Product sales analysis

Introduction

Product sales analysis is a critical business practice that involves the systematic examination of sales data to gain insights into the performance of products in the market.

This analysis provides valuable information that empowers businesses to make data-driven decisions, refine strategies, and ultimately maximize profitability.

From data collection to actionable insights, this overview will guide you through the key components and steps involved in product sales analysis.

COMPONENT

1. Data Collection
2. Data Cleaning and Preparation
3. Key Performance Metrics
4. Key Performance Metrics
5. Key Performance Metrics
6. Product Performance Analysis
7. Customer Analysis
8. Sales Channel Analysis
9. Competitive Analysis
10. Inventory Management
11. Forecasting and Planning
12. Data Visualization

13. Actionable Insights

14. Continuous Monitoring

1. Data Collection

The first step in product sales analysis is gathering comprehensive sales data. This data includes sales transactions, product details, customer information, and sales channels. It is collected from various sources such as point-of-sale (POS) systems, e-commerce platforms, and CRM software. High-quality data is the foundation of effective analysis.

2. Data Cleaning and Preparation

Once the data is collected, it must be cleansed and prepared for analysis.

This involves removing duplicates, correcting errors, and ensuring data consistency.

Data should also be formatted appropriately, with standardized product names, dates, and currency to facilitate meaningful analysis.

3.Key Performance Metrics

Define the key performance metrics (KPIs) that will be used to assess sales effectiveness.

These metrics include total revenue, gross profit margin, sales growth rate, average order value (AOV), customer acquisition cost (CAC), customer lifetime value (CLV), and inventory turnover rate.

Establishing clear KPIs is essential for measuring success.

4.Segmentation and Categorization

Segment sales data based on various criteria such as product categories, customer demographics, geographic regions, and sales channels. Categorize products based on attributes like popularity and seasonality.

This segmentation allows for more granular analysis and targeted decision-making.

5.Trend Analysis

Analyze historical sales data to identify trends over time. Look for patterns, seasonality, and cyclical trends that impact product sales.

Understanding these trends helps in forecasting and adapting strategies to market dynamics.

6. Product Performance Analysis

Evaluate the performance of individual products or product categories. Identify best-sellers and slow-moving inventory. Examine factors contributing to the success or failure of specific products, guiding decisions on product development and marketing strategies.

7.Customer Analysis

Understand customer behavior, preferences, and buying habits. Segment customers into groups based on their purchasing history. Identify high-value customers and those at risk of churn. Tailor marketing and retention efforts accordingly.

8.Sales Channel Analysis

Evaluate the effectiveness of different sales channels, such as online, retail, and wholesale. Determine which channels generate the highest sales and profitability.

Optimize resource allocation based on channel performance.

9.Competitive Analysis

Compare your product sales performance with that of competitors. Analyze market share, pricing strategies, and customer perception. This analysis helps refine pricing and positioning strategies.

10.Inventory Management

Optimize inventory levels to meet demand without overstocking or understocking. Identify slow-moving inventory that may require promotions or clearance.

Efficient inventory management ensures resources are used effectively.

11.Forecasting and Planning

Use historical sales data to forecast future sales. Plan inventory, marketing campaigns, and resource allocation based on sales projections. Accurate forecasting minimizes risks and maximizes opportunities.

12.Data Visualization

Create visual representations of sales data, such as charts, graphs, dashboards, and reports. Visualization makes complex data more understandable and facilitates quicker decision-making.

13.Actionable Insights

Extract actionable insights from the analysis. Use these insights to make informed business decisions. Implement strategies to improve sales performance, such as pricing adjustments, marketing campaigns, and product launches.

14.Continuous Monitoring

Regularly update and monitor sales data to track the impact of implemented strategies. Adapt strategies as market conditions change. Continuous monitoring ensures agility and responsiveness.

Conclusion:

Product sales analysis is a dynamic and iterative process that empowers businesses to optimize operations, increase revenue, and enhance customer satisfaction. By harnessing the power of data, organizations can stay competitive, adapt to market shifts, and drive long-term success. It is an essential practice for any business committed to thriving in a constantly evolving marketplace.

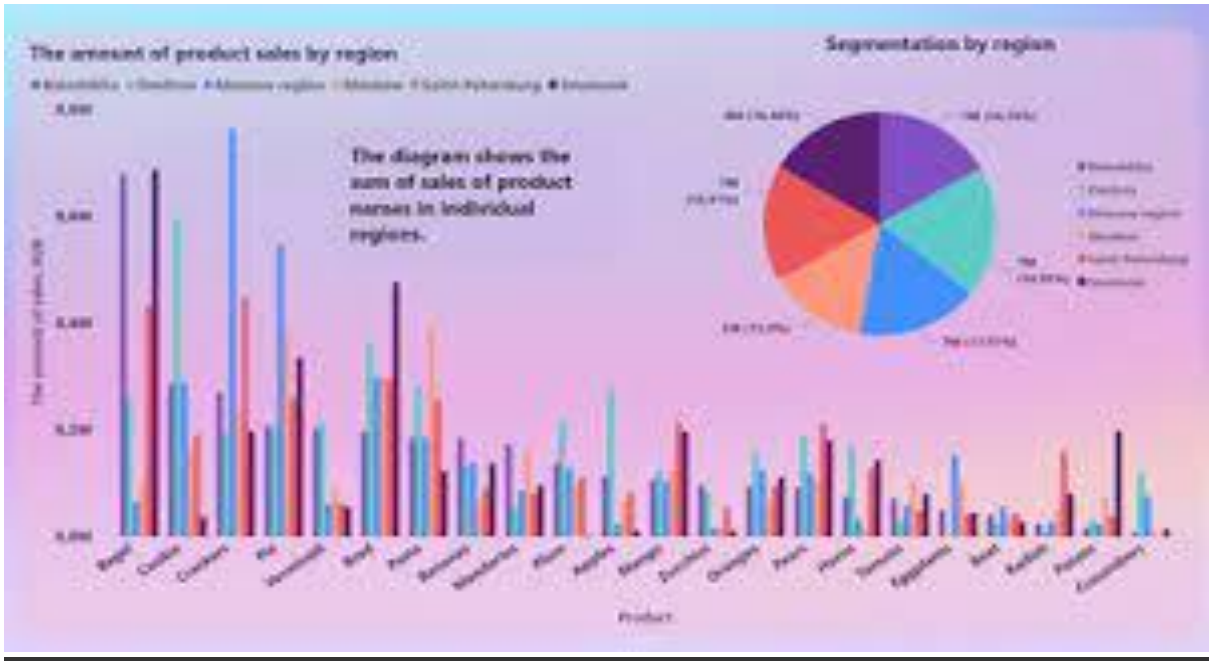
PRODUCT SALES ANALYSIS

Team memebers

- 511921104030 - Karthick.K
- 511921104034 - Madhan.N
- 511921104004 - Ajay Raj .V
- 511921104032 - Lokesh .H

Phase 2 submission document

Project Name : product sales analysis



INTRODUCTION :

Product sales analysis is a crucial process for businesses to understand their sales performance and make informed decisions.

