In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df_pb = pd.read_csv('purchase_behaviour.csv')
df_pb.head()
```

Out[2]:

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER
0	1000	YOUNG SINGLES/COUPLES	Premium
1	1002	YOUNG SINGLES/COUPLES	Mainstream
2	1003	YOUNG FAMILIES	Budget
3	1004	OLDER SINGLES/COUPLES	Mainstream
4	1005	MIDAGE SINGLES/COUPLES	Mainstream

In [3]:
```
df_pb.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72637 entries, 0 to 72636
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   LYLTY_CARD_NBR    72637 non-null  int64 
 1   LIFESTAGE         72637 non-null  object
 2   PREMIUM_CUSTOMER  72637 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.7+ MB
```

In [4]:
```
df_pb.isnull().sum()
```

Out[4]:
```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

==========

In [5]:
```
df_td = pd.read_excel('transaction_data.xlsx')
df_td.head()
```

Out[5]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_S
0	43390	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	
1	43599	1	1307	348	66	CCs Nacho Cheese 175g	3	
2	43605	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	
3	43329	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	
4	43330	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	

In [6]:
```
df_td.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   DATE            264836 non-null  int64  
 1   STORE_NBR       264836 non-null  int64  
 2   LYLTY_CARD_NBR  264836 non-null  int64  
 3   TXN_ID          264836 non-null  int64  
 4   PROD_NBR        264836 non-null  int64  
 5   PROD_NAME       264836 non-null  object 
 6   PROD_QTY        264836 non-null  int64  
 7   TOT_SALES       264836 non-null  float64
dtypes: float64(1), int64(6), object(1)
memory usage: 16.2+ MB
```

In [7]:
```
df_td.isnull().sum()
```

Out[7]:
```
DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
dtype: int64
```

In [8]:
```
df_td=df_td[~df_td.duplicated()]
```

In [9]:
```
df_td.duplicated().sum()
```

Out[9]:
```
0
```

Date Column

In [10]:
```
df_td['DATE'] = pd.to_datetime(df_td["DATE"], origin='1899-12-30', unit='D')
```

In [11]:
```
df_td.isnull().sum()
```

Out[11]:
```
DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
dtype: int64
```

In [12]:
```
df_td['PROD_WT']=df_td.PROD_NAME.str.extract(r'(\d+)')
```

In [13]:
```
df_td["PROD_NAME"]=df_td.PROD_NAME.str[:-4]
```

In [14]:
```
df_td.PROD_NAME = df_td.PROD_NAME.str.replace('&','')
```

In [15]:
```
df_td.PROD_NAME = df_td.PROD_NAME.str.replace('  ',' ')
```

In [16]:
```
 df_td = df_td[~df_td.PROD_NAME.str.contains('Salsa')]
```

In [17]:
```
df_td.isnull().sum()
```

Out[17]:
```
DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
PROD_WT           0
dtype: int64
```

