
ThinkPad Laptop	356996.43	209997.90	446995.53
USB-C Charging Cable	22884.25	14949.45	30603.95
Vareebadd Phone	69200.00	43200.00	85600.00
Wired Headphones	18932.21	13548.70	26641.78
iPhone	380800.00	263900.00	527100.00

City	Dallas (TX)	Los Angeles (CA)	New York City (NY)	\
Product				
20in Monitor	37726.57	72373.42	61594.40	
27in 4K Gaming Monitor	187585.19	391159.97	328371.58	
27in FHD Monitor	88044.13	183437.77	160789.28	
34in Ultrawide Monitor	194554.88	362890.45	329831.32	
AA Batteries (4-pack)	8682.24	17041.92	13939.20	
AAA Batteries (4-pack)	7486.96	14851.33	12330.76	
Apple Airpods Headphones	179100.00	370950.00	314700.00	
Bose SoundSport Headphones	106289.37	212478.75	179382.06	
Flatscreen TV	126000.00	218100.00	188400.00	
Google Phone	276600.00	508800.00	454800.00	
LG Dryer	26400.00	69600.00	46200.00	
LG Washing Machine	30600.00	63000.00	51000.00	
Lightning Charging Cable	27866.80	56391.40	45462.95	
Macbook Pro Laptop	649400.00	1276700.00	1116900.00	
ThinkPad Laptop	344996.55	640993.59	559994.40	
USB-C Charging Cable	22131.40	45194.90	39064.55	
Vareebadd Phone	71200.00	126800.00	112400.00	
Wired Headphones	20011.31	39207.30	32456.93	
iPhone	363300.00	782600.00	616700.00	

City	Portland (ME)	Portland (OR)	San Francisco (CA)	\
Product				
20in Monitor	6489.41	24087.81	109990.00	
27in 4K Gaming Monitor	33149.15	136106.51	569385.40	
27in FHD Monitor	17098.86	62395.84	272081.86	
34in Ultrawide Monitor	28879.24	124256.73	549465.54	
AA Batteries (4-pack)	1493.76	5952.00	25171.20	
AAA Batteries (4-pack)	1070.42	5148.78	22149.92	
Apple Airpods Headphones	34950.00	129900.00	559950.00	
Bose SoundSport Headphones	17998.20	70892.91	331666.83	
Flatscreen TV	18600.00	75000.00	346200.00	
Google Phone	46200.00	166800.00	814800.00	
LG Dryer	3600.00	18600.00	85800.00	

```
LG Washing Machine                6600.00       15600.00           108000.00
Lightning Charging Cable          4021.55       18866.90            83077.15
Macbook Pro Laptop              107100.00      465800.00          1931200.00
ThinkPad Laptop                  52999.47      220997.79           962990.37
USB-C Charging Cable              4063.00       14841.90            70433.30
Vareebadd Phone                   6800.00       42800.00           197600.00
Wired Headphones                  3345.21       12985.17            59542.34
iPhone                           55300.00      259700.00          1162700.00

City                         Seattle (WA)
Product
20in Monitor                      35636.76
27in 4K Gaming Monitor           209814.62
27in FHD Monitor                  86844.21
34in Ultrawide Monitor           202914.66
AA Batteries (4-pack)              8367.36
AAA Batteries (4-pack)             7313.54
Apple Airpods Headphones         188700.00
Bose SoundSport Headphones       106189.38
Flatscreen TV                    112500.00
Google Phone                     260400.00
LG Dryer                          33600.00
LG Washing Machine                34800.00
Lightning Charging Cable          26536.25
Macbook Pro Laptop               605200.00
ThinkPad Laptop                  332996.67
USB-C Charging Cable              22334.55
Vareebadd Phone                   71600.00
Wired Headphones                  19807.48
iPhone                           382200.00
```