By following these steps, you can conduct a thorough product sales analysis that helps your business make data-driven decisions and improve sales performance.

STEP 1: Gather Data

Collect all relevant data about your product sales. This includes sales records, transaction data, customer information, and any other relevant data sources. Ensure data accuracy and completeness.

STEP 2: Define Objectives

Clearly define the objectives of your sales analysis. Are you trying to identify top-selling products, understand seasonality, or optimize pricing strategies? Having clear objectives will guide your analysis.

STEP 3: Data Cleaning and Preparation

Clean and preprocess the data to remove errors, duplicates, and inconsistencies. Make sure all data is in a consistent format. This may involve data normalization, handling missing values, and data transformation.

STEP 4: Segmentation

Divide your sales data into meaningful segments. This can include product categories, geographical regions, customer demographics, or any other relevant factors. Segmentation helps identify trends and patterns.

STEP 5: Calculate Key Metrics

Calculate essential sales metrics, such as total revenue, sales volume, average order value, and gross margin. These metrics provide a baseline for your analysis.

STEP 6 : Time Series Analysis

Analyze sales data over time to identify trends, seasonality, and growth patterns. Use techniques like moving averages and time series decomposition to gain insights.

STEP 7 : Product Performance Analysis

Evaluate the performance of individual products. Identify best-selling products, slow-moving items, and those with the highest profit margins. This helps in inventory management and product development decisions.

STEP 8 : Customer Analysis

Analyze customer behavior. Identify your top customers, their purchasing habits, and lifetime value. This can help in customer retention and acquisition strategies.

STEP 9 : Market and Competition Analysis

Research the market and your competitors. Compare your product sales with market trends and competitor performance to identify opportunities and threats.

STEP 10 : Pricing Analysis

Analyze the impact of pricing on sales. Determine if price changes affect sales volumes and margins. Test different pricing strategies if needed.

STEP 11: Promotions and Campaign Analysis

Analyze the impact of marketing promotions and campaigns on product sales. Identify which marketing efforts are the most effective in driving sales.

STEP 12 :Inventory Management

Use sales analysis to optimize inventory levels. Ensure that you have enough stock of popular products and minimize excess inventory of slow-moving items.

STEP 13 : Visualize Data

Create charts, graphs, and visual representations of the data to make it easier to understand and share insights with stakeholders. Tools like Excel, Tableau, or Python libraries like Matplotlib and Seaborn can be useful for this.

STEP 14 : Draw Insights and Conclusions

Based on your analysis, draw meaningful insights and conclusions.Use statistical analysis, correlation tests, and other relevant methods to support your findings.

STEP 15 : Recommendations

Provide actionable recommendations based on your insights. These recommendations should be aligned with the objectives you defined in the beginning.

STEP 16 : Report and Presentation

Create a comprehensive report or presentation to share your findings and recommendations with relevant stakeholders

such as management, marketing teams, and sales teams.

STEP 17 : Monitor and Iterate

Sales analysis is an ongoing process. Continuously monitor sales data, and iterate your analysis to adapt to changing market conditions and business goals.

Sales Analysis

The main aim of developing this Product sales analysis application is to analyze the total sales data in a company year wise, month wise and week wise and day wise.

This analysis includes the ratio of sales data, profit maximization, etc.

The Analysis has done with the below 4 Charts namely Bar Chart, Pie Chart, Pivot Table and Straight Table.,

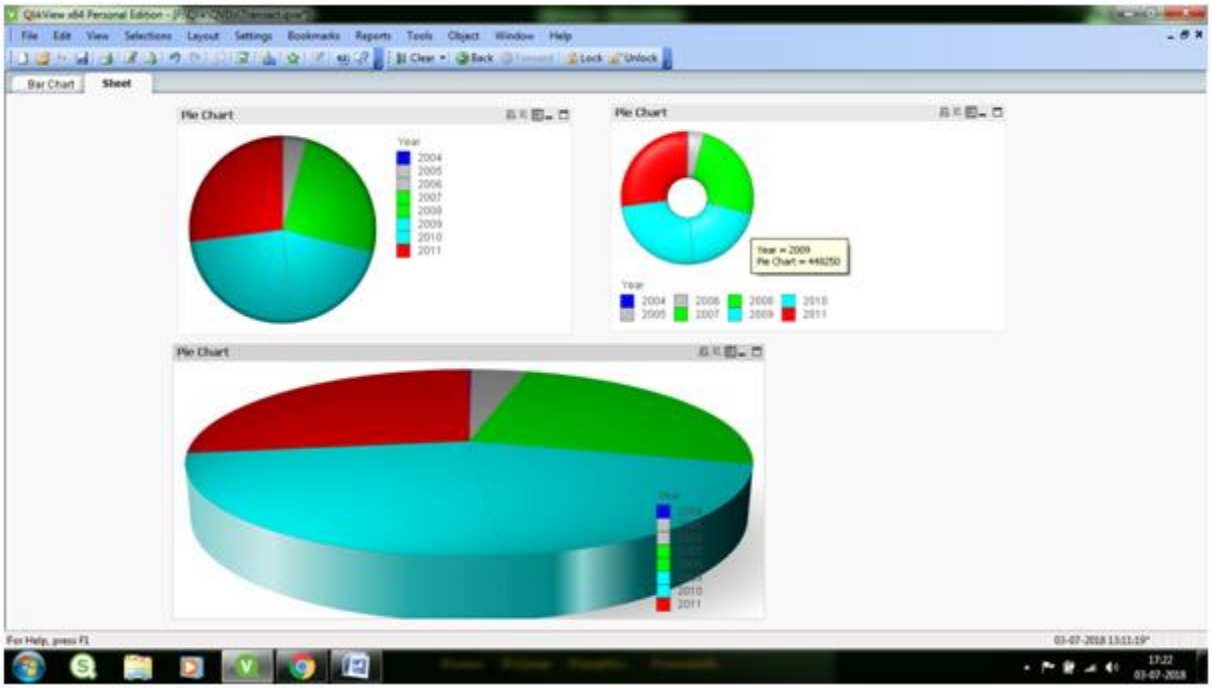
Bar Chart:

A bar graph shows comparisons among discrete categories. One axis of the chart shows the specific categories being compared, and the other axis represents a measured value.

