PYTHON DATA ANALYSIS PROJECT : ANALYSING FOOD SALES DATA

Submitted to : Prof. Amarnath Mitra

Submitted by : Parth Gautam

PGDM-BDA

045038

Background:

In this project, we conducted an analysis of food sales data to gain meaningful insights into the business's performance. The data was obtained from this source.

Objectives:

The primary objectives of this analysis were as follows:

1. Understand sales trends over time.
2. Identify top-selling products and categories.
3. Analyse regional variations in sales.
4. Discover any seasonal patterns in food sales.

Data Source:

The data for this project was sourced from https://www.contextures.com/excelsampledatafoodsales.html#download. It consists of information on food sales, including date, region, category, product, quantity (Qty), unit price, and total price.

Dataset Overview:

Data Source: Obtained from (https://www.contextures.com/excelsampledatafoodsales.html#download).

Structure: CSV format with columns for ID, Date, Region, City, Category, Product, Qty, UnitPrice and TotalPrice.

Size: Contains multiple rows and columns-sales transactions.

Data types: Includes categorical data for regions, cities, categories and products, along with numerical data for qty and prices.

Purpose: Used for business analysis, tracking sales trends, and optimizing inventory & pricing strategies.

	A	B	C	D	E	F	G	H	I
1	ID	Date	Region	City	Category	Product	Qty	UnitPrice	TotalPrice
2	ID07351	1-Jan	East	Boston	Bars	Carrot	33	1.77	58.41
3	ID07352	4-Jan	East	Boston	Crackers	Whole Wheat	87	3.49	303.63
4	ID07353	7-Jan	West	Los Angeles	Cookies	Chocolate Chip	58	1.87	108.46
5	ID07354	10-Jan	East	New York	Cookies	Chocolate Chip	82	1.87	153.34
6	ID07355	13-Jan	East	Boston	Cookies	Arrowroot	38	2.18	82.84
7	ID07356	16-Jan	East	Boston	Bars	Carrot	54	1.77	95.58
8	ID07357	19-Jan	East	Boston	Crackers	Whole Wheat	149	3.49	520.01
9	ID07358	22-Jan	West	Los Angeles	Bars	Carrot	51	1.77	90.27
10	ID07359	25-Jan	East	New York	Bars	Carrot	100	1.77	177
11	ID07360	28-Jan	East	New York	Snacks	Potato Chips	28	1.35	37.8
12	ID07361	31-Jan	East	Boston	Cookies	Arrowroot	36	2.18	78.48
13	ID07362	3-Feb	East	Boston	Cookies	Chocolate Chip	31	1.87	57.97
14	ID07363	6-Feb	East	Boston	Crackers	Whole Wheat	28	3.49	97.72
15	ID07364	9-Feb	West	Los Angeles	Bars	Carrot	44	1.77	77.88
16	ID07365	12-Feb	East	New York	Bars	Carrot	23	1.77	40.71
17	ID07366	15-Feb	East	New York	Snacks	Potato Chips	27	1.35	36.45
18	ID07367	18-Feb	East	Boston	Cookies	Arrowroot	43	2.18	93.74
19	ID07368	21-Feb	East	Boston	Cookies	Oatmeal Raisin	123	2.84	349.32
20	ID07369	24-Feb	West	Los Angeles	Bars	Bran	42	1.87	78.54
21	ID07370	27-Feb	West	Los Angeles	Cookies	Oatmeal Raisin	33	2.84	93.72
22	ID07371	2-Mar	East	New York	Cookies	Chocolate Chip	85	1.87	158.95
23	ID07372	5-Mar	West	San Diego	Cookies	Oatmeal Raisin	30	2.84	85.2
24	ID07373	8-Mar	East	Boston	Bars	Carrot	61	1.77	107.97
25	ID07374	11-Mar	East	Boston	Crackers	Whole Wheat	40	3.49	139.6
26	ID07375	14-Mar	West	Los Angeles	Cookies	Chocolate Chip	86	1.87	160.82
27	ID07376	17-Mar	East	New York	Bars	Carrot	38	1.77	67.26
28	ID07377	20-Mar	East	New York	Snacks	Potato Chips	68	1.68	114.24
29	ID07378	23-Mar	West	San Diego	Cookies	Chocolate Chip	39	1.87	72.93
30	ID07379	26-Mar	East	Boston	Bars	Bran	103	1.87	192.61
31	ID07380	29-Mar	East	Boston	Cookies	Oatmeal Raisin	193	2.84	548.12
32	ID07381	1-Apr	West	Los Angeles	Bars	Carrot	58	1.77	102.66
33	ID07382	4-Apr	West	Los Angeles	Snacks	Potato Chips	68	1.68	114.24
34	ID07383	7-Apr	East	New York	Bars	Carrot	91	1.77	161.07
35	ID07384	10-Apr	East	New York	Crackers	Whole Wheat	23	3.49	80.27
36	ID07385	13-Apr	West	San Diego	Snacks	Potato Chips	28	1.68	47.04
37	ID07386	16-Apr	East	Boston	Bars	Carrot	48	1.77	84.96
38	ID07387	19-Apr	East	Boston	Snacks	Potato Chips	134	1.68	225.12
39	ID07388	22-Apr	West	Los Angeles	Bars	Carrot	20	1.77	35.4
40	ID07389	25-Apr	East	New York	Bars	Carrot	53	1.77	93.81
41	ID07390	28-Apr	East	New York	Snacks	Potato Chips	64	1.68	107.52
42	ID07391	1-May	West	San Diego	Cookies	Chocolate Chip	63	1.87	117.81
43	ID07392	4-May	East	Boston	Bars	Bran	105	1.87	196.35
44	ID07393	7-May	East	Boston	Cookies	Oatmeal Raisin	138	2.84	391.92
45	ID07394	10-May	West	Los Angeles	Bars	Carrot	25	1.77	44.25
46	ID07395	13-May	West	Los Angeles	Crackers	Whole Wheat	21	3.49	73.29
47	ID07396	16-May	East	New York	Bars	Carrot	61	1.77	107.97
48	ID07397	19-May	East	New York	Snacks	Potato Chips	49	1.68	82.32
49	ID07398	22-May	West	San Diego	Cookies	Chocolate Chip	55	1.87	102.85
50	ID07399	25-May	East	Boston	Cookies	Arrowroot	27	2.18	58.86
51	ID07400	28-May	East	Boston	Bars	Carrot	58	1.77	102.66
52	ID07401	31-May	East	Boston	Crackers	Whole Wheat	33	3.49	115.17
53	ID07402	3-Jun	West	Los Angeles	Cookies	Oatmeal Raisin	288	2.84	817.92
54	ID07403	6-Jun	East	New York	Cookies	Chocolate Chip	76	1.87	142.12
55	ID07404	9-Jun	West	San Diego	Bars	Carrot	42	1.77	74.34
56	ID07405	12-Jun	West	San Diego	Crackers	Whole Wheat	20	3.49	69.8
57	ID07406	15-Jun	East	Boston	Bars	Carrot	75	1.77	132.75
58	ID07407	18-Jun	East	Boston	Crackers	Whole Wheat	38	3.49	132.62
59	ID07408	21-Jun	West	Los Angeles	Bars	Carrot	306	1.77	541.62
60	ID07409	24-Jun	West	Los Angeles	Snacks	Potato Chips	28	1.68	47.04
61	ID07410	27-Jun	East	New York	Bars	Bran	110	1.87	205.7
62	ID07411	30-Jun	East	New York	Cookies	Oatmeal Raisin	51	2.84	144.84