In [18]:
```
df_td.PROD_NAME[:50]
```

Out[18]:
```
0           Natural Chip    Compny SeaSalt
1                       CCs Nacho Cheese
2         Smiths Crinkle Cut Chips Chicken
3          Smiths Chip Thinly S/CreamOnion
4       Kettle Tortilla ChpsHnyJlpno Chili
6        Smiths Crinkle Chips Salt Vinegar
7             Grain Waves     Sweet Chilli
8       Doritos Corn Chip Mexican Jalapeno
9            Grain Waves Sour  CreamChives
10       Smiths Crinkle Chips Salt Vinegar
11         Kettle Sensations  Siracha Lime
12                      Twisties Cheese
13                WW Crinkle Cut   Chicken
14                 Thins Chips Light Tangy
15                             CCs Original
16                             Burger Rings
17           NCC Sour Cream   Garden Chives
18      Doritos Corn Chip Southern Chicken
19                    Cheezels Cheese Box
20               Smiths Crinkle   Original
21           NCC Sour Cream   Garden Chives
22       Infzns Crn Crnchers Tangy Gcamole
23            Kettle Sea Salt   And Vinegar
24      Kettle Tortilla ChpsHnyJlpno Chili
26         Smiths Chip Thinly Cut Original
27                          Kettle Original
28           Red Rock Deli Thai ChilliLime
29       Infzns Crn Crnchers Tangy Gcamole
30             Pringles Sthrn FriedChicken
31                  Pringles SweetSpcy BBQ
33           Thins Chips     Originl saltd
34          Red Rock Deli Sp  Salt Truffle
35         Smiths Thinly    Swt ChliS/Cream
36                           Kettle Chilli
37                        Doritos Mexicana
38                  Thins Chips Light Tangy
39      Smiths Crinkle Cut French OnionDip
40          Natural ChipCo   Hony Soy Chckn
41               Dorito Corn Chp   Supreme
42                         Twisties Chicken
43        Smiths Thinly Cut  Roast Chicken
45          Kettle Mozzarella  Basil Pesto
46       Infzns Crn Crnchers Tangy Gcamole
47     Infuzions Thai SweetChili PotatoMix
48               Smiths Crinkle   Original
49        Kettle Sensations  Camembert Fig
50         Smith Crinkle Cut  Mac N Cheese
51               Kettle Honey Soy  Chicken
52             Thins Chips Seasonedchicken
53         Smiths Crinkle Cut Salt Vinegar
Name: PROD_NAME, dtype: object
```

In [19]:
```
df_td.PROD_WT = df_td.PROD_WT.astype('int64')
```

In [20]:
```
df_td.head()
```

Out[20]:

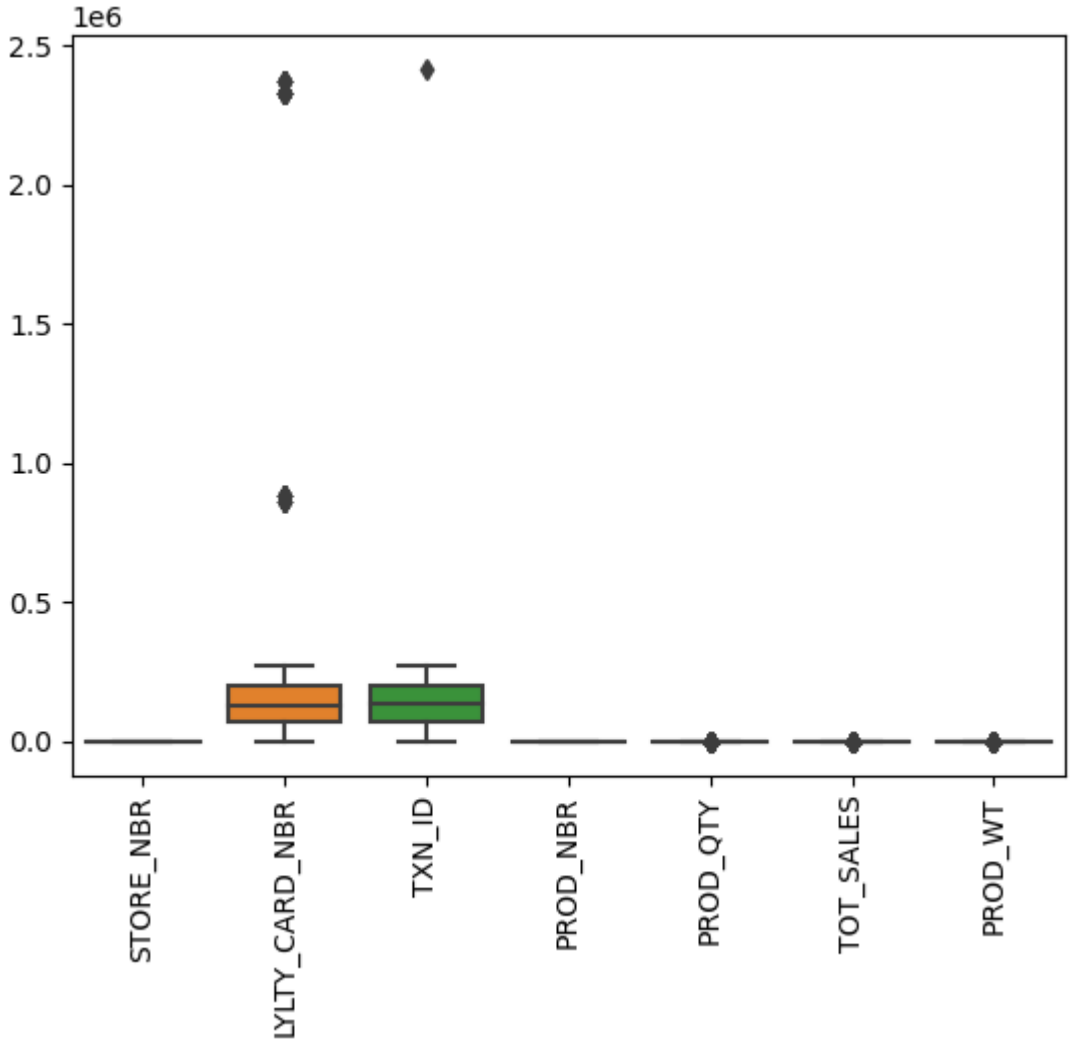
	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SAL
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt	2	
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese	3	
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken	2	
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/CreamOnion	5	1
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHnyJlpno Chili	3	1