1.0.3 Question 3 :What products are most sold together?

[33]:
```
df.head()
```

[33]:
```
   Order ID                   Product  Quantity Ordered  Price Each  \
0    147268          Wired Headphones                 1       11.99
1    148041       USB-C Charging Cable                1       11.95
2    149343  Apple Airpods Headphones                 1      150.00
3    149964     AAA Batteries (4-pack)                1        2.99
4    149350       USB-C Charging Cable                2       11.95

            Order Date                       Purchase Address  Month   Sales  \
0  2019-01-01 03:07:00     9 Lake St, New York City, NY 10001      1   11.99
1  2019-01-01 03:40:00  760 Church St, San Francisco, CA 94016      1   11.95
2  2019-01-01 04:56:00    735 5th St, New York City, NY 10001      1  150.00
3  2019-01-01 05:53:00        75 Jackson St, Dallas, TX 75001      1    2.99
```

```
4 2019-01-01 06:03:00           943 2nd St, Atlanta, GA 30301      1    23.90

                  City
0   New York City (NY)
1   San Francisco (CA)
2   New York City (NY)
3          Dallas (TX)
4         Atlanta (GA)
```

[34]:
```
sold_together=df[df["Order ID"].duplicated(keep=False)]
```

[35]:
```
sold_together.head(10)
```

[35]:
```
    Order ID                   Product  Quantity Ordered  Price Each  \
11   144804                    iPhone                 1      700.00
12   144804          Wired Headphones                 1       11.99
65   148074      USB-C Charging Cable                 1       11.95
66   148074              Google Phone                 1      600.00
79   148450                    iPhone                 1      700.00
80   148450  Lightning Charging Cable                 1       14.95
92   144679      USB-C Charging Cable                 1       11.95
93   144679              Google Phone                 1      600.00
94   147451          Wired Headphones                 1       11.99
95   147451              Google Phone                 1      600.00

            Order Date                         Purchase Address  Month  \
11 2019-01-01 07:29:00     628 Lake St, New York City, NY 10001      1
12 2019-01-01 07:29:00     628 Lake St, New York City, NY 10001      1
65 2019-01-01 11:25:00          6 Johnson St, Atlanta, GA 30301      1
66 2019-01-01 11:25:00          6 Johnson St, Atlanta, GA 30301      1
79 2019-01-01 12:02:00        761 Lakeview St, Dallas, TX 75001      1
80 2019-01-01 12:02:00        761 Lakeview St, Dallas, TX 75001      1
92 2019-01-01 12:51:00  984 Lakeview St, San Francisco, CA 94016      1
93 2019-01-01 12:51:00  984 Lakeview St, San Francisco, CA 94016      1
94 2019-01-01 12:57:00      229 Elm St, New York City, NY 10001      1
95 2019-01-01 12:57:00      229 Elm St, New York City, NY 10001      1

     Sales                 City
11  700.00   New York City (NY)
12   11.99   New York City (NY)
65   11.95         Atlanta (GA)
66  600.00         Atlanta (GA)
79  700.00          Dallas (TX)
80   14.95          Dallas (TX)
92   11.95   San Francisco (CA)
93  600.00   San Francisco (CA)
94   11.99   New York City (NY)
```

```
95  600.00   New York City (NY)
```

[36]:
```
sold_together["grouped"]=sold_together.groupby("Order ID")["Product"].
 ↪transform(lambda x : ",".join(x))
sold_together=sold_together[["Order ID","grouped"]].drop_duplicates()
```

[37]:
```
sold_together.reset_index(drop=True,inplace=True)
```

[38]:
```
sold_together
```

[38]:

	Order ID	grouped
0	144804	iPhone,Wired Headphones
1	148074	USB-C Charging Cable,Google Phone
2	148450	iPhone,Lightning Charging Cable
3	144679	USB-C Charging Cable,Google Phone
4	147451	Wired Headphones,Google Phone
…	…	…
7131	301832	20in Monitor,AAA Batteries (4-pack)
7132	311036	Macbook Pro Laptop,AAA Batteries (4-pack)
7133	311386	Apple Airpods Headphones,iPhone
7134	297817	iPhone,Lightning Charging Cable
7135	300519	Bose SoundSport Headphones,Lightning Charging …

[7136 rows x 2 columns]

[39]:
```
sold_together["grouped"].value_counts()
```

[39]:
```
grouped
Lightning Charging Cable,iPhone                                  458
USB-C Charging Cable,Google Phone                                453
iPhone,Lightning Charging Cable                                  433
Google Phone,USB-C Charging Cable                                415
Wired Headphones,iPhone                                          191
                                                                …
iPhone,Lightning Charging Cable,34in Ultrawide Monitor             1
iPhone,Bose SoundSport Headphones,Apple Airpods Headphones         1
LG Washing Machine,iPhone                                          1
Vareebadd Phone,27in 4K Gaming Monitor                             1
Google Phone,34in Ultrawide Monitor                                1
Name: count, Length: 418, dtype: int64
```

Using the Counter function from the collections module to create a dictionary where the key represents pairs of products sold together, and the value indicates the number of times those products were sold together.

