Sales Analysis Report

## Sales Analysis

import Necessary Libraries

python  
import pandas as pd  
import os

Task #1: Merge the 12 months of sales data into a single CSV file

```python

# df= pd.readcsv('./SalesData/SalesApril2019.csv')

```python

files= [file for file in os.listdir('./Sales\_Data')]

all\_months\_data=pd.DataFrame()

for file in files:

df = pd.read\_csv('./Sales\_Data/'+file)

all\_months\_data = pd.concat([all\_months\_data,df])

all\_months\_data.head()

all\_months\_data.to\_csv('all\_data.csv',index=False)

all\_data=pd.read\_csv('all\_data.csv')

```

python  
all\_data.head()

python  
all\_data.isna().sum()

Order ID 545  
Product 545  
Quantity Ordered 545  
Price Each 545  
Order Date 545  
Purchase Address 545

**Clean Up the data**

**Drop rows of NAN**

```python  
nan\_df = all\_data[all\_data.isna().any(axis=1)]

display(nan\_df.head())

all\_data = all\_data.dropna(how='all')

all\_data.head()  
```

Find 'OR' and delete it

```python  
all\_data = all\_data[all\_data['Order Date'].str[0:2] != 'Or']

```

python  
all\_data['Month'] = all\_data['Month'].astype('int32')  
all\_data['Month'] = all\_data['Month'].astype('int32')  
all\_data.head()

Convert Colums to the correcct type

python  
all\_data['Quantity Ordered'] = pd.to\_numeric(all\_data['Quantity Ordered'])  