In the above screen, it shows the different types to represent the Bar Chart (Sum of Sales per Year)

Pie Chart:

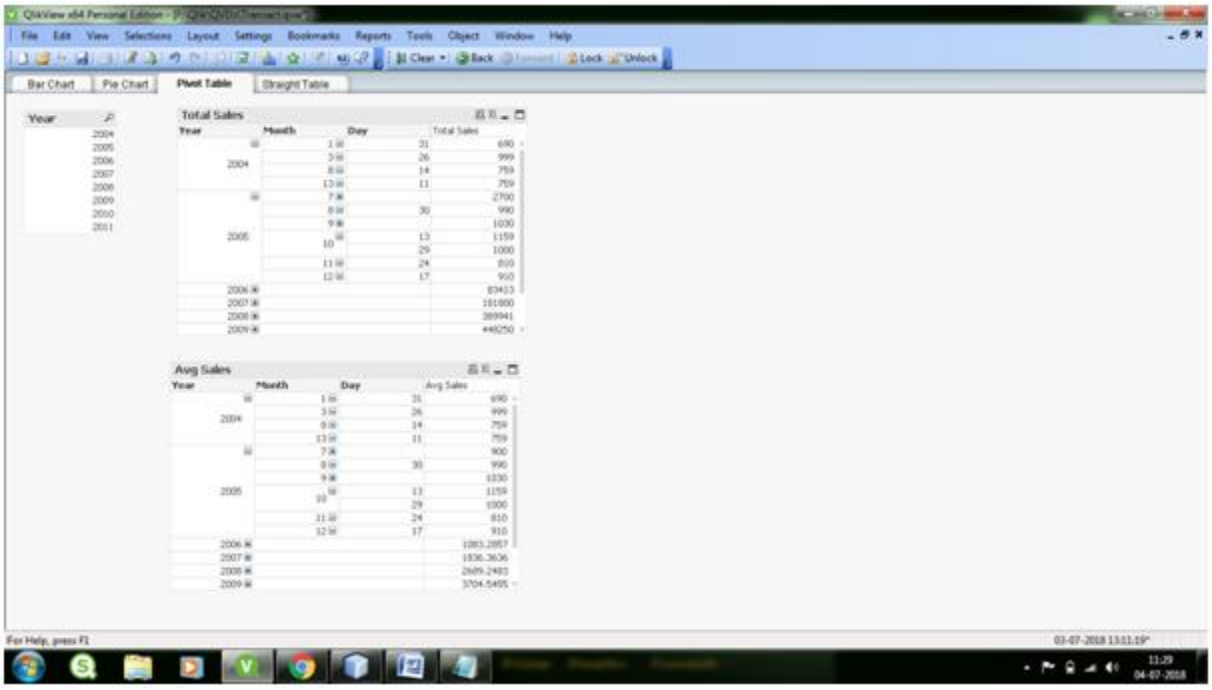
A pie chart (or a circle chart) is a circular statistical graphic which is divided into slices to illustrate the numerical proportion



In the above Screen, it shows the different ways to represent the Pie chart (Sales per Year)

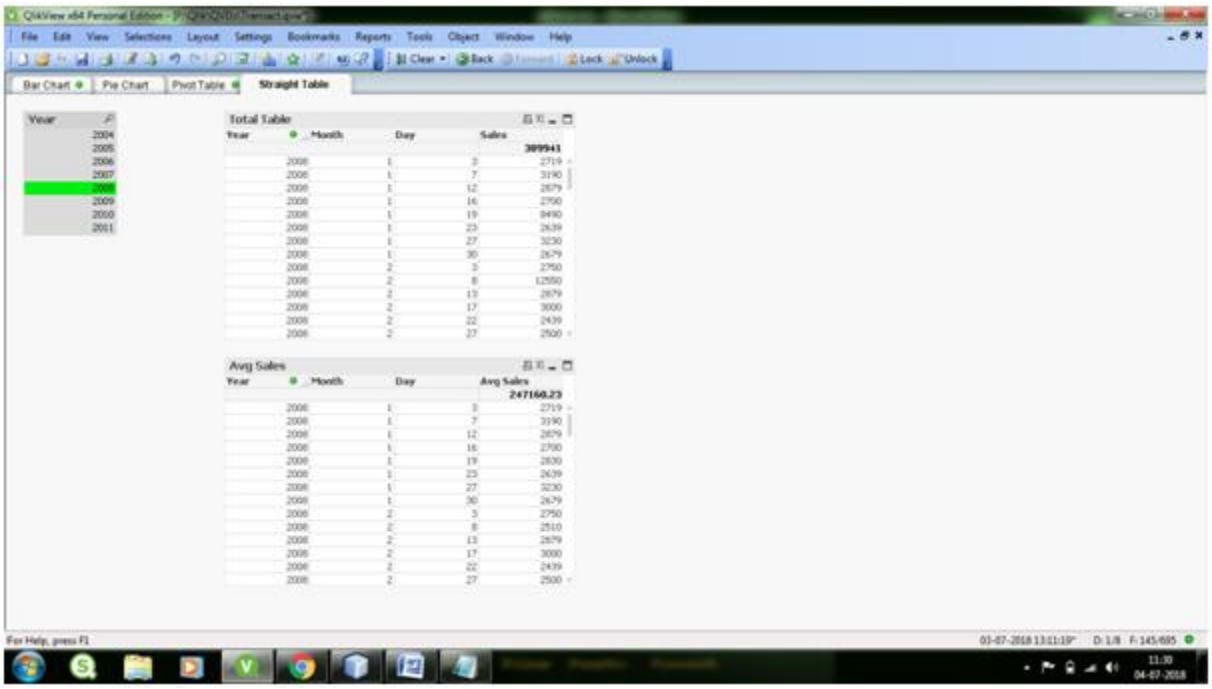
Pivot Table:

Pivot Tables are widely used in data analysis to present the sum of values across many dimensions available in the data.