[40]:
```
from collections import Counter

dict=Counter(sold_together["grouped"])
```

1.0.4 Question 4: What product sold the most and why do you think it sold the most ?

[42]:
```
df.head()
```

[42]:
```
  Order ID                   Product  Quantity Ordered  Price Each  \
0   147268          Wired Headphones                 1       11.99
1   148041      USB-C Charging Cable                 1       11.95
2   149343  Apple Airpods Headphones                 1      150.00
3   149964    AAA Batteries (4-pack)                 1        2.99
4   149350      USB-C Charging Cable                 2       11.95

            Order Date                        Purchase Address  Month   Sales  \
0 2019-01-01 03:07:00      9 Lake St, New York City, NY 10001      1   11.99
1 2019-01-01 03:40:00  760 Church St, San Francisco, CA 94016      1   11.95
2 2019-01-01 04:56:00     735 5th St, New York City, NY 10001      1  150.00
3 2019-01-01 05:53:00         75 Jackson St, Dallas, TX 75001      1    2.99
4 2019-01-01 06:03:00           943 2nd St, Atlanta, GA 30301      1   23.90

                  City
0   New York City (NY)
1   San Francisco (CA)
2   New York City (NY)
3          Dallas (TX)
4         Atlanta (GA)
```

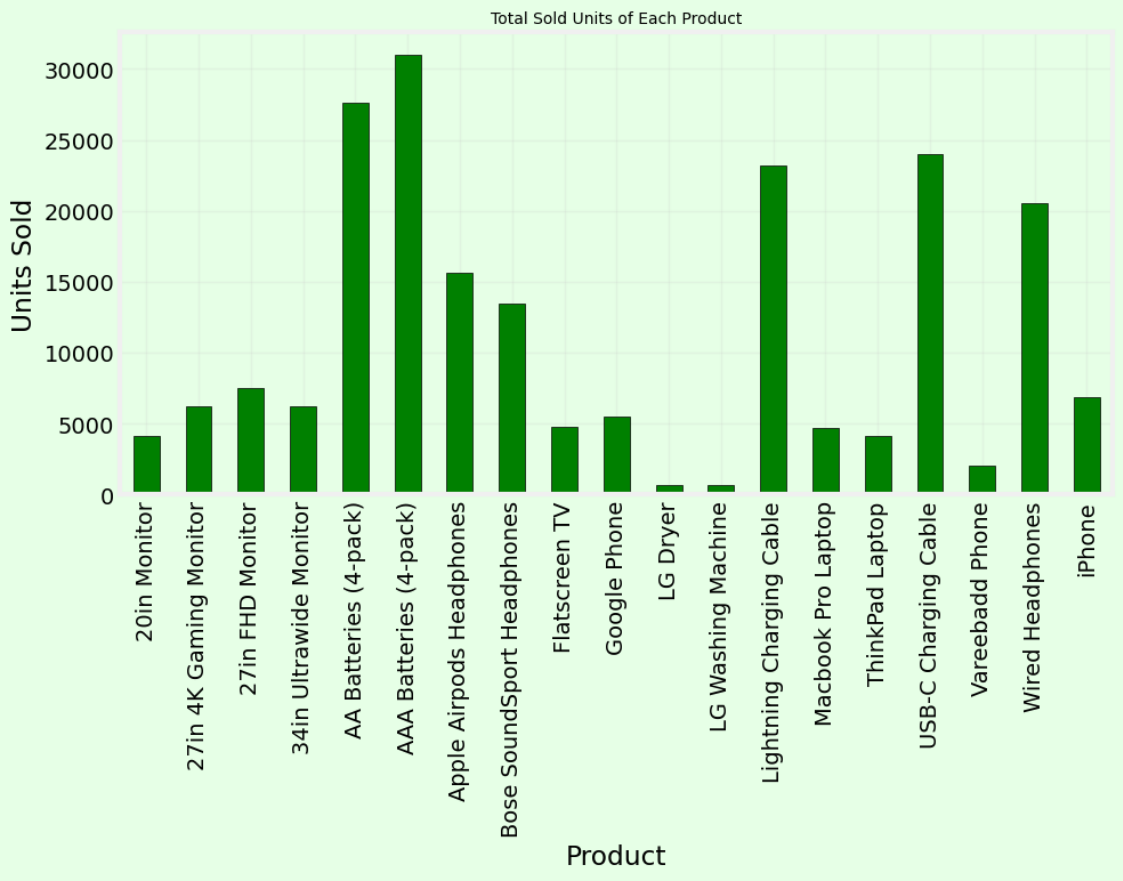
[43]:
```
quantity_sold=df.groupby(["Product"])["Quantity Ordered"].sum()
```