all\_data['Price Each'] = pd.to\_numeric(all\_data['Price Each'])  
all\_data.head()

# Augment data with additional columns

### Task2 Add Column Column

```python  
all\_data['Month'] = all\_data['Order Date'].str[0:2]

```

python  
all\_data.head()

**Task3: Add a sales column**

python  
all\_data['Sales'] = all\_data['Quantity Ordered'] \* all\_data['Price Each']  
all\_data.head()

**Add city Columns**

```python

# All\_data[['Street','City','State']]=alldata['Purchase Address'].str.split(',', expand=True)

# use .apply()

def get\_city(address):  
 return address.split(',')[1]

def get\_state(address):  
 return address.split(',')[2].split(' ')[1]

alldata['City']=alldata['Purchase Address'].apply(lambda x: f"{getcity(x)} ({getstate(x)})")

all\_data  
```

**Question 1: What was the best Month for Sales? How much was earned that month**

python  
results = all\_data.groupby('Month').sum()

python  
import matplotlib.pyplot as plt

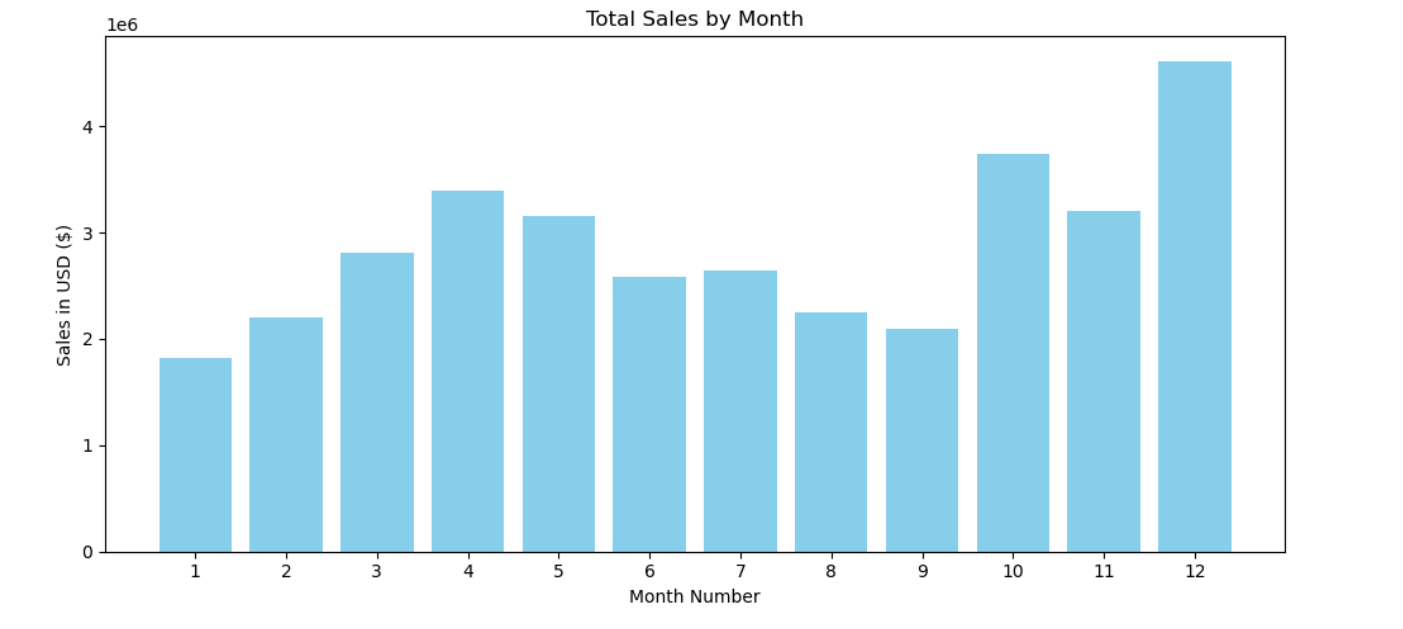
months = range(1, 13)

plt.figure(figsize=(10, 5)) # Optional: Add size for better readability  
plt.bar(months, results['Sales'], color='skyblue')

plt.xticks(months) # Set month numbers on x-axis  
plt.ylabel('Sales in USD ($)')  
plt.xlabel('Month Number')  
plt.title('Total Sales by Month')  
plt.tight\_layout()

plt.show()