DESCRIPTIVE ANALYSIS AND DATA VISUALIZATION(CHARTS) USING PYTHON :

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate summary statistics for 'UnitPrice'
unit_price_summary = data['UnitPrice'].describe()

# Print the summary statistics for unit price
print(unit_price_summary)
```

```
count    61.000000
mean      2.170656
std       0.659037
min       1.350000
25%       1.770000
50%       1.870000
75%       2.840000
max       3.490000
Name: UnitPrice, dtype: float64
```

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Group the data by 'Category' and calculate summary statistics for 'UnitPrice'
category_summary = data.groupby('Category')['UnitPrice'].describe()

# Print the summary statistics for unit price
print(category_summary)
```

```
          count      mean           std   min   25%   50%   75%   max
Category
Bars       23.0  1.787391  3.875534e-02  1.77  1.77  1.77  1.77  1.87
Cookies    20.0  2.271500  4.440635e-01  1.87  1.87  2.18  2.84  2.84
Crackers    9.0  3.490000  9.420555e-16  3.49  3.49  3.49  3.49  3.49
Snacks      9.0  1.606667  1.455163e-01  1.35  1.68  1.68  1.68  1.68
```

- Analysing the mean unit price and conducting statistical analysis provides insights into pricing patterns and strategies. The mean unit price provides the average price per unit across all products and transactions. Key takeaways include understanding average pricing, detecting outliers, assessing price variability, tracking trends, and aligning with business goals. This analysis aids in optimizing pricing, product segmentation, and profitability decisions.

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate summary statistics for 'TotalPrice'
total_price_summary = data['TotalPrice'].describe()

# Print the summary statistics for total price
print(total_price_summary)
```

```
count     61.000000
mean     148.096066
std      144.057853
min       35.400000
25%       77.880000
50%      102.660000
75%      153.340000
max      817.920000
Name: TotalPrice, dtype: float64
```

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate summary statistics for 'TotalPrice' based on different categories
category_total_price_summary = data.groupby('Category')['TotalPrice'].describe()

# Print the summary statistics for total price by category
print(category_total_price_summary)
```

```
          count        mean         std    min    25%      50%       75%  \
Category
Bars       23.0  124.772609  103.661032  35.40  76.11   95.580  146.9100
Cookies    20.0  191.010500  194.312306  57.97  84.61  113.135  159.4175
Crackers    9.0  170.234444  149.315091  69.80  80.27  115.170  139.6000
Snacks      9.0   90.196667   60.292744  36.45  47.04   82.320  114.2400

             max
Category
Bars      541.62
Cookies   817.92
Crackers  520.01
Snacks    225.12
```

- Analysing the mean total price and conducting statistical analysis on total prices provides insights into revenue patterns and variability. Examining categorical total prices further reveals revenue variations among product categories. The mean total price represents the average transaction revenue, while statistical analysis uncovers distribution characteristics and identifies outliers. A high standard deviation indicates revenue fluctuations. Tracking changes in the mean total price over time reflects pricing strategies' effectiveness. Comparing total prices within categories aids in assessing competitiveness, aligning with customer behaviour, and optimizing profitability. These insights inform pricing, marketing strategies, and decisions to enhance financial performance.

```
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate summary statistics for 'Qty'
qty_summary = data['Qty'].describe()

# Print the summary statistics for Qty
print(qty_summary)
```

```
count     61.000000
mean      67.360656
std       55.665080
min       20.000000
25%       33.000000
50%       51.000000
75%       82.000000
max      306.000000
Name: Qty, dtype: float64
```

```
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate summary statistics for 'Qty' based on different categories
category_qty_summary = data.groupby('Category')['Qty'].describe()

# Print the summary statistics for Qty by category
print(category_qty_summary)
```

```
          count       mean        std   min   25%   50%    75%    max
Category
Bars       23.0  69.608696  58.068720  20.0  42.0  54.0  83.00  306.0
Cookies    20.0  78.750000  64.971147  27.0  37.5  56.5  85.25  288.0
Crackers    9.0  48.777778  42.783694  20.0  23.0  33.0  40.00  149.0
Snacks      9.0  54.888889  34.761489  27.0  28.0  49.0  68.00  134.0
```

- Analysing the mean quantity (Qty) and conducting statistical analysis on Qty provides insights into sales trends and variability. Additionally, examining categorical Qty offers insights into how sales quantities vary across different product categories. The mean Qty serves as the average quantity sold per transaction, while statistical analysis unveils distribution patterns and identifies outliers. A high standard deviation indicates significant variability in sales quantities. Monitoring changes in the mean Qty over time reflects the effectiveness of sales and demand strategies. Comparing Qty within product categories helps assess market competitiveness, customer preferences, and inventory management. These insights inform sales strategies and inventory optimization decisions to enhance overall business performance.

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Get unique categories
unique_categories = data['Category'].unique()

# Get unique products
unique_products = data['Product'].unique()

# Display unique categories
print("Unique Categories:")
for category in unique_categories:
    print(category)

# Display unique products
print("\nUnique Products:")
for product in unique_products:
    print(product)
```

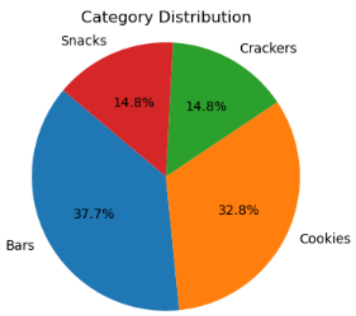
```
Unique Categories:
Bars
Crackers
Cookies
Snacks

Unique Products:
Carrot
Whole Wheat
Chocolate Chip
Arrowroot
Potato Chips
Oatmeal Raisin
Bran
```

- These are the unique categories and products available in the dataset.

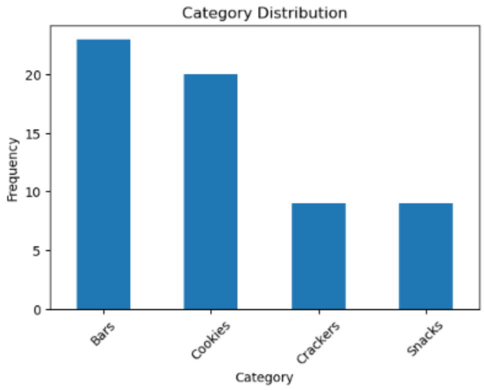
```python
category_counts = data['Category'].value_counts()
print(category_counts)
```

```
Bars        23
Cookies     20
Crackers     9
Snacks       9
Name: Category, dtype: int64
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate the frequency distribution
category_counts = data['Category'].value_counts()

# Create a pie chart
plt.figure(figsize=(4, 4))  # Optional: Set the figure size
plt.pie(category_counts, labels=category_counts.index, autopct='%1.1f%%', startangle=140)
plt.axis('equal')  # Equal aspect ratio ensures that the pie is circular.

# Optional: Add a title
plt.title('Category Distribution')

# Display the pie chart
plt.show()
```

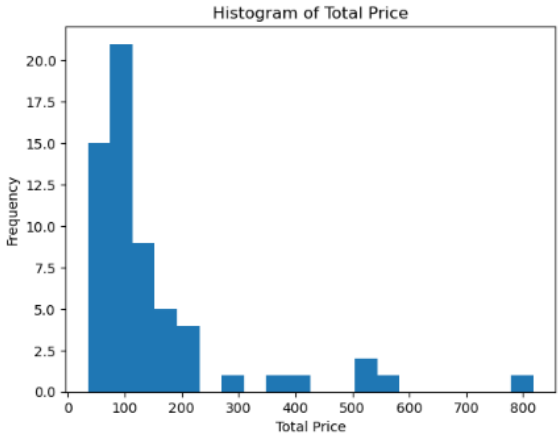


```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate the frequency distribution
category_counts = data['Category'].value_counts()

# Create a bar graph
plt.figure(figsize=(6, 4))  # Optional: Set the figure size
category_counts.plot(kind='bar')
plt.xlabel('Category')
plt.ylabel('Frequency')
plt.title('Category Distribution')

# Rotate x-axis labels for better readability (optional)
plt.xticks(rotation=45)

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

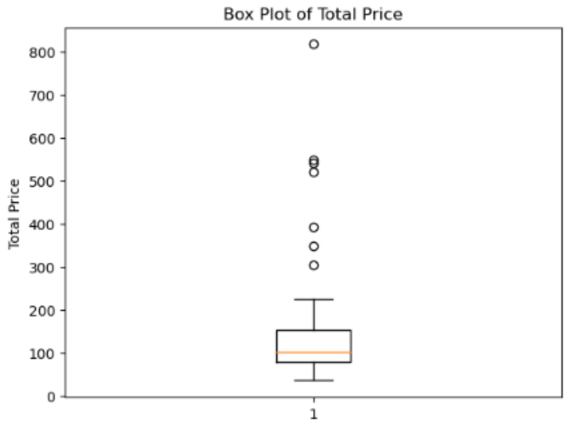


- Category frequency distribution reveals the occurrence of each product category within a dataset. It's valuable for gauging category popularity, assessing market emphasis, and understanding product segmentation. High-frequency categories may signify strong customer demand and marketing focus, while diverse category representation indicates a broad product portfolio. Analysing category frequencies aids in inventory management and tailoring marketing strategies to customer preferences. It also helps identify market trends and overall category performance, guiding business decisions related to product assortment and market positioning.

```python
import matplotlib.pyplot as plt

plt.hist(data['TotalPrice'], bins=20)
plt.xlabel('Total Price')
plt.ylabel('Frequency')
plt.title('Histogram of Total Price')
plt.show()
```




```python
plt.boxplot(data['TotalPrice'])
plt.ylabel('Total Price')
plt.title('Box Plot of Total Price')
plt.show()
```

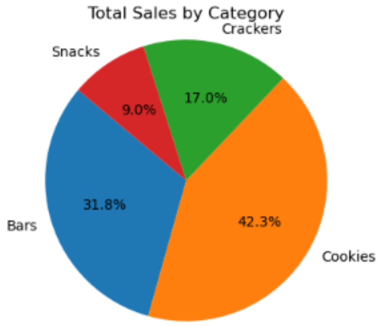


- Box plots of total price reveal central tendency, spread, and outliers. The median shows central pricing, while a wider box indicates greater variability. Outliers, located outside the whiskers, suggest unusual transactions. Histograms display total price distribution shape and mode. Peaks indicate common price ranges, aiding in identifying pricing tiers or clusters. Histogram width shows data spread, with a narrower distribution indicating consistent pricing. Understanding these visualizations aids in pricing strategies, identifying anomalies, and analysing the distribution's characteristics.

```python
category_sales = data.groupby('Category')['TotalPrice'].sum()
print(category_sales)
```

```
Category
Bars        2869.77
Cookies     3820.21
Crackers    1532.11
Snacks       811.77
Name: TotalPrice, dtype: float64
```

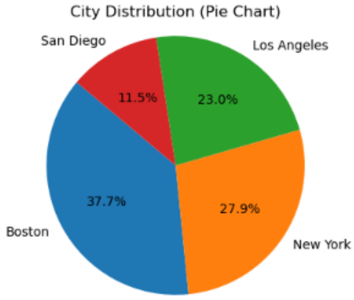
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate total sales for each category
category_sales = data.groupby('Category')['TotalPrice'].sum()

# Create a pie chart
plt.figure(figsize=(4, 4))  # Optional: Set the figure size
plt.pie(category_sales, labels=category_sales.index, autopct='%1.1f%%', startangle=140)
plt.axis('equal')  # Equal aspect ratio ensures that the pie is circular.

# Optional: Add a title
plt.title('Total Sales by Category')

# Display the pie chart
plt.show()
```



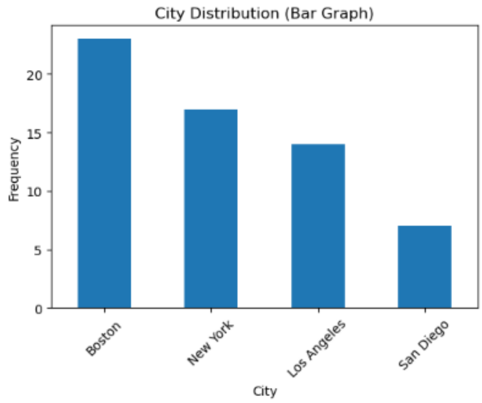
- The sum of categorical total prices offers insights into revenue distribution among product categories. It identifies high-performing categories contributing the most to overall revenue. This data aids in market segmentation, highlighting categories that cater to larger customer segments. Additionally, it informs pricing strategies, allowing businesses to assess the effectiveness of pricing within each category. By comparing sums across categories, market share can be gauged, guiding strategic decisions. Profitability assessments can be made by evaluating total prices in relation to costs. Furthermore, it emphasizes the need for effective inventory management in categories with higher revenue sums. In summary, understanding the sum of categorical total prices helps optimize category-specific strategies, marketing efforts, and resource allocation.

```python
import pandas as pd

data = pd.read_csv('FoodSales.csv')

# Calculate the frequency distribution of the 'City' column
city_distribution = data['City'].value_counts()

# Print the frequency distribution
print(city_distribution)
```

```
Boston        23
New York      17
Los Angeles   14
San Diego      7
Name: City, dtype: int64
```

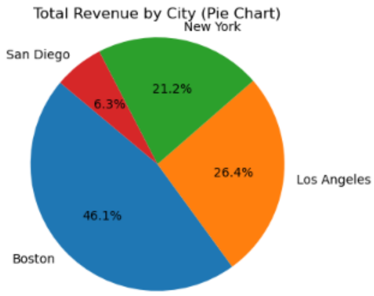



```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate the frequency distribution of the 'City' column
city_distribution = data['City'].value_counts()

# Create a pie chart
plt.figure(figsize=(4, 4))  # Optional: Set the figure size for the pie chart
plt.pie(city_distribution, labels=city_distribution.index, autopct='%1.1f%%', startangle=140)
plt.axis('equal')  # Equal aspect ratio ensures that the pie is circular.
plt.title('City Distribution (Pie Chart)')

# Display the pie chart
plt.show()

# Create a bar graph
plt.figure(figsize=(6, 4))  # Optional: Set the figure size for the bar graph
city_distribution.plot(kind='bar')
plt.xlabel('City')
plt.ylabel('Frequency')
plt.title('City Distribution (Bar Graph)')

# Rotate x-axis labels for better readability (optional)
plt.xticks(rotation=45)

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

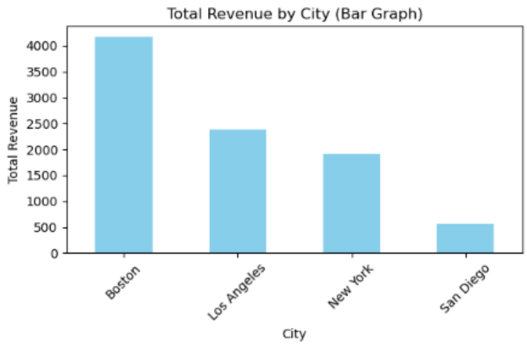



```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')  # Replace 'your_data.csv' with the path to your dataset

# Group data by 'City' and calculate the sum of 'TotalPrice' for each city
city_revenue = data.groupby('City')['TotalPrice'].sum()

# Display the total revenue for each city
print("Total Revenue from Different Cities:")
print(city_revenue)
```

```
Total Revenue from Different Cities:
City
Boston         4166.41
Los Angeles    2386.11
New York       1911.37
San Diego       569.97
Name: TotalPrice, dtype: float64
```




```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')  # Replace 'your_data.csv' with your dataset's path

# Group data by 'City' and calculate the sum of 'TotalPrice' for each city
city_revenue = data.groupby('City')['TotalPrice'].sum()

# Create a pie chart
plt.figure(figsize=(4, 4))
plt.pie(city_revenue, labels=city_revenue.index, autopct='%1.1f%%', startangle=140)
plt.title('Total Revenue by City (Pie Chart)')
plt.axis('equal')  # Equal aspect ratio ensures that the pie chart is circular
plt.show()

# Create a bar graph
plt.figure(figsize=(6, 4))
city_revenue.plot(kind='bar', color='skyblue')
plt.xlabel('City')
plt.ylabel('Total Revenue')
plt.title('Total Revenue by City (Bar Graph)')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```

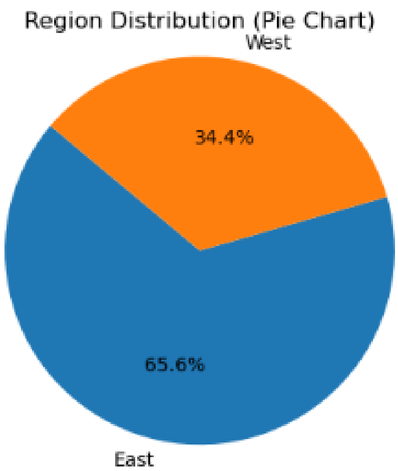


- Analysing the frequency distribution of cities, with Boston having the highest frequency and San Diego the lowest, reveals regional sales dynamics. Boston's high frequency suggests market dominance, effective marketing, and a strong customer base. Conversely, San Diego's lower frequency indicates untapped market potential or a smaller market share. This data guides resource allocation, regional strategies, and marketing efforts. It also highlights opportunities for expansion and market targeting in cities with lower frequencies. Additionally, it aids in competitive analysis, helping businesses adapt to regional preferences and market conditions.

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate the frequency distribution of the 'Region' column
region_distribution = data['Region'].value_counts()

# Print the frequency distribution
print(region_distribution)
```

```
East    40
West    21
Name: Region, dtype: int64
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate the frequency distribution of the 'Region' column
region_distribution = data['Region'].value_counts()

# Create a pie chart
plt.figure(figsize=(4, 4))  # Optional: Set the figure size for the pie chart
plt.pie(region_distribution, labels=region_distribution.index, autopct='%1.1f%%', startangle=140)
plt.axis('equal')  # Equal aspect ratio ensures that the pie is circular.
plt.title('Region Distribution (Pie Chart)')

# Display the pie chart
plt.show()
```



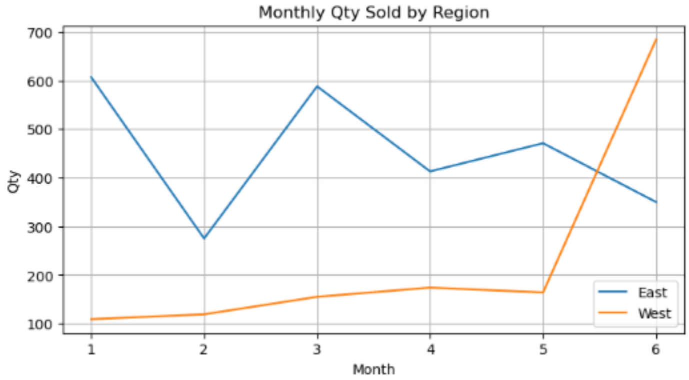
- ➢ The frequency distribution highlighting the East region's dominance over the West region, with almost double the percentage share, suggests significant regional disparities. The East likely boasts a larger customer base and higher demand, making it the primary market. In contrast, the West presents untapped market potential or a smaller market share, offering opportunities for expansion. Analysing regional preferences helps tailor marketing strategies. Resource allocation decisions can be guided by these findings, with more significant investments directed toward the East. This distribution underscores the importance of understanding regional dynamics for strategic business decisions, such as market targeting and resource allocation.

The following shows the monthly sales data based on regions and cities:

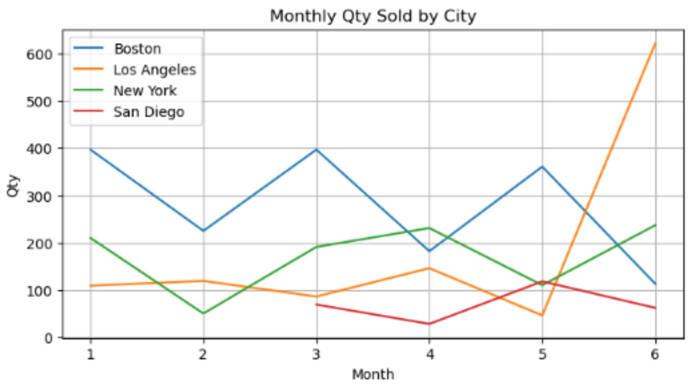
```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Convert the 'Date' column to a datetime object
data['Date'] = pd.to_datetime(data['Date'], format='%d-%b')

# Extract the month from the 'Date' column
data['Month'] = data['Date'].dt.month

# Group the data by 'Region', 'City', and 'Month' while summing the 'Qty'
monthly_qty = data.groupby(['Region', 'City', 'Month'])['Qty'].sum().reset_index()

# Print the result
print(monthly_qty)
```

```
   Region         City  Month  Qty
0    East       Boston      1  397
1    East       Boston      2  225
2    East       Boston      3  397
3    East       Boston      4  182
4    East       Boston      5  361
5    East       Boston      6  113
6    East     New York      1  210
7    East     New York      2   50
8    East     New York      3  191
9    East     New York      4  231
10   East     New York      5  110
11   East     New York      6  237
12   West  Los Angeles      1  109
13   West  Los Angeles      2  119
14   West  Los Angeles      3   86
15   West  Los Angeles      4  146
16   West  Los Angeles      5   46
17   West  Los Angeles      6  622
18   West    San Diego      3   69
19   West    San Diego      4   28
20   West    San Diego      5  118
21   West    San Diego      6   62
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Convert the 'Date' column to a datetime object
data['Date'] = pd.to_datetime(data['Date'], format='%d-%b')

# Extract the month from the 'Date' column
data['Month'] = data['Date'].dt.month

# Group the data by 'Region' and 'Month' while summing the 'Qty'
region_monthly_qty = data.groupby(['Region', 'Month'])['Qty'].sum().reset_index()

# Create a line chart for regions
plt.figure(figsize=(8, 4))  # Optional: Set the figure size
for region in region_monthly_qty['Region'].unique():
    region_data = region_monthly_qty[region_monthly_qty['Region'] == region]
    plt.plot(region_data['Month'], region_data['Qty'], label=region)

plt.xlabel('Month')
plt.ylabel('Qty')
plt.title('Monthly Qty Sold by Region')
plt.legend()
plt.grid(True)

# Display the line chart for regions
plt.show()
```




```python
# Group the data by 'City' and 'Month' while summing the 'Qty'
city_monthly_qty = data.groupby(['City', 'Month'])['Qty'].sum().reset_index()

# Create a line chart for cities
plt.figure(figsize=(8, 4))  # Optional: Set the figure size
for city in city_monthly_qty['City'].unique():
    city_data = city_monthly_qty[city_monthly_qty['City'] == city]
    plt.plot(city_data['Month'], city_data['Qty'], label=city)

plt.xlabel('Month')
plt.ylabel('Qty')
plt.title('Monthly Qty Sold by City')
plt.legend()
plt.grid(True)

# Display the line chart for cities
plt.show()
```

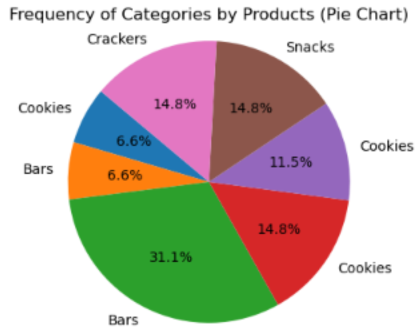


The following shows the frequency distribution of categories based on different products:

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Group the data by 'Product' and 'Category' and count occurrences
product_category_frequency = data.groupby(['Product', 'Category']).size().reset_index(name='Frequency')

# Print the frequency of categories based on products
print(product_category_frequency)
```

```
           Product  Category  Frequency
0        Arrowroot   Cookies          4
1             Bran      Bars          4
2           Carrot      Bars         19
3  Chocolate Chip   Cookies          9
4   Oatmeal Raisin   Cookies          7
5     Potato Chips    Snacks          9
6      Whole Wheat  Crackers          9
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Group the data by 'Product' and 'Category' and count occurrences
product_category_frequency = data.groupby(['Product', 'Category']).size().reset_index(name='Frequency')

# Create a pie chart
plt.figure(figsize=(4, 4))
plt.pie(product_category_frequency['Frequency'], labels=product_category_frequency['Category'], autopct='%1.1f%%', startangle=14
plt.axis('equal')
plt.title('Frequency of Categories by Products (Pie Chart)')
plt.show()
```



- ➢ The frequency distribution of categories based on products reveals varying levels of product popularity. "Bars - Carrot" and "Cookies - Chocolate Chip" enjoy higher frequencies, signifying their broad appeal and market dominance within their respective categories. These findings reflect customer preferences, indicating a preference for "Carrot" over "Bran" in the "Bars" category and "Chocolate Chip" over "Oatmeal Raisin" and "Arrowroot" in the "Cookies" category. Effective marketing, product quality, and pricing likely contribute to these preferences. This information guides inventory management and marketing strategies, emphasizing the importance of offering a diverse product range to cater to varying customer tastes and optimize sales.

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Find the row with the minimum unit price
min_unit_price_row = data[data['UnitPrice'] == data['UnitPrice'].min()]

# Find the row with the maximum unit price
max_unit_price_row = data[data['UnitPrice'] == data['UnitPrice'].max()]

# Extract category and product from the rows
min_category = min_unit_price_row['Category'].values[0]
min_product = min_unit_price_row['Product'].values[0]

max_category = max_unit_price_row['Category'].values[0]
max_product = max_unit_price_row['Product'].values[0]

# Print the results
print("Category and Product with Minimum Unit Price:")
print("Category:", min_category)
print("Product:", min_product)
print()

print("Category and Product with Maximum Unit Price:")
print("Category:", max_category)
print("Product:", max_product)
```

```
Category and Product with Minimum Unit Price:
Category: Snacks
Product: Potato Chips

Category and Product with Maximum Unit Price:
Category: Crackers
Product: Whole Wheat
```

- The minimum unit price is observed in the "Snacks" category, specifically for "Potato Chips," suggests a focus on affordability and may cater to price-sensitive customers. In contrast, the maximum unit price is observed in the "Crackers" category, specifically for "Whole Wheat," indicates a potential emphasis on quality or premium positioning. This variation implies segmentation in the customer base, with some seeking budget-friendly options and others willing to invest in perceived quality or health benefits. The pricing differences also impact profit margins and reflect the competitive landscape within these categories. Businesses can leverage this data to align their pricing strategies, product development, and market positioning accordingly.

```python
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Find the row with the minimum total price
min_total_price_row = data[data['TotalPrice'] == data['TotalPrice'].min()]

# Find the row with the maximum total price
max_total_price_row = data[data['TotalPrice'] == data['TotalPrice'].max()]

# Extract category, product, and total price from the rows
min_category = min_total_price_row['Category'].values[0]
min_product = min_total_price_row['Product'].values[0]
min_total_price = min_total_price_row['TotalPrice'].values[0]

max_category = max_total_price_row['Category'].values[0]
max_product = max_total_price_row['Product'].values[0]
max_total_price = max_total_price_row['TotalPrice'].values[0]

# Print the results
print("Category and Product with Minimum Total Price:")
print("Category:", min_category)
print("Product:", min_product)
print("Total Price:", min_total_price)
print()

print("Category and Product with Maximum Total Price:")
print("Category:", max_category)
print("Product:", max_product)
print("Total Price:", max_total_price)
```

```
Category and Product with Minimum Total Price:
Category: Bars
Product: Carrot
Total Price: 35.4

Category and Product with Maximum Total Price:
Category: Cookies
Product: Oatmeal Raisin
Total Price: 817.92
```

- This shows the minimum and the maximum transaction out of all the transactions.

```
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Convert the 'Date' column to a datetime object
data['Date'] = pd.to_datetime(data['Date'], format='%d-%b')

# Extract the month from the 'Date' column
data['Month'] = data['Date'].dt.month

# Group the data by 'Month' and calculate the mean Qty and mean TotalPrice
monthly_mean_sales = data.groupby('Month').agg({'Qty': 'mean', 'TotalPrice': 'mean'}).reset_index()

# Print the mean sales throughout months
print(monthly_mean_sales)
```

```
   Month         Qty  TotalPrice
0      1   65.090909  155.074545
1      2   43.777778  102.894444
2      3   74.300000  164.770000
3      4   58.700000  105.209000
4      5   57.727273  126.677273
5      6  103.400000  230.875000
```

- This shows the average(mean) quantity sold and total price of each month(Jan-June).

```
import pandas as pd

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Convert the 'Date' column to a datetime object
data['Date'] = pd.to_datetime(data['Date'], format='%d-%b')

# Extract the month from the 'Date' column
data['Month'] = data['Date'].dt.month

# Group the data by 'Month' and calculate the total Qty for each month
monthly_qty_total = data.groupby('Month')['Qty'].sum()

# Find the most successful and least successful months based on Qty
most_successful_month_qty = monthly_qty_total.idxmax()
least_successful_month_qty = monthly_qty_total.idxmin()

# Print the results
print("Most Successful Month based on Qty:", most_successful_month_qty)
print("Least Successful Month based on Qty:", least_successful_month_qty)
```

```
Most Successful Month based on Qty: 6
Least Successful Month based on Qty: 2
```

```
# Group the data by 'Month' and calculate the total TotalPrice for each month
monthly_total_price_total = data.groupby('Month')['TotalPrice'].sum()

# Find the most successful and least successful months based on TotalPrice
most_successful_month_total_price = monthly_total_price_total.idxmax()
least_successful_month_total_price = monthly_total_price_total.idxmin()

# Print the results
print("Most Successful Month based on Total Price:", most_successful_month_total_price)
print("Least Successful Month based on Total Price:", least_successful_month_total_price)
```

```
Most Successful Month based on Total Price: 6
Least Successful Month based on Total Price: 2
```

- This data clearly shows that June was the most successful month both in terms of quantity sold and total price, whereas as February was the least successful month.

The following data shows the most successful and least successful categories and products respectively, on the basis of Total Price:

```python
import pandas as pd
from tabulate import tabulate

# Load your data into a Pandas DataFrame
data = pd.read_csv('FoodSales.csv')

# Calculate the total TotalPrice for each category
category_total_price = data.groupby('Category')['TotalPrice'].sum()

# Find the most successful and least successful categories based on TotalPrice
most_successful_category = category_total_price.idxmax()
least_successful_category = category_total_price.idxmin()

# Calculate the total TotalPrice for each product
product_total_price = data.groupby('Product')['TotalPrice'].sum()

# Find the most successful and least successful products based on TotalPrice
most_successful_product = product_total_price.idxmax()
least_successful_product = product_total_price.idxmin()

# Filter the data for the most successful and least successful categories and products
most_successful_category_data = data[data['Category'] == most_successful_category]
least_successful_category_data = data[data['Category'] == least_successful_category]
most_successful_product_data = data[data['Product'] == most_successful_product]
least_successful_product_data = data[data['Product'] == least_successful_product]

# Display the data in tabular form
print("Most Successful Category based on TotalPrice:")
print(tabulate(most_successful_category_data, headers='keys', tablefmt='pretty'))
print("\nLeast Successful Category based on TotalPrice:")
print(tabulate(least_successful_category_data, headers='keys', tablefmt='pretty'))
print("\nMost Successful Product based on TotalPrice:")
print(tabulate(most_successful_product_data, headers='keys', tablefmt='pretty'))
print("\nLeast Successful Product based on TotalPrice:")
print(tabulate(least_successful_product_data, headers='keys', tablefmt='pretty'))
```

Most Successful Category based on TotalPrice:

	ID	Date	Region	City	Category	Product	Qty	UnitPrice	TotalPrice
2	ID07353	7-Jan	West	Los Angeles	Cookies	Chocolate Chip	58	1.87	108.46
3	ID07354	10-Jan	East	New York	Cookies	Chocolate Chip	82	1.87	153.34
4	ID07355	13-Jan	East	Boston	Cookies	Arrowroot	38	2.18	82.84
10	ID07361	31-Jan	East	Boston	Cookies	Arrowroot	36	2.18	78.48
11	ID07362	3-Feb	East	Boston	Cookies	Chocolate Chip	31	1.87	57.97
16	ID07367	18-Feb	East	Boston	Cookies	Arrowroot	43	2.18	93.74
17	ID07368	21-Feb	East	Boston	Cookies	Oatmeal Raisin	123	2.84	349.32
19	ID07370	27-Feb	West	Los Angeles	Cookies	Oatmeal Raisin	33	2.84	93.72
20	ID07371	2-Mar	East	New York	Cookies	Chocolate Chip	85	1.87	158.95
21	ID07372	5-Mar	West	San Diego	Cookies	Oatmeal Raisin	30	2.84	85.2
24	ID07375	14-Mar	West	Los Angeles	Cookies	Chocolate Chip	86	1.87	160.82
27	ID07378	23-Mar	West	San Diego	Cookies	Chocolate Chip	39	1.87	72.93
29	ID07380	29-Mar	East	Boston	Cookies	Oatmeal Raisin	193	2.84	548.12
40	ID07391	1-May	West	San Diego	Cookies	Chocolate Chip	63	1.87	117.81
42	ID07393	7-May	East	Boston	Cookies	Oatmeal Raisin	138	2.84	391.92
47	ID07398	22-May	West	San Diego	Cookies	Chocolate Chip	55	1.87	102.85
48	ID07399	25-May	East	Boston	Cookies	Arrowroot	27	2.18	58.86
51	ID07402	3-Jun	West	Los Angeles	Cookies	Oatmeal Raisin	288	2.84	817.92
52	ID07403	6-Jun	East	New York	Cookies	Chocolate Chip	76	1.87	142.12
60	ID07411	30-Jun	East	New York	Cookies	Oatmeal Raisin	51	2.84	144.84

Least Successful Category based on TotalPrice:

	ID	Date	Region	City	Category	Product	Qty	UnitPrice	TotalPrice
9	ID07360	28-Jan	East	New York	Snacks	Potato Chips	28	1.35	37.8
15	ID07366	15-Feb	East	New York	Snacks	Potato Chips	27	1.35	36.45
26	ID07377	20-Mar	East	New York	Snacks	Potato Chips	68	1.68	114.24
31	ID07382	4-Apr	West	Los Angeles	Snacks	Potato Chips	68	1.68	114.24
34	ID07385	13-Apr	West	San Diego	Snacks	Potato Chips	28	1.68	47.04
36	ID07387	19-Apr	East	Boston	Snacks	Potato Chips	134	1.68	225.12
39	ID07390	28-Apr	East	New York	Snacks	Potato Chips	64	1.68	107.52
46	ID07397	19-May	East	New York	Snacks	Potato Chips	49	1.68	82.32
58	ID07409	24-Jun	West	Los Angeles	Snacks	Potato Chips	28	1.68	47.04

Most Successful Product based on TotalPrice:

	ID	Date	Region	City	Category	Product	Qty	UnitPrice	TotalPrice
17	ID07368	21-Feb	East	Boston	Cookies	Oatmeal Raisin	123	2.84	349.32
19	ID07370	27-Feb	West	Los Angeles	Cookies	Oatmeal Raisin	33	2.84	93.72
21	ID07372	5-Mar	West	San Diego	Cookies	Oatmeal Raisin	30	2.84	85.2
29	ID07380	29-Mar	East	Boston	Cookies	Oatmeal Raisin	193	2.84	548.12
42	ID07393	7-May	East	Boston	Cookies	Oatmeal Raisin	138	2.84	391.92
51	ID07402	3-Jun	West	Los Angeles	Cookies	Oatmeal Raisin	288	2.84	817.92
60	ID07411	30-Jun	East	New York	Cookies	Oatmeal Raisin	51	2.84	144.84

Least Successful Product based on TotalPrice:

	ID	Date	Region	City	Category	Product	Qty	UnitPrice	TotalPrice
4	ID07355	13-Jan	East	Boston	Cookies	Arrowroot	38	2.18	82.84
10	ID07361	31-Jan	East	Boston	Cookies	Arrowroot	36	2.18	78.48
16	ID07367	18-Feb	East	Boston	Cookies	Arrowroot	43	2.18	93.74
48	ID07399	25-May	East	Boston	Cookies	Arrowroot	27	2.18	58.86

FINDINGS AND INFERENCES:

From the analysis of the dataset, several findings and inferences can be made:

1. Category and Product Insights:

- The "Snacks" category has the lowest minimum unit price, with the product "Potato Chips" having the minimum unit price in the dataset.
- The "Crackers" category has the highest maximum unit price, with the product "Whole Wheat" having the maximum unit price in the dataset.
- The "Bars" category has the lowest minimum total price, with the product "Carrot" having the minimum total price.
- The "Cookies" category has the highest maximum total price, with the product "Oatmeal Raisin" having the maximum total price.

2. Monthly Sales Insights:

- The most successful month based on quantity sold and total price is June (Month 6), indicating a peak in sales during that month.
- The least successful month based on quantity sold is February (Month 2), and based on total price is February (Month 2).

3. Category-Based Analysis:

- Among the categories, "Cookies" generate the highest total revenue, followed by "Bars" and "Crackers."
- "Snacks" have the lowest total revenue among the categories.

4. City-Based Analysis:

- "Boston" is the city with the highest total revenue, followed by "Los Angeles" and "New York," while "San Diego" has the lowest total revenue.

5. Region-Based Analysis:

- The "East" region generates significantly higher total revenue compared to the "West" region, with "East" having almost double the revenue of "West."

6. Product Frequency Distribution:

- "Carrot" and "Potato Chips" are the most frequently sold products in their respective categories.

7. Category Frequency Distribution:

- In the "Bars" category, "Carrot" and "Bran" are equally popular products.
- In the "Cookies" category, "Chocolate Chip" is the most popular product.
- In the "Snacks" category, "Potato Chips" is the most popular product.

These findings provide valuable insights into sales patterns, category and product performance, and regional variations, which can inform business strategies and decision-making.

MANAGERIAL INSIGHTS/IMPLICATIONS:

Based on the data analysis, here are some managerial insights and implications:

1. Category and Product Focus:

- Given that "Cookies" generate the highest total revenue, the management should consider allocating more resources and marketing efforts to promote and expand this category.
- "Snacks," while having lower total revenue, may still have potential for growth. Managers can explore strategies to increase sales in this category, such as introducing new snack products or marketing campaigns.

2. Seasonal Trends:

- The analysis shows that June is the most successful month in terms of sales. Management should analyse the reasons behind this peak and consider implementing seasonal promotions or discounts to capitalize on this trend.

3. City-Specific Strategies:

- "Boston" is the top-performing city in terms of total revenue. Managers should investigate why this city outperforms others and consider replicating successful strategies in other locations.

- "San Diego" has the lowest total revenue, indicating potential growth opportunities. Managers can explore ways to boost sales in this city through targeted marketing or promotions.

4. Regional Emphasis:

- The "East" region significantly outperforms the "West" region in terms of total revenue. Managers should focus on maximizing revenue in the "East" region, possibly by expanding distribution channels or increasing product offerings.

5. Product Selection and Inventory Management:

- The product "Potato Chips" in the "Snacks" category has the lowest unit price, which may suggest that it is a popular choice among price-sensitive customers. Management should ensure an adequate supply of this product to meet demand.

- On the other hand, "Crackers" with the highest unit price may cater to a more premium segment. Managers should carefully manage inventory to avoid overstocking.

6. Product Diversification:

- To reduce reliance on a single product, managers can explore product diversification strategies within categories. This can involve introducing new flavours or variations of existing products to attract a broader customer base.

7. Competitive Pricing:

- Managers should monitor and adjust pricing strategies to remain competitive within each category while maintaining profitability.

8. Marketing and Promotion:

- Data indicates that certain products within categories are more popular. Managers can tailor marketing and promotional efforts to highlight these products to customers.

9. Inventory and Stocking Policies:

- Inventory management plays a crucial role in maintaining profitability. Managers should optimize inventory levels to reduce carrying costs while ensuring products are readily available to meet customer demand.

10. Market Expansion:

- Considering the significant difference in total revenue between regions, managers may explore opportunities for market expansion, either by targeting new cities or regions or by evaluating the potential for online sales channels.

These insights and implications provide a foundation for strategic decision-making and can help the management team refine their business strategies to enhance overall performance and profitability.