[44]:
```
quantity_sold.sort_values(ascending=False)
```

[44]:
```
Product
AAA Batteries (4-pack)        31017
AA Batteries (4-pack)         27635
USB-C Charging Cable          23975
Lightning Charging Cable      23217
Wired Headphones              20557
Apple Airpods Headphones      15661
Bose SoundSport Headphones    13457
27in FHD Monitor               7550
iPhone                         6849
27in 4K Gaming Monitor         6244
34in Ultrawide Monitor         6199
Google Phone                   5532
Flatscreen TV                  4819
Macbook Pro Laptop             4728
ThinkPad Laptop                4130
20in Monitor                   4129
Vareebadd Phone                2068
```

```
LG Washing Machine              666
LG Dryer                        646
Name: Quantity Ordered, dtype: int32
```

[45]:
```python
# Apply fivethirtyeight style
plt.style.use("fivethirtyeight")

# Create a figure and axes
fig, ax = plt.subplots(figsize=(10, 5))

# Set the background colors
fig.patch.set_facecolor('#e6ffe6')  # Set figure background color
ax.set_facecolor('#e6ffe6')            # Set plot area (axes) background color

# Plot the data as a bar chart
quantity_sold.plot(kind="bar", ax=ax, color="#008000", edgecolor='black')

# Add title and labels
plt.title("Total Sold Units of Each Product", fontsize=10)
plt.ylabel("Units Sold")
plt.xlabel("Product")

# Add a grid with lower alpha for transparency
plt.grid(alpha=0.2)

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



The best-selling product is the AAA Batteries (4-pack), with a total of 31,017 units sold, followed by the AA Batteries (4-pack) with a total of 27,635 units sold. The high sales of AAA batteries can likely be attributed to their widespread use in small electronic devices such as remote controls, flashlights, toys, clocks, and more.**

1.0.5 Question 5:What time should we display the advertisements to maximize likelihood of customer's buying?

[46]:
```
df['Order Date']=pd.to_datetime(df['Order Date'])
df.head()
```

[46]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales
0	147268	Wired Headphones	1	11.99	2019-01-01 03:07:00	9 Lake St, New York City, NY 10001	1	11.99
1	148041	USB-C Charging Cable	1	11.95				
2	149343	Apple Airpods Headphones	1	150.00				
3	149964	AAA Batteries (4-pack)	1	2.99				
4	149350	USB-C Charging Cable	2	11.95				

```
1 2019-01-01 03:40:00  760 Church St, San Francisco, CA 94016      1   11.95
2 2019-01-01 04:56:00     735 5th St, New York City, NY 10001      1  150.00
3 2019-01-01 05:53:00         75 Jackson St, Dallas, TX 75001      1    2.99
4 2019-01-01 06:03:00           943 2nd St, Atlanta, GA 30301      1   23.90