Straight Table:

Straight Tables are the most widely used sheet object to display data in QlikView. They are very simple yet powerful with features like column rearrangement, sorting and coloring the background etc.



Conclusion :

Product sales analysis is a dynamic and iterative process that empowers businesses to optimize operations, increase revenue, and enhance customer satisfaction. By harnessing the power of data, organizations can stay competitive, adapt to market shifts, and drive long-term success. It is an essential practice for any business committed to thriving in a constantly evolving marketplace

PRODUCT SALES ANALYSIS

PHASE-3

INTRODUCTION:

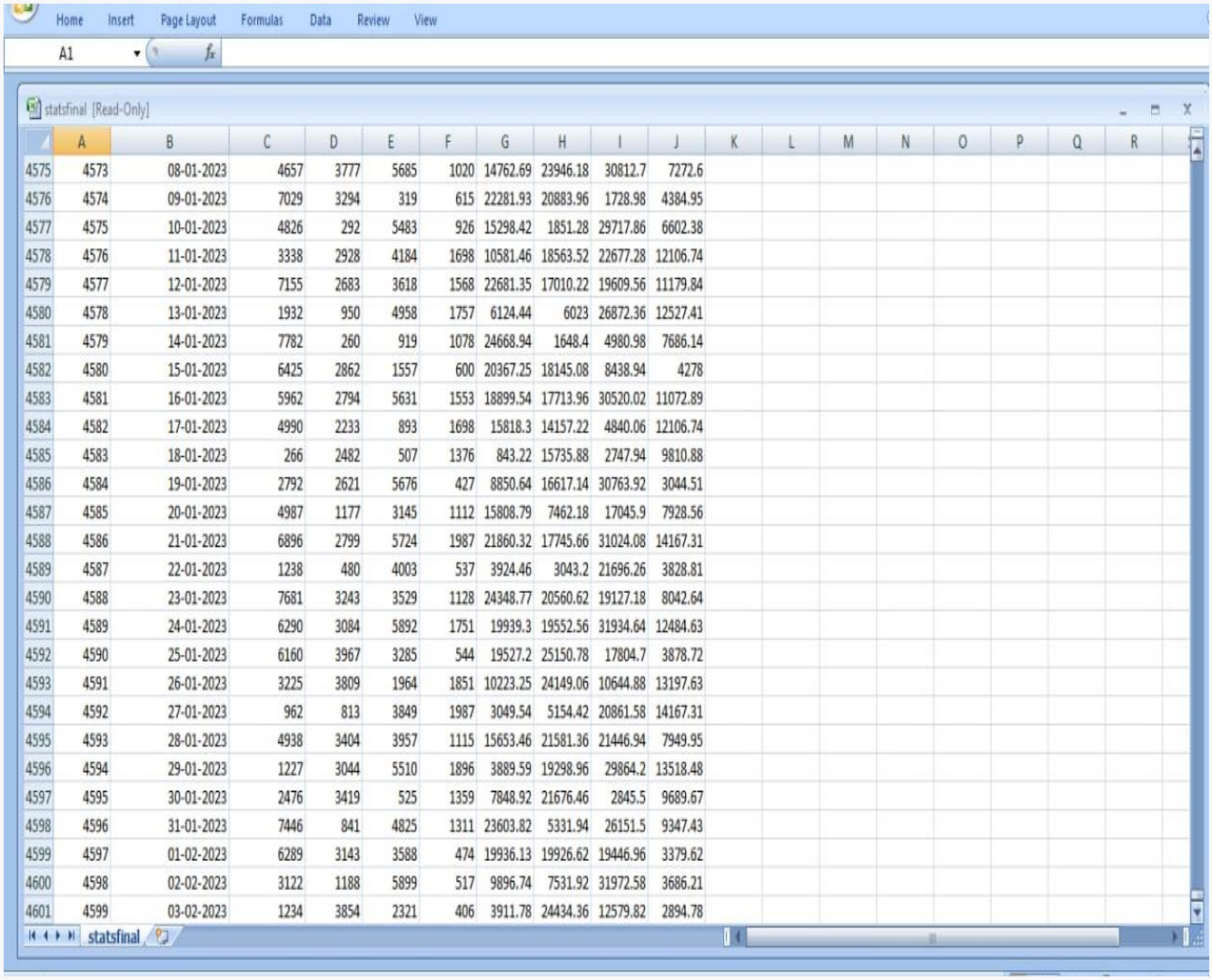
In this section we are presenting the data set for my Product Sales Analysis projects. And development the phase 2 section also. We are attaching the data set link in below content.

Data Set :

A data set is a collection of the whole data. Whether you want to work with predictions or classification, these datasets are both interesting and helpful for machine learning projects. The data is relatively clean and lends nicely to machine learning. Plenty of variables that can help make predictions for the target column.

CLICK HERE,

Dataset Link: https://www.kaggle.com/datasets/ksabishek/product-sales-data



	A	B	C	D	E	F	G	H	I	J
4575	4573	08-01-2023	4657	3777	5685	1020	14762.69	23946.18	30812.7	7272.6
4576	4574	09-01-2023	7029	3294	319	615	22281.93	20883.96	1728.98	4384.95
4577	4575	10-01-2023	4826	292	5483	926	15298.42	1851.28	29717.86	6602.38
4578	4576	11-01-2023	3338	2928	4184	1698	10581.46	18563.52	22677.28	12106.74
4579	4577	12-01-2023	7155	2683	3618	1568	22681.35	17010.22	19609.56	11179.84
4580	4578	13-01-2023	1932	950	4958	1757	6124.44	6023	26872.36	12527.41
4581	4579	14-01-2023	7782	260	919	1078	24668.94	1648.4	4980.98	7686.14
4582	4580	15-01-2023	6425	2862	1557	600	20367.25	18145.08	8438.94	4278
4583	4581	16-01-2023	5962	2794	5631	1553	18899.54	17713.96	30520.02	11072.89
4584	4582	17-01-2023	4990	2233	893	1698	15818.3	14157.22	4840.06	12106.74
4585	4583	18-01-2023	266	2482	507	1376	843.22	15735.88	2747.94	9810.88
4586	4584	19-01-2023	2792	2621	5676	427	8850.64	16617.14	30763.92	3044.51
4587	4585	20-01-2023	4987	1177	3145	1112	15808.79	7462.18	17045.9	7928.56
4588	4586	21-01-2023	6896	2799	5724	1987	21860.32	17745.66	31024.08	14167.31
4589	4587	22-01-2023	1238	480	4003	537	3924.46	3043.2	21696.26	3828.81
4590	4588	23-01-2023	7681	3243	3529	1128	24348.77	20560.62	19127.18	8042.64
4591	4589	24-01-2023	6290	3084	5892	1751	19939.3	19552.56	31934.64	12484.63
4592	4590	25-01-2023	6160	3967	3285	544	19527.2	25150.78	17804.7	3878.72
4593	4591	26-01-2023	3225	3809	1964	1851	10223.25	24149.06	10644.88	13197.63
4594	4592	27-01-2023	962	813	3849	1987	3049.54	5154.42	20861.58	14167.31
4595	4593	28-01-2023	4938	3404	3957	1115	15653.46	21581.36	21446.94	7949.95
4596	4594	29-01-2023	1227	3044	5510	1896	3889.59	19298.96	29864.2	13518.48
4597	4595	30-01-2023	2476	3419	525	1359	7848.92	21676.46	2845.5	9689.67
4598	4596	31-01-2023	7446	841	4825	1311	23603.82	5331.94	26151.5	9347.43
4599	4597	01-02-2023	6289	3143	3588	474	19936.13	19926.62	19446.96	3379.62
4600	4598	02-02-2023	3122	1188	5899	517	9896.74	7531.92	31972.58	3686.21
4601	4599	03-02-2023	1234	3854	2321	406	3911.78	24434.36	12579.82	2894.78

Necessary step to follow:

1. **Import Libraries:**

Start by importing the necessary libraries,

PROGRAM :

```
from tabulate import tabulate
table=[['SERIAL NUMBER','PRODUCTS','PRODUCT SOLD PERCENTAGE','PRODUCT STOCK PERCENTAGE','LOACATION'],
['1','Grocery',19,81,'Tirupattur']
['2','Vegetables',89,11,'Tirupattur']
['3','Fruits',58,42,'Tirupattur']
['4','Cosmetics',0,0,'Tirupattur']
['5','Home Expenditure',42,58,'Tirupattur']]
Print(tabulate(table))
```

```
In [11]: runfile('C:/Users/SMILEYROCKE/Downloads/project.py', wdir='C:/Users/SMILEYROCKE/Downloads')
```

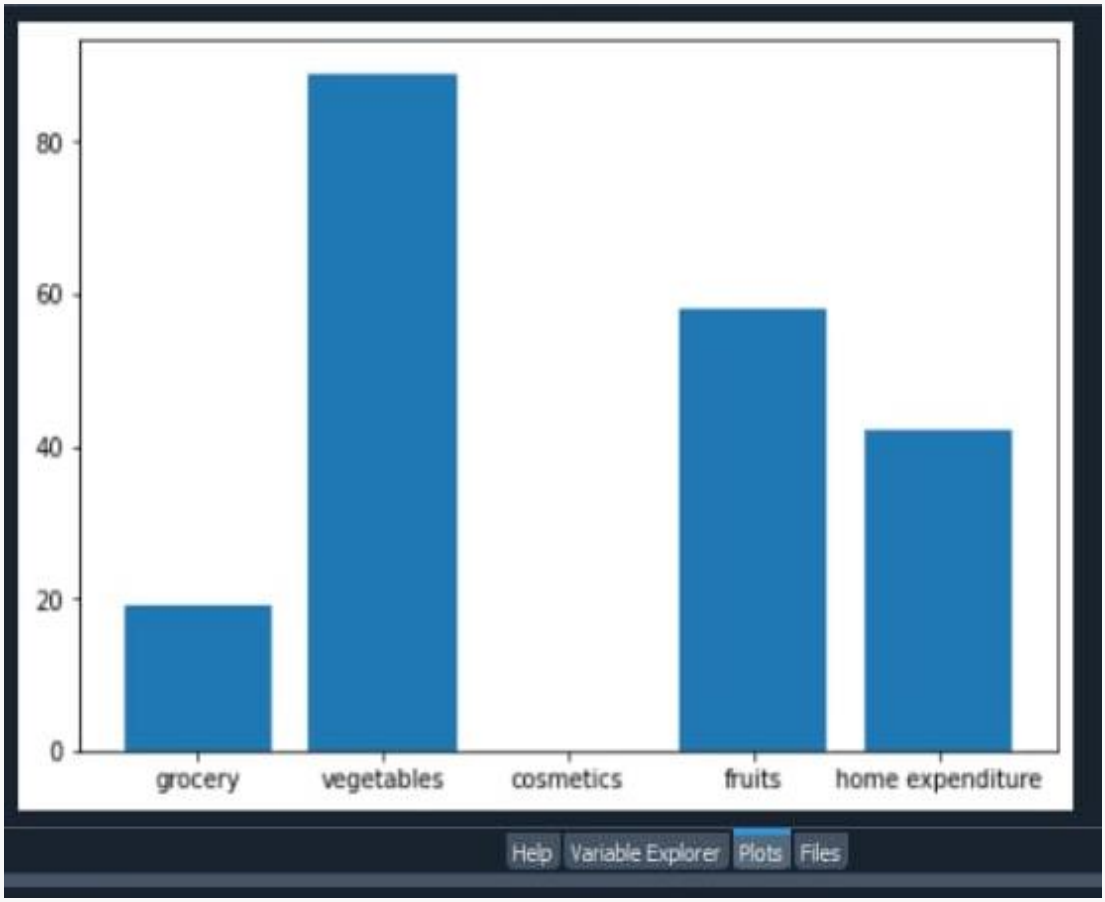
SERIAL NUMBER	PRODUCTS	PRODUCT SOLD PERCENTAGE	PRODUCT STOCK PERCENTAGE	AVAILABLE
1	Grocery	19	81	Tirupattur
2	Vegetables	89	11	Tirupattur
3	fruits	58	42	Tirupattur
4	Cosmetics	0	0	Tirupattur
5	Home Expenditure	42	58	Tirupattur

2. Product sold detail,

We are using the bar chart for this product Sales Analysis. This is very useful to analyze the product details.

PROGRAM :

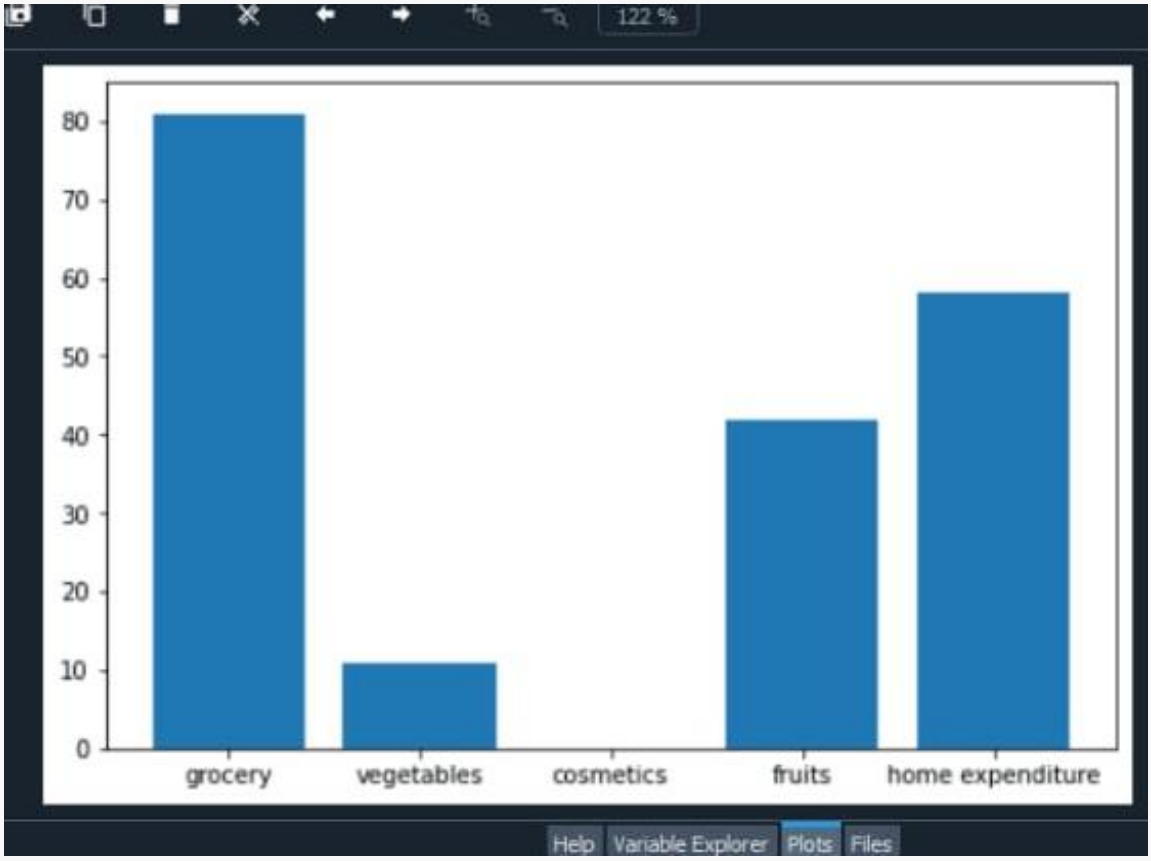
```
Import matplotlib.pyplot as plt

Fig=plt,figure()

ax =fig.add_axes([0,0,1,1])

products=['Grocery','Vegetables','Fruits','Cosmetics','Home Expenditure']

product sold=[19,89,58,0,42]

plt.show ( )
```


80
60
40
20
0
grocery
vegetables
cosmetics
fruits
home expenditure
Help
Variable Explorer
Plots
Files

3.Product stock details,

PROGRAM :

```
Import matplotlib.pyplot as plt

Fig=plt,figure()

ax =fig.add_axes([0,0,1,1])

products=['Grocery','Vegetables','Fruits','Cosmetics','Home Expenditure']

product stock=[81,11,42,0,58]

plt.show ( )
```



Conclusion:

In this section we are develop our ptoject dataset and attaching some files in their section.

PRODUCT SALES ANALYSIS

Team members

- **511921104030 -Karthick. K**
- **511921104034 -Madhan. N**
- **511921104004 - Ajay Raj . V**
- **511921104032- Lokesh. H**

Phase 4 submission document

What city sold the most product.

To answer this question, obviously we need to create a new column called “City” column. How do we get that? As usual, we’re gonna check the top 5 data in our dataframe to figure out where can we get our “City” column using *.head()* method.

Question 2: What city sold the most product?

Task 3: Add a "City" Column

```
In [11]: 1 all_data.head()
```

Out[11]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99

Figure 16. Showing Our Top 5 Updated Dataframe

As you can see at Figure 16, the “Purchase Address” Column contain the city. We can’t get it directly, we need to extract the data. We can use one of most useful function in pandas, *.apply()* method.

Question 2: What city sold the most product?

Task 3: Add a "City" Column

In [15]:
```
all_data['City'] = all_data['Purchase Address'].apply(lambda x: x.split(',')[1])

all_data.head()
```

Out[15]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles

Figure 17. Using .apply() Method to Extract The Data

To make it neater, we can use this

Question 2: What city sold the most product?

Task 3: Add a "City" Column

In [17]:
```
#Function
def get_city(address):
    return address.split(',')[1]

#Extract the city and the state
all_data['City'] = all_data['Purchase Address'].apply(lambda x: get_city(x))

all_data.head()
```

Out[17]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles

Figure 18. Using def function to Make The Code Neater

apply and lambda usually used to create new column based on other column (example .apply(lambda x: x*2. It means every input x in other column will be changed to x*2 in a new column). In this case

we create “City” column based on “Purchase Address” column and we split the data into 3 part. The first one is before the first comma (index = 0), the second one is between the commas (index = 1), and the third on is after the last comma (index = 2). As we need to extract the city data, we use [1] to state it to index 1.

As you can see at Figure 17, we successfully created a “City” column. So are we ready to answer the second question? Not yet. We get an issue here. It’s not error, it’s the value of the “City” Column. This is just a rare case when there are 2 cities are named exactly the same. Example someone in New England and someone in West Coast would think Portland in different way. Someone in New England thinks Portland as Portland Maine and someone in West Coast thinks Portland as Portland Oregon. So in our dataset we actually had the overlapping cities between these two. So, we should also grab the state.

Question 2: What city sold the most product?

Task 3: Add a "City" Column

In [18]:

```
#Function
def get_city(address):
    return address.split(',')[1]

def get_state(address):
    return address.split(',')[2].split(' ')[1]

#Extract the city and the state
all_data['City'] = all_data['Purchase Address'].apply(lambda x: get_city(x) + ' ' + get_state(x))

all_data.head()
```

Out[18]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas TX
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston MA
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles CA
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles CA
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles CA

Figure 19. Extracting the state to “City” Column

The function get_state() basically works as explained before. But in this function, we seperate the data again become three parts. The first one is before the whitespace (index = 0), the second one is between the whitespaces (index = 1), and the third one is after the last whitespace (index = 2). So that’s why we use *.split(‘ ‘)[1]* in the second split.

Now we’re ready to answer the second question, **what city sold the most product**? As we did before, we’re gonna group it by the city and summing all the values based on the group.

```
In [19]:
1 results2 = all_data.groupby('City').sum()
2 results2
```

Out[19]:

City	Quantity Ordered	Price Each	Month	Sales
Atlanta GA	16602	2.779908e+06	104794	2.795499e+06
Austin TX	11153	1.809874e+06	69829	1.819582e+06
Boston MA	22528	3.637410e+06	141112	3.661642e+06
Dallas TX	16730	2.752628e+06	104620	2.767975e+06
Los Angeles CA	33289	5.421435e+06	208325	5.452571e+06
New York City NY	27932	4.635371e+06	175741	4.664317e+06
Portland ME	2750	4.471893e+05	17144	4.497583e+05
Portland OR	11303	1.860558e+06	70621	1.870732e+06
San Francisco CA	50239	8.211462e+06	315520	8.262204e+06
Seattle WA	16553	2.733296e+06	104941	2.747755e+06

Figure 20. Grouping The Dataframe by The City

It’s too messy, but if you look carefully you can see that San Fransisco is the highest sold product of all cities with approximately $8,200,000. We clearly need to visualize it because it’s so hard to conclude anything just based on that numbers and also it will make our bussiness partner easier to understand.

```
In [22]:
#We've already import the matplotlib

cities = all_data['City'].unique()

plt.bar(cities, results2['Sales'])
plt.xticks(cities, rotation='vertical', size = 8)
labels, location = plt.yticks()
plt.yticks(labels, (labels/1000000).astype(int)) #Scaling in million USD
plt.ylabel('Sales in million USD')
plt.xlabel('City Name')
plt.show()
```

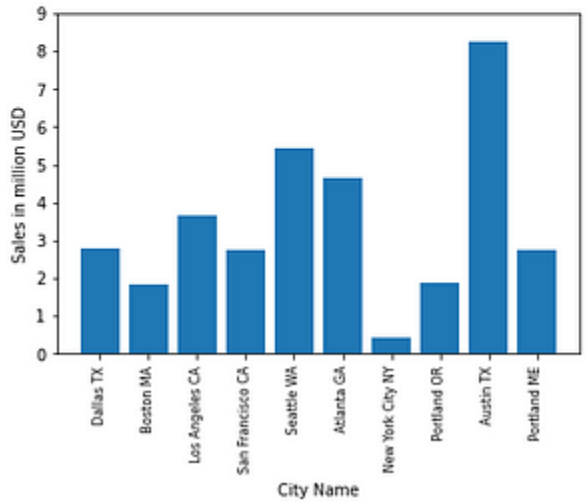


Figure 21. Plotting the Sales Grouped by Cities.

Now we successfully plot it. But there is a big issue here. If you notice that the values (Figure 20) and the plot (Figure 21) are not synchronized. The highest sales should be San Fransisco. What's wrong with our code?

There's an issue between *.unique()* method and *plt.bar()*. Their cities order are different. we're gonna syncronized the order by simply fixing the variable 'cities'.

```
In [23]:
#We've already import the matplotlib

#Fixing the cities order
#cities = all_data['City'].unique()
cities = [city for city, df in all_data.groupby('City')]

plt.bar(cities, results2['Sales'])
plt.xticks(cities, rotation='vertical', size = 8)
labels, location = plt.yticks()
plt.yticks(labels, (labels/1000000).astype(int)) #Scaling in million USD
plt.ylabel('Sales in million USD')
plt.xlabel('City Name')
plt.show()
```

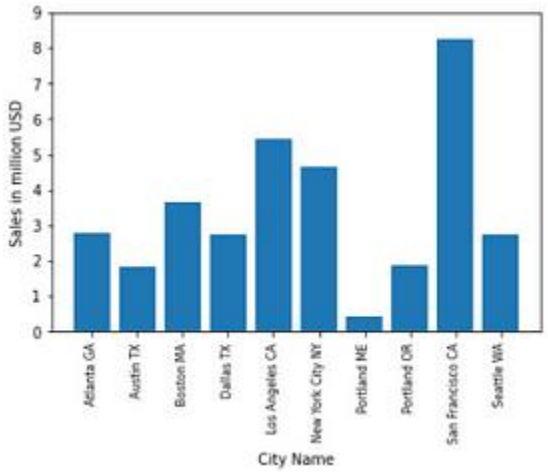


Figure 22. Fixing The Code

Now, we fix the issue and successfully plot it. As a data scientist, we need to figure out why San Fransisco is the highest sale compare to other cities. Maybe Sillicon Valley need more electronic products. Maybe the advertisement is better in San Fransisco.

We can use this data to improve the sales of bussiness.

What Time Should We Display Advertisements to Maximize Likelihood of Customer's Buying Product.

As usual, to remind what our data look like, we use the .head() method to show the top 5 of our updated dataframe.

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

```
In [24]: 1 all_data.head()
```

Out[24]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas TX
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston MA
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles CA
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles CA
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles CA

Figure 23. Showing our Top 5 updated dataframe

If we're gonna use our data to answer this question, we need to aggregate the period in 24 hours distribution. Look carefully at Figure 23. In "Order Date" column, there are times data. We could extract it like we did before.

But to make it more consistent, we need to convert the "Order Date" Column into date time object.

We're gonna use *pd.to_datetime()* method.

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

Task 4: Aggregate the period in 24-hours distribution

In [29]:
```
#Create new column in date-time Object (DTO)
all_data['Order_Date_DTO'] = pd.to_datetime(all_data['Order Date'])
all_data.head()
```

Out[29]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Order_Date_DTO
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas TX	2019-04-19 08:46:00
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston MA	2019-04-07 22:30:00
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles CA	2019-04-12 14:38:00
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles CA	2019-04-12 14:38:00
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles CA	2019-04-30 09:27:00

Figure 24. Converting the “Order Date” Column into Date-Time Object

It will take a little bit longer because of the heavy calculation. Now we can create a new column called “Hour” contain the extraction of “Order_Date_DTO” data.

We only need the hours data, so we can extract them by doing this.

Task 4: Aggregate the period in 24-hours distribution

In [30]:
```
#Create new column in date-time Object (DTO)
all_data['Order_Date_DTO'] = pd.to_datetime(all_data['Order Date'])

#Extraction the hours data
all_data['Hour'] = all_data['Order_Date_DTO'].dt.hour

all_data.head()
```

Out[30]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Order_Date_DTO	Hour
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas TX	2019-04-19 08:46:00	8
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston MA	2019-04-07 22:30:00	22
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles CA	2019-04-12 14:38:00	14
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles CA	2019-04-12 14:38:00	14
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles CA	2019-04-30 09:27:00	9

Figure 25. Extracting The Hours Data Into The New Column

Now we can answer the third question, **what time should we display advertisements to maximize likelihood of customer's buying product?** To answer this, we're gonna group it by the hours and counting all of the orders.

```
In [40]: 1 results3 = all_data.groupby(['Hour']).count()
         2 results3
         3
```

Out[40]:

Hour	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Order_Date_DTO
0	3910	3910	3910	3910	3910	3910	3910	3910	3910	3910
1	2350	2350	2350	2350	2350	2350	2350	2350	2350	2350
2	1243	1243	1243	1243	1243	1243	1243	1243	1243	1243
3	831	831	831	831	831	831	831	831	831	831
4	854	854	854	854	854	854	854	854	854	854
5	1321	1321	1321	1321	1321	1321	1321	1321	1321	1321
6	2482	2482	2482	2482	2482	2482	2482	2482	2482	2482
7	4011	4011	4011	4011	4011	4011	4011	4011	4011	4011
8	6256	6256	6256	6256	6256	6256	6256	6256	6256	6256
9	8748	8748	8748	8748	8748	8748	8748	8748	8748	8748
10	10944	10944	10944	10944	10944	10944	10944	10944	10944	10944
11	12411	12411	12411	12411	12411	12411	12411	12411	12411	12411
12	12587	12587	12587	12587	12587	12587	12587	12587	12587	12587
13	12129	12129	12129	12129	12129	12129	12129	12129	12129	12129
14	10984	10984	10984	10984	10984	10984	10984	10984	10984	10984
15	10175	10175	10175	10175	10175	10175	10175	10175	10175	10175
16	10384	10384	10384	10384	10384	10384	10384	10384	10384	10384
17	10899	10899	10899	10899	10899	10899	10899	10899	10899	10899
18	12280	12280	12280	12280	12280	12280	12280	12280	12280	12280
19	12905	12905	12905	12905	12905	12905	12905	12905	12905	12905
20	12228	12228	12228	12228	12228	12228	12228	12228	12228	12228
21	10921	10921	10921	10921	10921	10921	10921	10921	10921	10921
22	8822	8822	8822	8822	8822	8822	8822	8822	8822	8822
23	6275	6275	6275	6275	6275	6275	6275	6275	6275	6275

Figure 26. Grouping the data by the hours

If we want to answer the third question, we only need the "Quantity Ordered" column. Now let's visualize it. We want it to be the line chart because this spesific data (hours) are more logical to show using line chart than bar chart because the data has to be continue.

```
In [41]:
#Plotting
results3 = all_data.groupby(['Hour'])['Quantity Ordered'].count()
hours = [hour for hour, df in all_data.groupby('Hour')]

plt.plot(hours, results3)
plt.xticks(hours)
plt.xlabel('Hour')
plt.ylabel('Number of Orders')
plt.grid()
plt.show()
```

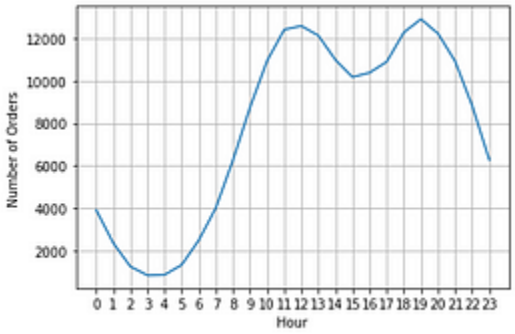


Figure 27. Visualizing the Number of Orders in 24 hours format

As you can see from Figure 27, there are approximately 2 peaks at the data. They are 12 (12 PM) and 19 (7 PM). It makes sense since most people shopping during the day.

From this data, we can suggest to our bussiness partner to advertise their product right before 12 PM and/or 7 PM. It could be 11.30 AM and/or 6.30 PM.

Remember, this chart is the total orders of **all cities**. Maybe you could make a spesific chart for a spesific city and planning the advertisement better for that city.

Conclusion :

In this phase ,which city may sold most products and What Time Should We Display Advertisements to Maximize Likelihood of Customer's Buying Product. for further queries we may answer in next phase which is gonna final phase in the project. In that final phase this section you will document the complete project and prepare it for submission.