In [21]:
```
df_td.describe()
```

Out[21]:

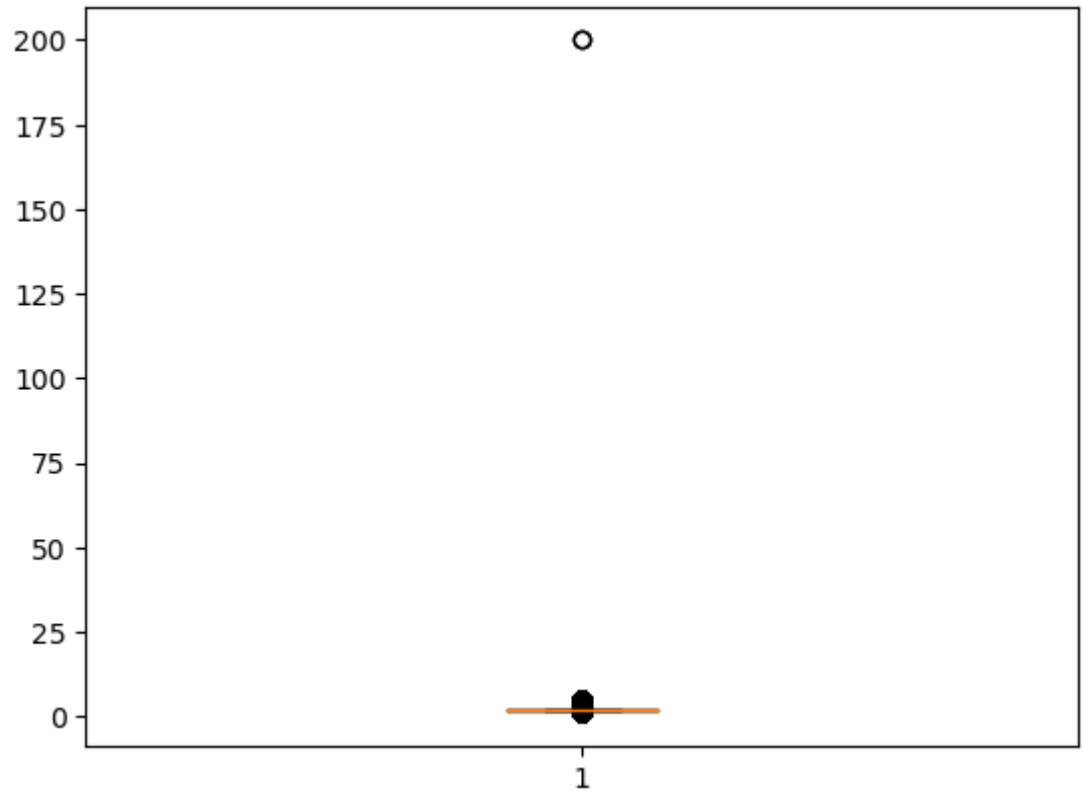
	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_QTY	TOT_SALE
count	246741.000000	2.467410e+05	2.467410e+05	246741.000000	246741.000000	246741.00000
mean	135.051212	1.355311e+05	1.351312e+05	56.351835	1.908061	7.32132
std	76.787231	8.071542e+04	7.814786e+04	33.695488	0.659832	3.07783
min	1.000000	1.000000e+03	1.000000e+00	1.000000	1.000000	1.70000
25%	70.000000	7.001500e+04	6.756900e+04	26.000000	2.000000	5.80000
50%	130.000000	1.303670e+05	1.351840e+05	53.000000	2.000000	7.40000
75%	203.000000	2.030840e+05	2.026540e+05	87.000000	2.000000	8.80000
max	272.000000	2.373711e+06	2.415841e+06	114.000000	200.000000	650.00000

In [22]:
```
sns.boxplot(data = df_td)
plt.xticks(rotation =90)
plt.show()
```

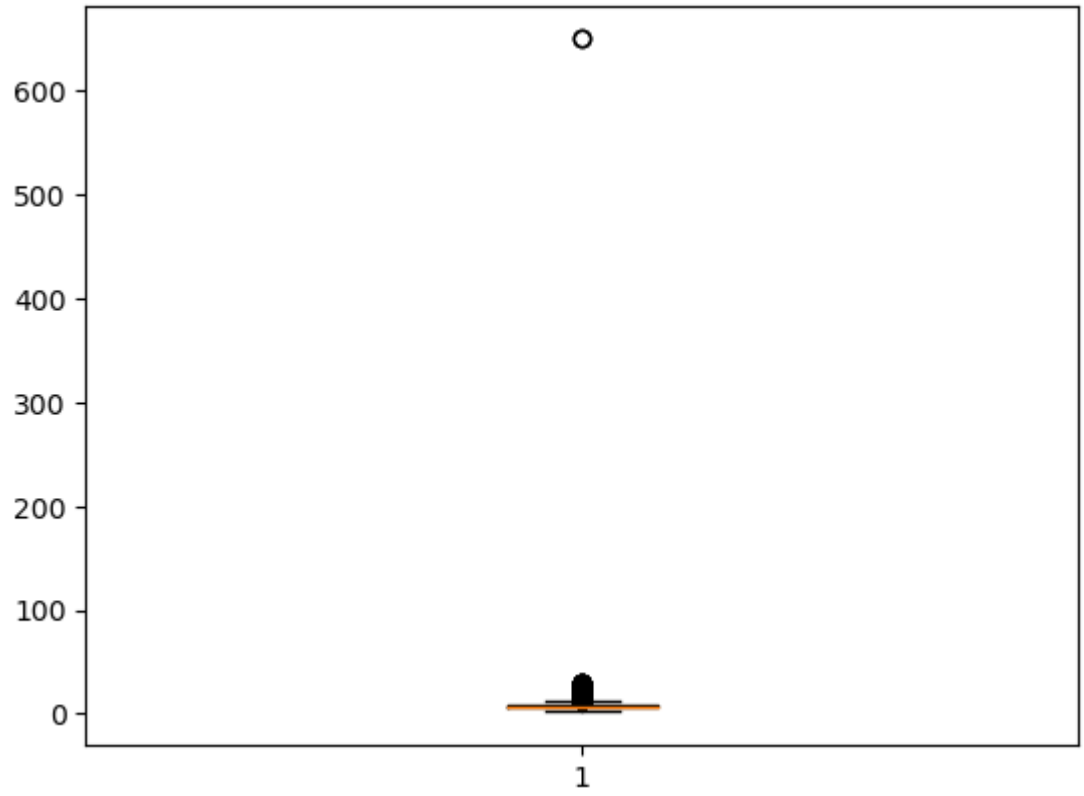


In [23]:
```
col = ["PROD_QTY", "TOT_SALES" , "PROD_WT"]
for i in col:
    print(i)
    plt.boxplot(df_td[i])
    plt.show()
```

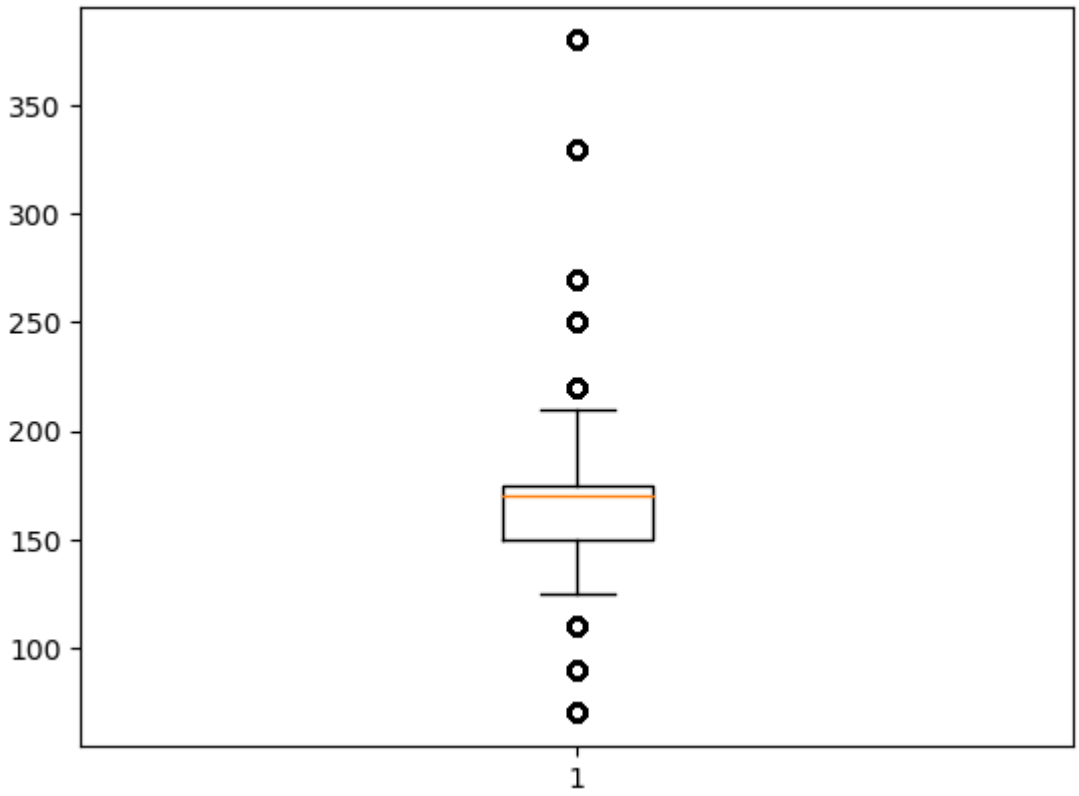
```
PROD_QTY
```



TOT_SALES



PROD_WT



In [24]:
```
col =["TOT_SALES","PROD_WT"]
for i in col:
    Q1 = df[i].quantile(.25)
    Q3 = df[i].quantile(.75)
    IQR = Q3 - Q1
    upper_limit = Q3 +(1.5*IQR)
    lower_limit = Q1 -(1.5*IQR)

    df[i] = df[i].apply(lambda x: upper_limit if x > upper_limit else (lower_limit


    print(i)
    plt.boxplot(df[i])
    plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_11308\1052694866.py in <module>
      1 col =["TOT_SALES","PROD_WT"]
      2 for i in col:
----> 3     Q1 = df[i].quantile(.25)
      4     Q3 = df[i].quantile(.75)
      5     IQR = Q3 - Q1

NameError: name 'df' is not defined
```

In [ ]:
```
df = pd.merge(df_td,df_pb, on="LYLTY_CARD_NBR", how='left')
```

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

Data Cleaning Completed

Exploratory Data Analysis

Demographic Analysis

In [ ]:
```
lifestage_count = df['LIFESTAGE'].value_counts()
lifestage_count.plot(kind='bar', color='skyblue')
plt.title('Distribution of Lifestages')
plt.xlabel('Lifestage')
plt.ylabel('Number of Customers')
plt.show()
```

Older Singles/couples purchased higher than other lifestyles followed by Retirees and purchasing of New Families is least

In [ ]:
```
premium_customer = df['PREMIUM_CUSTOMER'].value_counts()
premium_customer.plot(kind='bar', color='salmon')
plt.title("Distribution of Premium Customers")
plt.ylabel("Number of Customer")
plt.xlabel("Premium Customer Type")
plt.show()
```

Mainstream Premium customer are more than as compared to other followed by Budget and Premium customer are less as compared

In [ ]:
```

```

In [ ]:
```
df1 =df.copy()

df1['Segment'] = df['LIFESTAGE']+ "-" + df['PREMIUM_CUSTOMER']
plt.figure(figsize=(12,6))
segment_count = df1['Segment'].value_counts()
segment_count.plot(kind='bar', color='green')
plt.title('Customer Segmentation')
plt.xlabel("Segment")
plt.ylabel("Number of Customer")
plt.show()
```

Number of customer in OLDER Families-Budget segment are more than other segments followed by Retirees-Mainstream and New Families-Premium are less as compared followed by New-Families-Mainstream, New-Families-Budgets.

Product Analysis

In [ ]:
```
sales_product = df1.groupby('PROD_NAME')['TOT_SALES'].sum().sort_values(ascending=F
sales_product.plot(kind='bar')
plt.title("Top 10 Sales By Product")
plt.xlabel("Products")
plt.ylabel("Sales")
plt.show()
```

Older Singles/Couples generated the highest total sales followed by Ritrees

In [ ]:
```
quantity_by_product = df.groupby('PROD_NAME')['PROD_QTY'].sum().sort_values(ascendi
quantity_by_product.plot(kind='bar', color='salmon')
plt.title('Quantity Sold by Product')
plt.xlabel('Product')
plt.ylabel('Total Quantity Sold')
plt.show()
```

It has been observed that the highest quantity sold within the chip category is attributed to 'Dorito Corn Supreme.' Following closely in terms of sales quantity is the product 'Kettle Mozzarella Basil Pesto.'

In [ ]:
```
weight_by_product = df.groupby('PROD_NAME')['PROD_WT'].mean().sort_values(ascending
weight_by_product.plot(kind='bar', color='lightgreen')
plt.title('Average Weight of Products')
plt.xlabel('Product')
plt.ylabel('Average Weight (g)')
plt.show()
```

On average, Smiths Crinkle Chips Orignal Big Bang is the heaviest product followed by Dorito Corn Chips Supreme, suggesting that customers who prefer larger quantities tend to choose Smiths Crinkle Chips Orignal Big Bang .

Purchasing Behaviour

In [ ]:
```
date_df =df1.copy()
date_df['Month']=date_df['DATE'].dt.month_name()
month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'Aug
               'October', 'November', 'December']
```

In [ ]:
```
date_df['Month']= pd.Categorical(date_df['Month'], categories=month_order, ordered=
```

In [ ]:
```
monthly_sales =  date_df.groupby('Month')['TOT_SALES'].sum()
plt.figure(figsize=(10,5))
monthly_sales.plot(color ="skyblue", marker="o")
plt.ylabel("Total Sales")
plt.title("Monthly Sales")
plt.show()
```

Total sales peaked on December followed by March and July , indicating a significant month for transactions.

In [ ]:
```
monthly_QTY =  date_df.groupby('Month')['PROD_QTY'].sum()
plt.figure(figsize=(10,5))
monthly_QTY.plot(color ="red", marker="o")
plt.ylabel("Total Quantity")
plt.title("Monthly Quantity sold")
plt.show()
```

Quantity sold follows a similar trend to total sales, with a peak on December followed by March and July

In [ ]:
```
avg_monthly_sales =  date_df.groupby('Month')['TOT_SALES'].mean()
plt.figure(figsize=(10,5))
avg_monthly_sales.plot(color ="g", marker="o")
plt.ylabel("Average Sales")
plt.title("Total Average Monthly Sales")
plt.show()
```

Average transaction size remained relatively consistent Except in the month of may and August.

In [ ]:
```
Sales_by_Customer = df.groupby('PREMIUM_CUSTOMER')['TOT_SALES'].sum().sort_values(a
Sales_by_Customer.plot(kind= 'bar')
plt.title("Sales By Premium Customer")
plt.xlabel("Customer Type")
plt.ylabel("Total Sales")
plt.show()
```

Mainstream Customer generated the highest sales followed by Bugdget

In [ ]:
```
lifestage = df.groupby('LIFESTAGE')['TOT_SALES'].sum().sort_values(ascending = Fals
lifestage.plot(kind= 'bar')
plt.title(" Sales By Lifestage")
```

```
plt.xlabel("Lifestage")
plt.ylabel("Total Sales")
plt.show()
```

Doritto Corn chips Supreme generated the highest total sales followed by Smiths Crinkle Chips Big Bag And Smiths Crinkle Chips Salt Vinegar, outperforming other products. This suggests a strong demand for Chips A in our customer base.

Correlation Analysis

In [ ]:
```
correlation_matrix=df[['PROD_QTY', 'PROD_WT', 'TOT_SALES']].corr()
sns.heatmap(correlation_matrix,annot=True, cbar=False)
plt.show()
```

1. PROD_QTY and TOT_SALES: - Correlation Coefficient: 0.73 - Insight: There is a strong positive correlation (0.73) between the quantity of products sold (PROD_QTY) and total sales (TOT_SALES). This suggests that as the quantity of products sold increases, total sales also tend to increase proportionally. 2. PROD_QTY and PROD_WT: - Correlation Coefficient: 0.01 - Insight: There is a weak correlation (0.01) between the quantity of products sold (PROD_QTY) and the weight of the products (PROD_WT). This implies that there is a slight tendency for the quantity of products sold to decrease as the weight of the products increases, although the correlation is not very strong. 3. PROD_WT and TOT_SALES: - Correlation Coefficient: 0.35 - Insight:There is a strong positive correlation (0.35) between the weight of the products (PROD_WT) and total sales (TOT_SALES). This indicates that heavier products are associated with higher total sales, suggesting that customers might prefer or be willing to pay more for larger or heavier items.

Insight:

Demographic Analysis: 1. Older Singles/Couples Purchases: - Insight: Older Singles/Couples have the highest purchasing activity, followed by Retirees. New Families show the least engagement in chip purchases. - Implication: Tailor marketing strategies to cater more to Older Singles/Couples and Retirees, while exploring ways to attract New Families. 2. Customer Segmentation: - Insight: Mainstream Premium customers are more prevalent, followed by Budget customers. Premium customers are the least common. - Implication: Focus marketing efforts on Mainstream Premium customers, but also consider strategies to attract Budget customers. Understand the preferences and behaviors of Premium customers for targeted campaigns. 3. Segment-Specific Purchases: - Insight: OLDER Families in the Budget segment have the highest number of customers, while New Families in the Premium segment have the least. - Implication:Prioritize product offerings and promotions for OLDER Families in the Budget segment. Explore ways to increase engagement with New Families in the Premium segment. Product Analysis: 1. Top-Performing Products: - Insight: 'Dorito Corn Supreme' and 'Kettle Mozzarella Basil Pesto' are the top-performing products in terms of sales quantity. 'Dorito Corn Supreme' leads in total sales. - Implication: Consider promoting these popular products further and analyze customer preferences to optimize the product portfolio. 2. Weight Considerations: - Insight: Smiths Crinkle Chips Original Big Bang is the heaviest product, and Dorito Corn Chips Supreme is also significant in weight. - Implication:Recognize the preference for heavier products, and leverage this insight for targeted marketing or bundling strategies.Purchasing Behavior: 1. Monthly Peaks: - Insight: Total sales and quantity sold peak in December, March, and July. - Implication:Align inventory, marketing, and staffing resources to meet increased demand during these peak months. 2. Average Transaction Size: - Insight: Average transaction size remains consistent, except for May and August. - Implication: Investigate factors contributing to the fluctuations in May and August. Consider promotions or incentives to maintain a consistent transaction size. 3. Product Contribution to Sales: - Insight:'Dorito Corn Chips Supreme' is a strong contributor to total sales, outperforming other products. - Implication: Strategize promotions or marketing campaigns around 'Dorito Corn Chips Supreme' to capitalize on its popularity.