                  City
0   New York City (NY)
1   San Francisco (CA)
2   New York City (NY)
3          Dallas (TX)
4         Atlanta (GA)
```

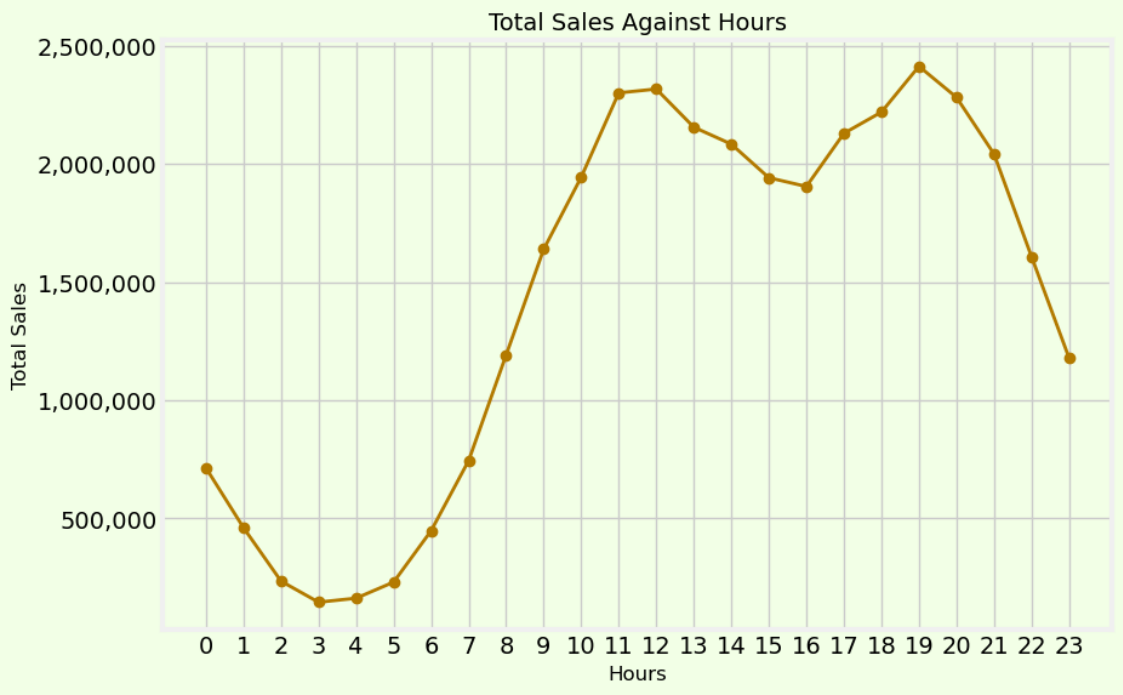
[47]:
```
df['Hours']=df['Order Date'].dt.hour
df.head()
```

[47]:
```
  Order ID                   Product  Quantity Ordered  Price Each  \
0   147268          Wired Headphones                 1       11.99
1   148041      USB-C Charging Cable                 1       11.95
2   149343  Apple Airpods Headphones                 1      150.00
3   149964    AAA Batteries (4-pack)                 1        2.99
4   149350      USB-C Charging Cable                 2       11.95

           Order Date                        Purchase Address  Month   Sales  \
0 2019-01-01 03:07:00      9 Lake St, New York City, NY 10001      1   11.99
1 2019-01-01 03:40:00  760 Church St, San Francisco, CA 94016      1   11.95
2 2019-01-01 04:56:00     735 5th St, New York City, NY 10001      1  150.00
3 2019-01-01 05:53:00         75 Jackson St, Dallas, TX 75001      1    2.99
4 2019-01-01 06:03:00           943 2nd St, Atlanta, GA 30301      1   23.90

                  City  Hours
0   New York City (NY)      3
1   San Francisco (CA)      3
2   New York City (NY)      4
3          Dallas (TX)      5
4         Atlanta (GA)      6
```

[48]:
```
by_hours=df.groupby('Hours')['Sales'].sum()
gp_by_hours = by_hours.reset_index()
gp_by_hours.columns = ['Hours','Total sales']
# gp_by_hours
```

[49]:
```
# Set the style to fivethirtyeight
plt.style.use("fivethirtyeight")

# Create a figure and axes
fig, ax = plt.subplots(figsize=(9, 6))
```

```
# Set the background color
fig.patch.set_facecolor("#f2ffe6")  # Figure background
ax.set_facecolor('#f2ffe6')  # Plot area background

# Plotting the data
ax.plot(gp_by_hours['Hours'], gp_by_hours['Total sales'], color="#b37a00",␣
 ↪marker="o", linestyle="-", linewidth=2, markersize=6)