```



**Question 2: What city had the highest number of sales**

python  
all\_data['Purchase Address']

0 917 1st St, Dallas, TX 75001  
2 682 Chestnut St, Boston, MA 02215  
3 669 Spruce St, Los Angeles, CA 90001  
4 669 Spruce St, Los Angeles, CA 90001  
5 333 8th St, Los Angeles, CA 90001  
 ...   
186845 840 Highland St, Los Angeles, CA 90001  
186846 216 Dogwood St, San Francisco, CA 94016  
186847 220 12th St, San Francisco, CA 94016  
186848 511 Forest St, San Francisco, CA 94016  
186849 250 Meadow St, San Francisco, CA 94016  
Name: Purchase Address, Length: 185950, dtype: object

python  
results = all\_data.groupby('City')[['Sales','Quantity Ordered','Price Each']].sum().reset\_index()

python  
print(results)

City Sales Quantity Ordered Price Each  
0 Atlanta (GA) 2795498.58 16602 2779908.20  
1 Austin (TX) 1819581.75 11153 1809873.61  
2 Boston (MA) 3661642.01 22528 3637409.77  
3 Dallas (TX) 2767975.40 16730 2752627.82  
4 Los Angeles (CA) 5452570.80 33289 5421435.23  
5 New York City (NY) 4664317.43 27932 4635370.83  
6 Portland (ME) 449758.27 2750 447189.25  
7 Portland (OR) 1870732.34 11303 1860558.22  
8 San Francisco (CA) 8262203.91 50239 8211461.74  
9 Seattle (WA) 2747755.48 16553 2733296.01

```python  
import matplotlib.pyplot as plt

# Step 1: Calculate total sales by city

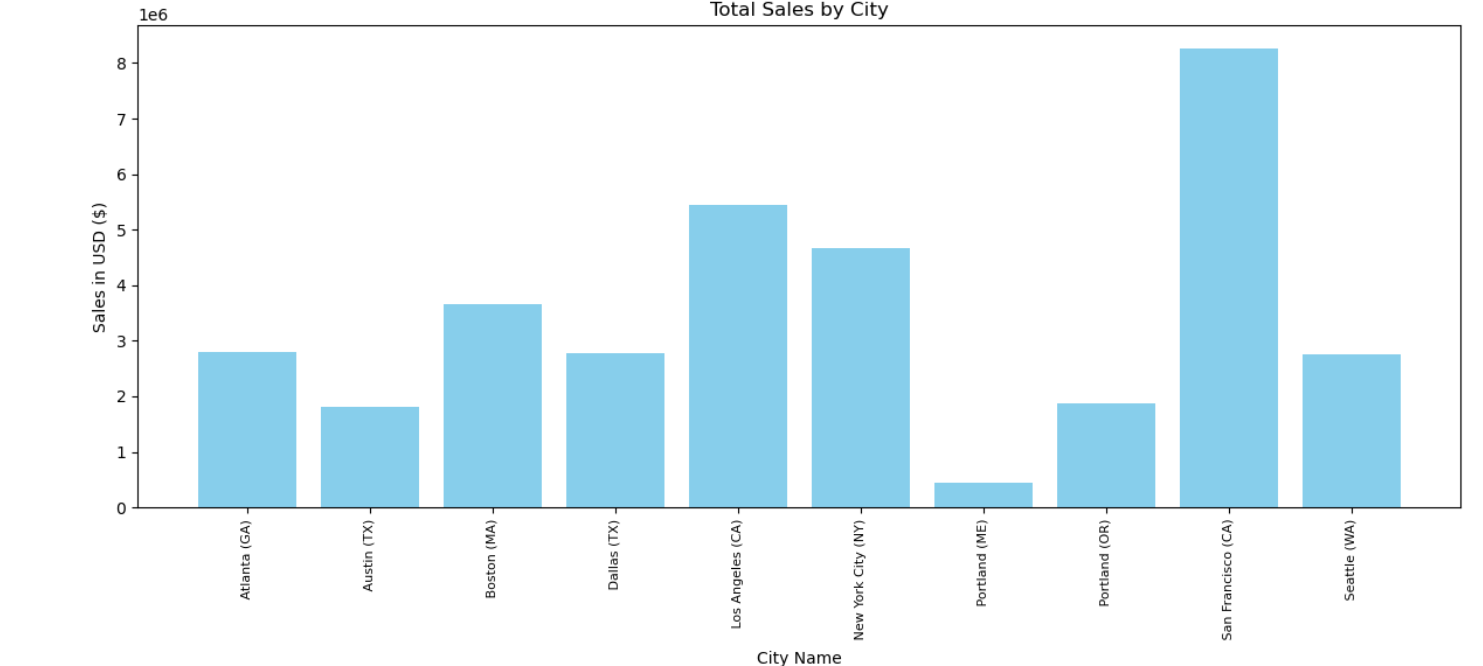
citysales = alldata.groupby('City')['Sales'].sum()

# Step 2: Extract cities (this keeps correct order)

cities = citysales.index  
sales = citysales.values

# Step 3: Plotting

plt.figure(figsize=(12, 6))  
plt.bar(cities, sales, color='skyblue')  
plt.xticks(rotation=90, fontsize=8)  
plt.ylabel('Sales in USD ($)')  
plt.xlabel("City Name")  
plt.title("Total Sales by City")  
plt.tight\_layout()  
plt.show()



```

**Question 3: What time should we display advertisements to maximize likeihood of customer's buying product**

python  
all\_data['Order Date'] = pd.to\_datetime(all\_data['Order Date'])

python  
all\_data['Date'] = all\_data['Order Date'].dt.date  
all\_data['Time'] = all\_data['Order Date'].dt.time  
all\_data

python  
all\_data['Hour']=all\_data['Order Date'].dt.hour

python  
all\_data['Minute'] = all\_data['Order Date'].dt.minute

```python  
import matplotlib.pyplot as plt

# Group by hour and count number of orders

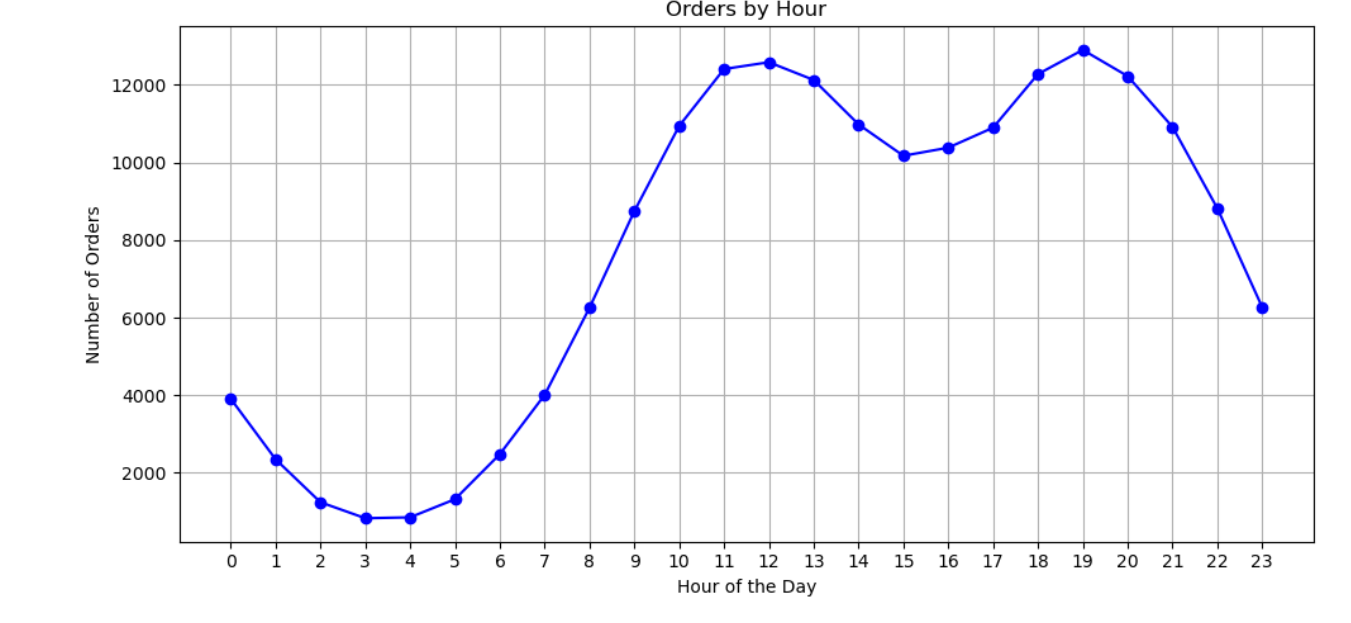
ordersperhour = all\_data.groupby('Hour').count()['Order ID']

# Extract the hours (0–23)

hours = ordersperhour.index

# Plot

plt.figure(figsize=(10, 5))  
plt.plot(hours, ordersperhour, marker='o', color='blue')  
plt.xticks(hours) # Show all hours from 0 to 23  
plt.xlabel('Hour of the Day')  
plt.ylabel('Number of Orders')  
plt.title('Orders by Hour')  
plt.grid(True)  
plt.tight\_layout()  
plt.show()



**Question 4: What productss are most often sold together**

```python

# Step 1: Filter duplicates (same Order ID → same customer ordered multiple items)

df = alldata[alldata['Order ID'].duplicated(keep=False)].copy()

# Step 2: Group by Order ID and join product names into a single string

df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))

# Step 3: Drop duplicate rows to get one row per order

df = df[['Order ID', 'Grouped']].drop\_duplicates()

df.head()

```

```python  
from itertools import combinations  
from collections import Counter

count = Counter()

for row in df['Grouped']:  
 rowlist = row.split(',')  
 count.update(Counter(combinations(rowlist,2)))

for key,value in count.most\_common(10):  
 print(key,value)  
```

('iPhone', 'Lightning Charging Cable') 1005  
('Google Phone', 'USB-C Charging Cable') 987  
('iPhone', 'Wired Headphones') 447  
('Google Phone', 'Wired Headphones') 414  
('Vareebadd Phone', 'USB-C Charging Cable') 361  
('iPhone', 'Apple Airpods Headphones') 360  
('Google Phone', 'Bose SoundSport Headphones') 220  
('USB-C Charging Cable', 'Wired Headphones') 160  
('Vareebadd Phone', 'Wired Headphones') 143  
('Lightning Charging Cable', 'Wired Headphones') 92

**Question 5: What product sold the most? Why do you think it sold the most?**

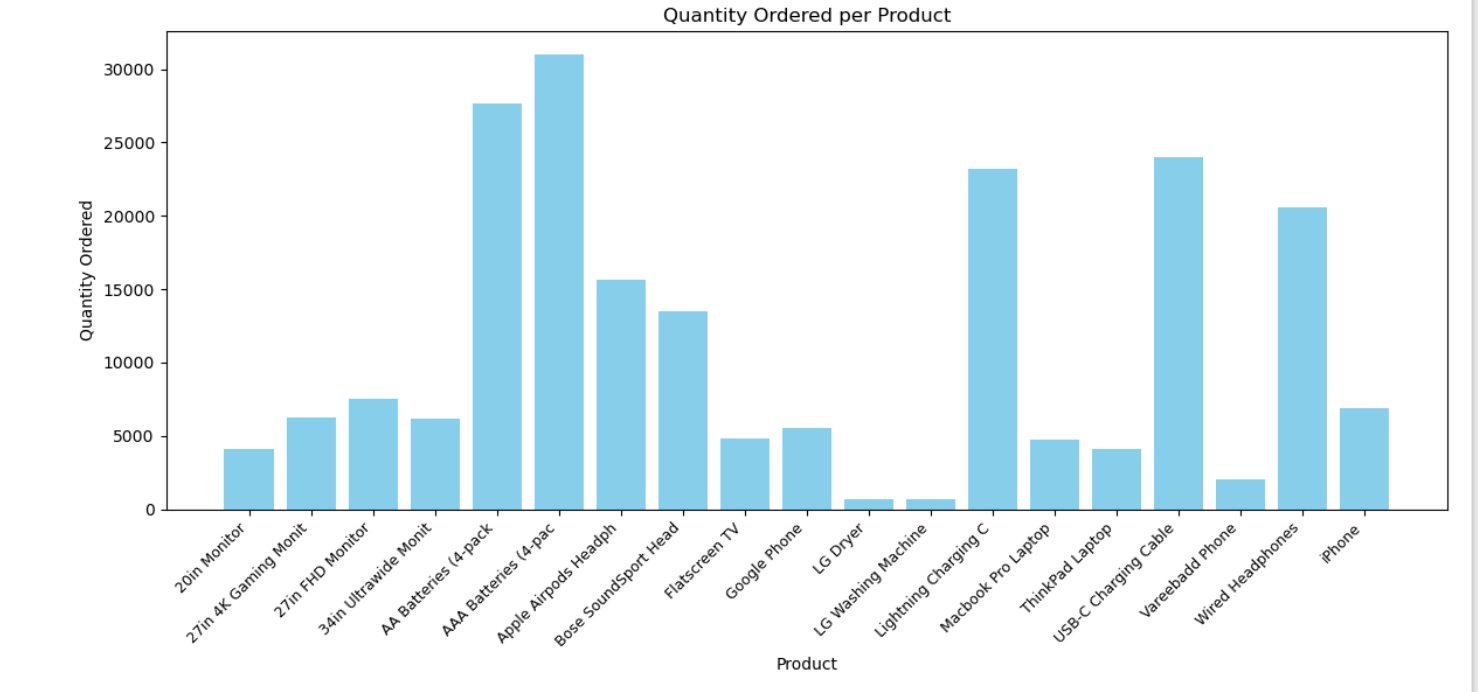
```python  
import matplotlib.pyplot as plt

productgroup = alldata.groupby('Product')  
quantityordered = productgroup['Quantity Ordered'].sum()

products = [p[:20] for p in quantity\_ordered.index]

plt.figure(figsize=(12, 6))

plt.bar(products, quantityordered.values, color='skyblue')  
plt.xticks(rotation=45, ha='right', fontsize=9)  
plt.xlabel('Product')  
plt.ylabel('Quantity Ordered')  
plt.title('Quantity Ordered per Product')  
plt.tightlayout()  
plt.show()



```

```python  
import matplotlib.pyplot as plt

# Step 1: Calculate quantity ordered and prices per product

productgroup = alldata.groupby('Product')  
quantityordered = productgroup['Quantity Ordered'].sum()  
prices = productgroup['Price Each'].mean()  
products = list(quantityordered.index) # Use index to preserve order

# Step 2: Plotting

fig, ax1 = plt.subplots(figsize=(12, 6)) # Set figure size

ax2 = ax1.twinx() # Create second y-axis

# Bar for quantity ordered

ax1.bar(products, quantityordered, color='g')  
ax1.setylabel('Quantity Ordered', color='g')

# Line for price

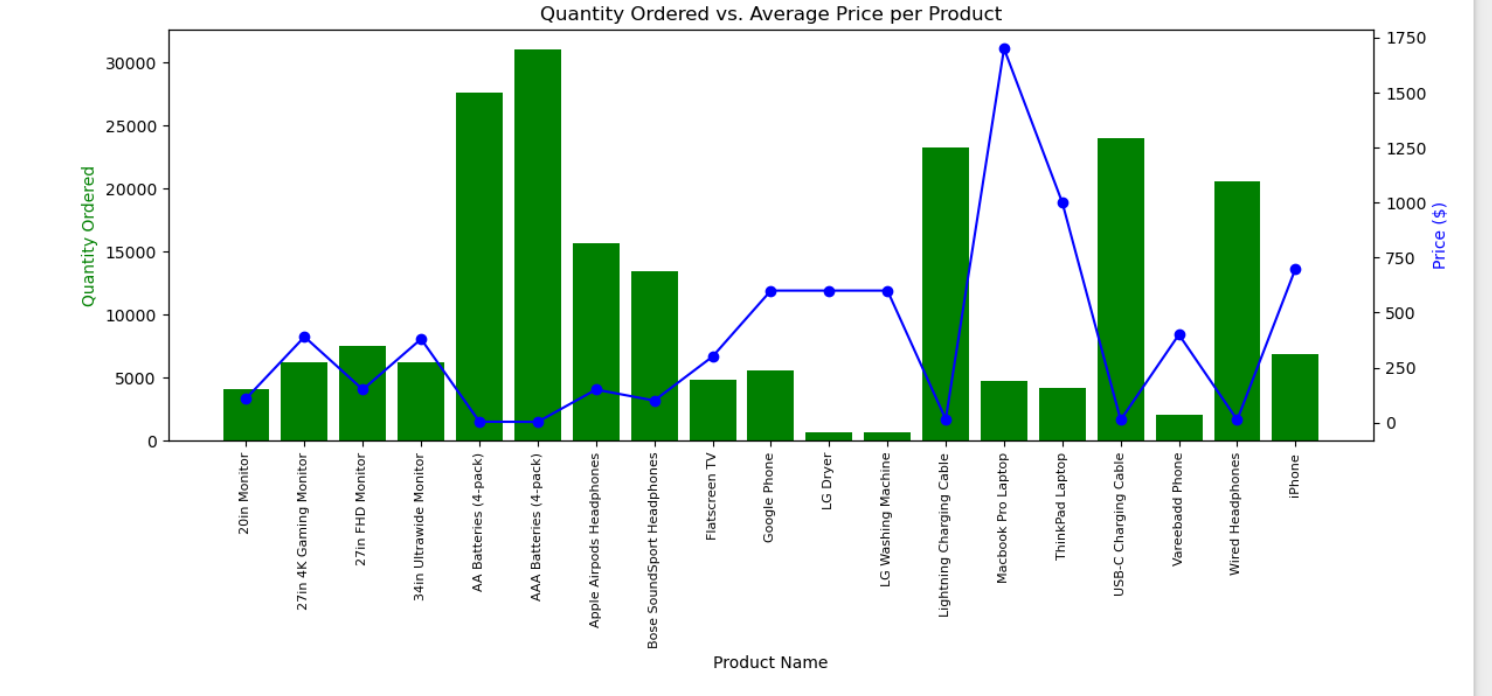
ax2.plot(products, prices, 'b-o') # add markers for visibility  
ax2.set\_ylabel('Price ($)', color='b')

# X-axis formatting

ax1.setxlabel('Product Name')  
ax1.setxticks(range(len(products)))  
ax1.set\_xticklabels(products, rotation='vertical', fontsize=8)

plt.title("Quantity Ordered vs. Average Price per Product")  
plt.tight\_layout()  
plt.show()

```



```python

```