# Set x and y labels
plt.xlabel("Hours", fontsize=12)
plt.ylabel("Total Sales", fontsize=12)

# Set title
plt.title("Total Sales Against Hours", fontsize=14)

# Customize the x-axis ticks
ax.xaxis.set_major_locator(MultipleLocator(1))  # Set ticks every 1 hour
ax.set_xticks(gp_by_hours['Hours'])  # Align ticks with data points

# Format y-axis labels with commas
ax.yaxis.set_major_formatter(FuncFormatter(lambda x, pos: f'{x:,.0f}'))

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



The graph shows:

Low sales during early morning (2-5 AM) and late night (after 9 PM). Peak sales in the mid-morning (10-11 AM) and evening (5-8 PM). Midday stability with a slight dip post-lunch (2-3 PM). Optimize resources during peak hours and reduce costs during low-activity periods.

1.0.6 Let us store this cleaned data into a new excel file naming "Sales_analysis.xlsx"

[50]:
```
df.to_excel("Sales_analysis_Dashboard.xlsx",index=False)
```

Read the file above file

[ ]:
```
df=pd.read_excel("Sales_analysis_Dashboard.xlsx")
df.head()
```

1.0.7 Let us compare the Price and sales of each product

[ ]:
```
# Group the data by 'Product' and 'Price Each' to calculate the total quantity␣
 ↪ordered for each combination.
# This provides insight into how the quantity sold varies with product and␣
 ↪price.
price_sale=df.groupby(["Product","Price Each"])["Quantity Ordered"].sum()
```

[ ]:
```
price_sale_df=pd.DataFrame(price_sale)
```

[ ]:
```
price_sale_df.reset_index(inplace=True)
```

[ ]:
```
fig, ax1 = plt.subplots(figsize=(10, 6))
plt.style.use("default")
# Bar plot for Sales
ax1.bar(price_sale_df['Product'], price_sale_df['Quantity Ordered'],␣
 ↪color='skyblue', label='Sales')
ax2 = ax1.twinx()
# line plot for Price
ax2.plot(price_sale_df['Product'], price_sale_df['Price Each'], color='orange',␣
 ↪marker='o', label='Price')


ax1.set_xlabel('Product',fontsize=15)
ax1.set_ylabel('Sales', color='skyblue',fontsize=15)
ax1.tick_params(axis="y",color="skyblue")
ax2.set_ylabel('Price', color='orange',fontsize=15)
# ax1.set_xticklabels(rotation="vertical")
ax1.set_xticklabels(price_sale_df['Product'], rotation=90)

# Title and show plot
plt.title('Sales and Price of Products',fontsize=15)
plt.show()
```

Products like USB-C Charging Cable and AAA Batteries are low in price but show decent sales, making them affordable and accessible.MacBook Pro Laptop has a high price but moderate sales.Bose SoundSport Headphones and Apple AirPods Headphones have a balanced combination of moderate pricing and high sales, suggesting good value and popularity. FlatScreen TV has moderate sales but a noticeable price.

2 CONCLUSION

This analysis highlights key trends in sales performance, regional differences, and product popularity. - December was the best month for sales, nearly doubling January's figures, suggesting a strong seasonal boost driven by holiday shopping. - Regionally, San Francisco led with over 8.26 million dollars in sales, while Portland (ME) showed the lowest sales at under 450K, indicating significant regional variations. - In terms of products, the AAA Batteries (4-pack) were the best-sellers, with 31,017 units sold, reflecting strong demand for affordable, everyday items. - Other popular products like the AA Batteries and USB-C Charging Cables also performed well, while higher-ticket items such as the MacBook Pro and Apple AirPods showed solid sales at higher price points. - Low sales during early morning (2-5 AM) and late night (after 9 PM). Peak sales in the mid-morning (10-11 AM) and evening (5-8 PM). Midday stability with a slight dip post-lunch (2-3 PM). Optimize resources during peak hours and reduce costs during low-activity periods.

Overall, this data underscores the importance of seasonal trends, regional markets, and consumer preferences in shaping sales strategies. Products with low price points and high utility, like batteries, drive volume sales, while premium items cater to customers seeking value in quality and performance.

[ ]: