In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df_transaction = pd.read_csv("C:/Users/sona3/Downloads/QVI_transaction_data (1).csv")
```

In [3]:

```
df_purchase = pd.read_csv("C:/Users/sona3/Downloads/QVI_purchase_behaviour (1).csv")
```

In [4]:

```
df_transaction.head(10)
```

Out[4]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALES
0	43390	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.0
1	43599	1	1307	348	66	CCs Nacho Cheese 175g	3	6.3
2	43605	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.9
3	43329	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	15.0
4	43330	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	13.8
5	43604	4	4074	2982	57	Old El Paso Salsa Dip Tomato Mild 300g	1	5.1
6	43601	4	4149	3333	16	Smiths Crinkle Chips Salt & Vinegar 330g	1	5.7
7	43601	4	4196	3539	24	Grain Waves Sweet Chilli 210g	1	3.6
8	43332	5	5026	4525	42	Doritos Corn Chip Mexican Jalapeno 150g	1	3.9
9	43330	7	7150	6900	52	Grain Waves Sour Cream&Chives 210G	2	7.2

In [5]:

```
df_transaction.shape
```

Out[5]:

```
(264836, 8)
```

Data Summary

In [6]:

```
df_transaction.dtypes
```

Out[6]:

```
DATE                int64
STORE_NBR           int64
LYLTY_CARD_NBR      int64
TXN_ID              int64
PROD_NBR            int64
PROD_NAME          object
PROD_QTY            int64
TOT_SALES         float64
dtype: object
```

In [7]:

```
df_transaction.describe()
```

Out[7]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_QTY	TOT_SALES
count	264836.000000	264836.00000	2.648360e+05	2.648360e+05	264836.000000	264836.000000	264836.000000
mean	43464.036260	135.08011	1.355495e+05	1.351583e+05	56.583157	1.907309	7.304200
std	105.389282	76.78418	8.057998e+04	7.813303e+04	32.826638	0.643654	3.083226
min	43282.000000	1.00000	1.000000e+03	1.000000e+00	1.000000	1.000000	1.500000
25%	43373.000000	70.00000	7.002100e+04	6.760150e+04	28.000000	2.000000	5.400000
50%	43464.000000	130.00000	1.303575e+05	1.351375e+05	56.000000	2.000000	7.400000
75%	43555.000000	203.00000	2.030942e+05	2.027012e+05	85.000000	2.000000	9.200000
max	43646.000000	272.00000	2.373711e+06	2.415841e+06	114.000000	200.000000	650.000000

In [8]:

```
df_transaction.isnull().sum()
```

Out[8]:

```
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 [9]:

```
df_purchase.head(10)
```

Out[9]:

	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
5	1007	YOUNG SINGLES/COUPLES	Budget
6	1009	NEW FAMILIES	Premium
7	1010	YOUNG SINGLES/COUPLES	Mainstream
8	1011	OLDER SINGLES/COUPLES	Mainstream
9	1012	OLDER FAMILIES	Mainstream

In [10]:

```
df_purchase.shape
```

Out[10]:

```
(72637, 3)
```

In [11]:

```
df_purchase.dtypes
```

Out[11]:

```
LYLTY_CARD_NBR       int64
LIFESTAGE           object
PREMIUM_CUSTOMER    object
dtype: object
```

In [12]:

```
df_purchase.describe()
```

Out[12]:

	LYLTY_CARD_NBR
count	7.263700e+04
mean	1.361859e+05
std	8.989293e+04
min	1.000000e+03
25%	6.620200e+04
50%	1.340400e+05
75%	2.033750e+05
max	2.373711e+06

In [13]:

```
df_purchase.isnull().sum()
```

Out[13]:

```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

Changing Date Format

In [14]:

```
df_transaction['DATE'] = pd.to_datetime(df_transaction['DATE'], unit='d', origin='1899-12-30')
```

In [15]:

```
df_transaction.head()
```

Out[15]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALES
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.0
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	3	6.3
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.9
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	15.0
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	13.8

Extraction

In [16]:

```
df_transaction['PACKET_SIZE'] = df_transaction['PROD_NAME'].str.extract(r'(\d+)\s*[gG]?')
```

In [17]:

```
df_transaction.head()
```

Out[17]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALES	PACKET_SIZE
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.0	175
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	3	6.3	175
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.9	170
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	15.0	175
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	13.8	150

In [18]:

```
df_transaction['BRAND'] = df_transaction['PROD_NAME'].str.extract(r'^(\w+)')
```

In [19]:

```
df_transaction.head()
```

Out[19]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALES	PACKET_SIZE	BRAND
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.0	175	Natural
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	3	6.3	175	CCs
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.9	170	Smiths
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	15.0	175	Smiths
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	13.8	150	Kettle

Merging

In [75]:

```
merged_df = pd.merge(df_transaction, df_purchase, on='LYLTY_CARD_NBR', how='inner')
```

In [76]:

```
merged_df.head()
```

Out[76]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALES	PACKET_SIZE	BRAND	LIFESTAGE	PREMIUM_CUSTOMER
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.0	175	Natural	YOUNG SINGLES/COUPLES	Premium
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	3	6.3	175	CCs	MIDAGE SINGLES/COUPLES	Budget
2	2018-11-10	1	1307	346	96	WW Original Stacked Chips 160g	2	3.8	160	WW	MIDAGE SINGLES/COUPLES	Budget
3	2019-03-09	1	1307	347	54	CCs Original 175g	1	2.1	175	CCs	MIDAGE SINGLES/COUPLES	Budget
4	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.9	170	Smiths	MIDAGE SINGLES/COUPLES	Budget

Outliers

In [77]:

```
Q1 = merged_df['TOT_SALES'].quantile(0.25)
Q3 = merged_df['TOT_SALES'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR


sales_outliers = merged_df[(merged_df['TOT_SALES'] < lower_bound) | (merged_df['TOT_SALES'] > upper_bound)]

# Output results
print(f"Total outliers in 'TOT_SALES': {len(sales_outliers)}")
print(sales_outliers[['TOT_SALES']])
```

```
Total outliers in 'TOT_SALES': 578
        TOT_SALES
5            15.0
37           23.0
225          15.5
315          28.5
458          19.0
...           ...
259116       21.6
259138       18.4
259156       16.5
259170       15.0
259414       18.5

[578 rows x 1 columns]
```

In [78]:

```
df_cleaned = merged_df[(merged_df['TOT_SALES'] >= lower_bound) & (merged_df['TOT_SALES'] <= upper_bound)]
```

In [79]:

```
df_cleaned = merged_df
```

Business Requirements

In [80]:

```
# Total Sales

Total_sales = merged_df['TOT_SALES'].sum()
print(f"Total Sales : {Total_sales :,.0f} ")
```

```
Total Sales : 1,934,415
```

In [81]:

```
# Total number of customers

Total_customers = merged_df['LYLTY_CARD_NBR'].nunique()
print(f"Total Customers : {Total_customers :,.0f} ")
```

```
Total Customers : 72,637
```

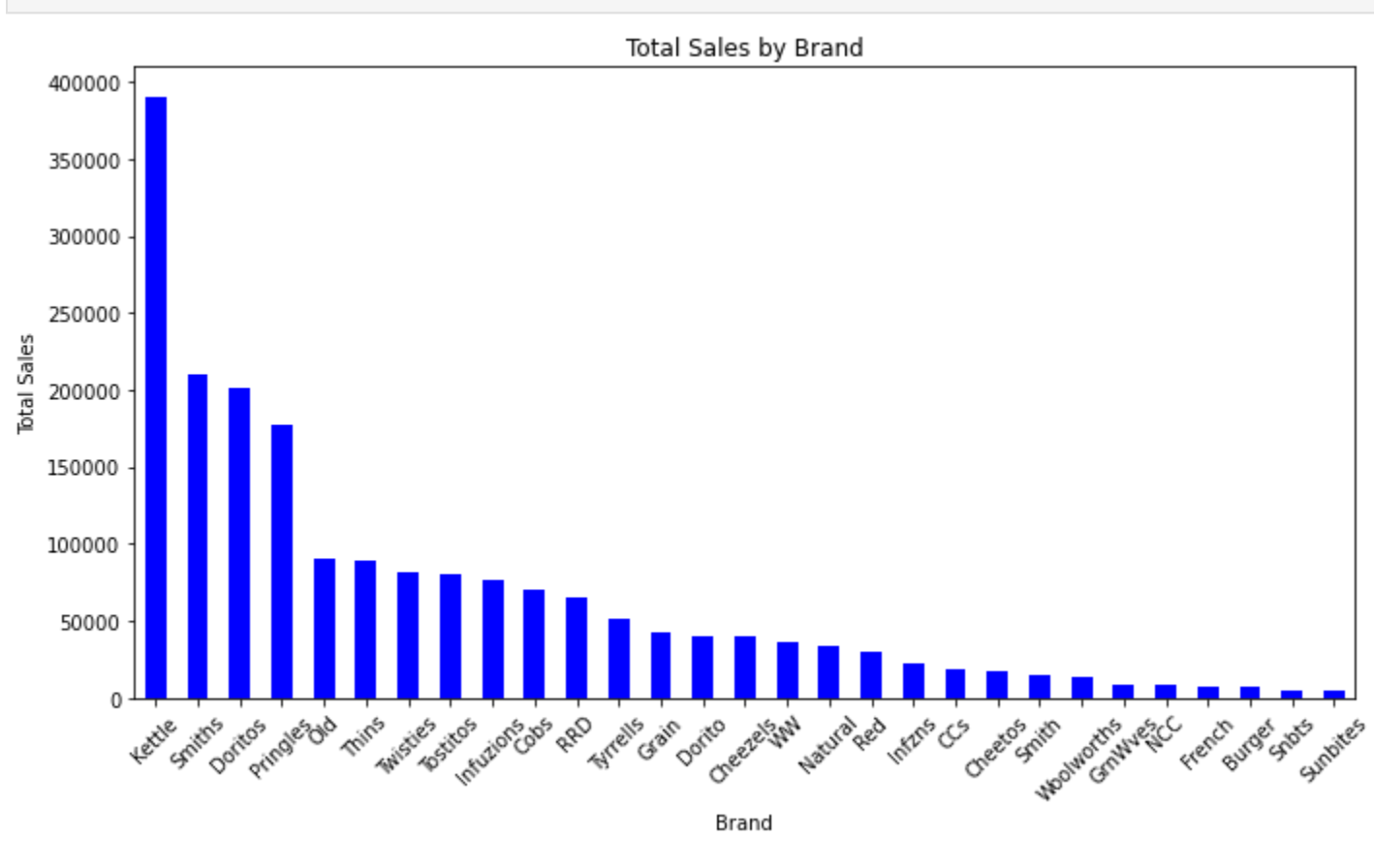
In [82]:

```
# average number of transactions per customer

avg_transactions = merged_df.shape[0] / merged_df['LYLTY_CARD_NBR'].nunique()
print(f"Average transactions per customer: {avg_transactions:.0f}")
```

```
Average transactions per customer: 4
```

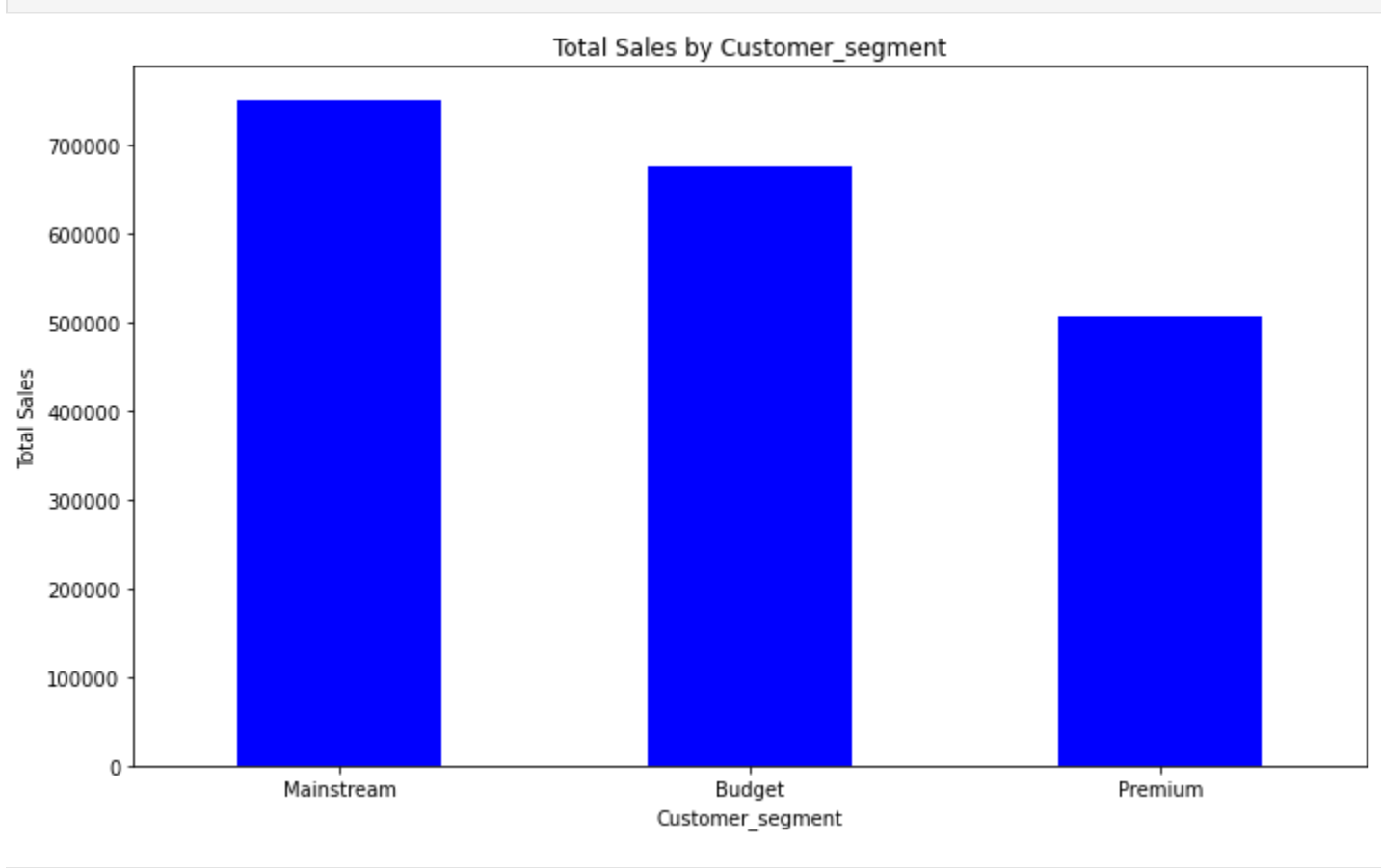
In [83]:

```
# Which brand generates highest sale

brand_sales = merged_df.groupby('BRAND')['TOT_SALES'].sum().sort_values(ascending=False)

# Plotting bar graph
plt.figure(figsize=(10, 6))
brand_sales.plot(kind='bar', color='blue')
plt.title('Total Sales by Brand')
plt.xlabel('Brand')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



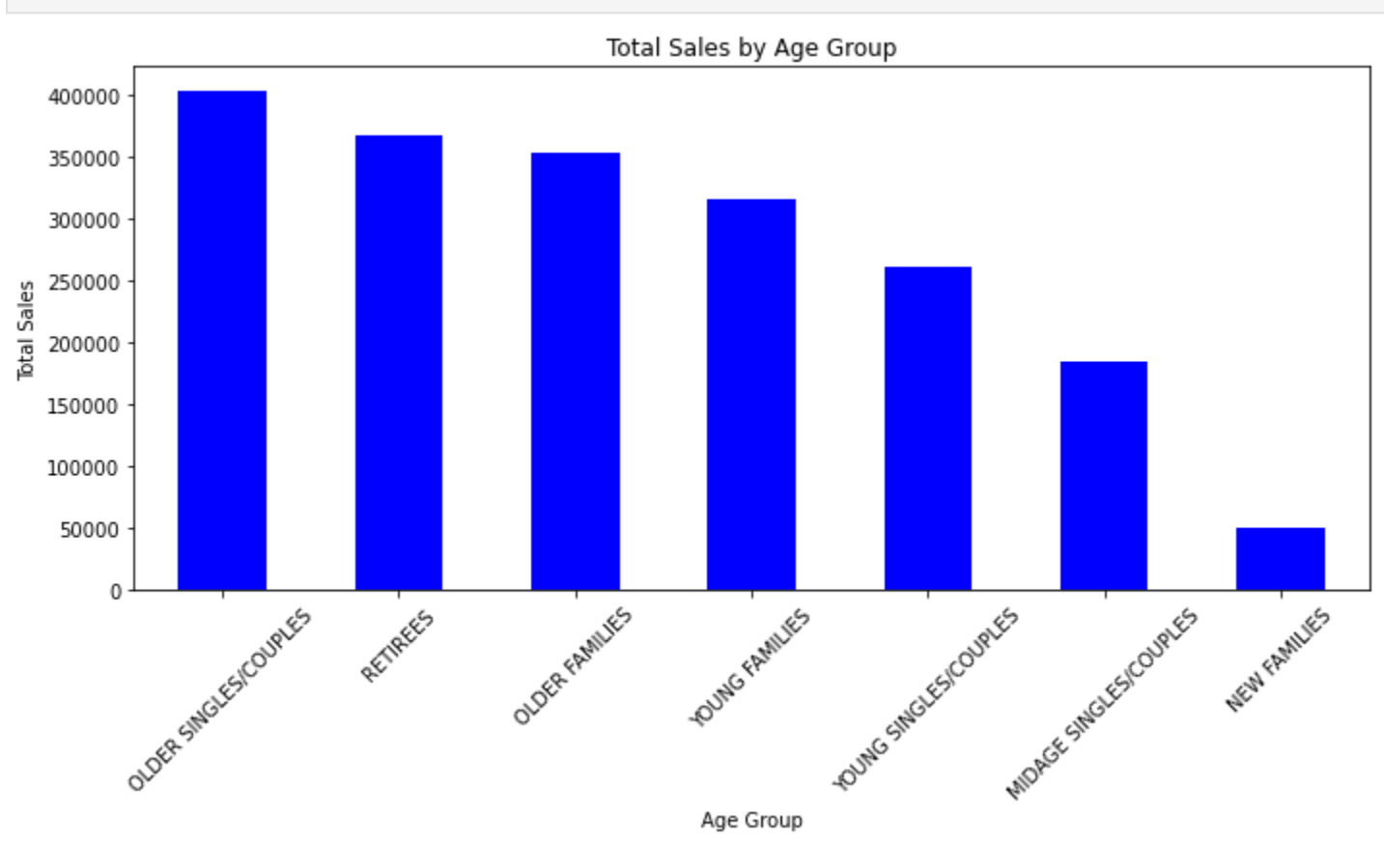
In [84]:

```
# Which customer segment should target

customer_segment = merged_df.groupby('PREMIUM_CUSTOMER')['TOT_SALES'].sum().sort_values(ascending=False)

# Plotting bar graph
plt.figure(figsize=(10, 6))
customer_segment.plot(kind='bar', color='blue')
plt.title('Total Sales by Customer_segment')
plt.xlabel('Customer_segment')
plt.ylabel('Total Sales')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



In [85]:

```
# Which age group should target

age_group = merged_df.groupby('LIFESTAGE')['TOT_SALES'].sum().sort_values(ascending=False)

# Plotting bar graph
plt.figure(figsize=(10, 6))
age_group.plot(kind='bar', color='blue')
plt.title('Total Sales by Age Group')
plt.xlabel('Age Group')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [86]:

```
merged_df['PACKET_SIZE'] = pd.to_numeric(merged_df['PACKET_SIZE'], errors='coerce')
```

In [87]:

```
merged_df.dtypes
```

Out[87]:

```
DATE                datetime64[ns]
STORE_NBR                    int64
LYLTY_CARD_NBR               int64
TXN_ID                       int64
PROD_NBR                     int64
PROD_NAME                   object
PROD_QTY                     int64
TOT_SALES                  float64
PACKET_SIZE                  int64
BRAND                       object
LIFESTAGE                   object
PREMIUM_CUSTOMER            object
dtype: object
```

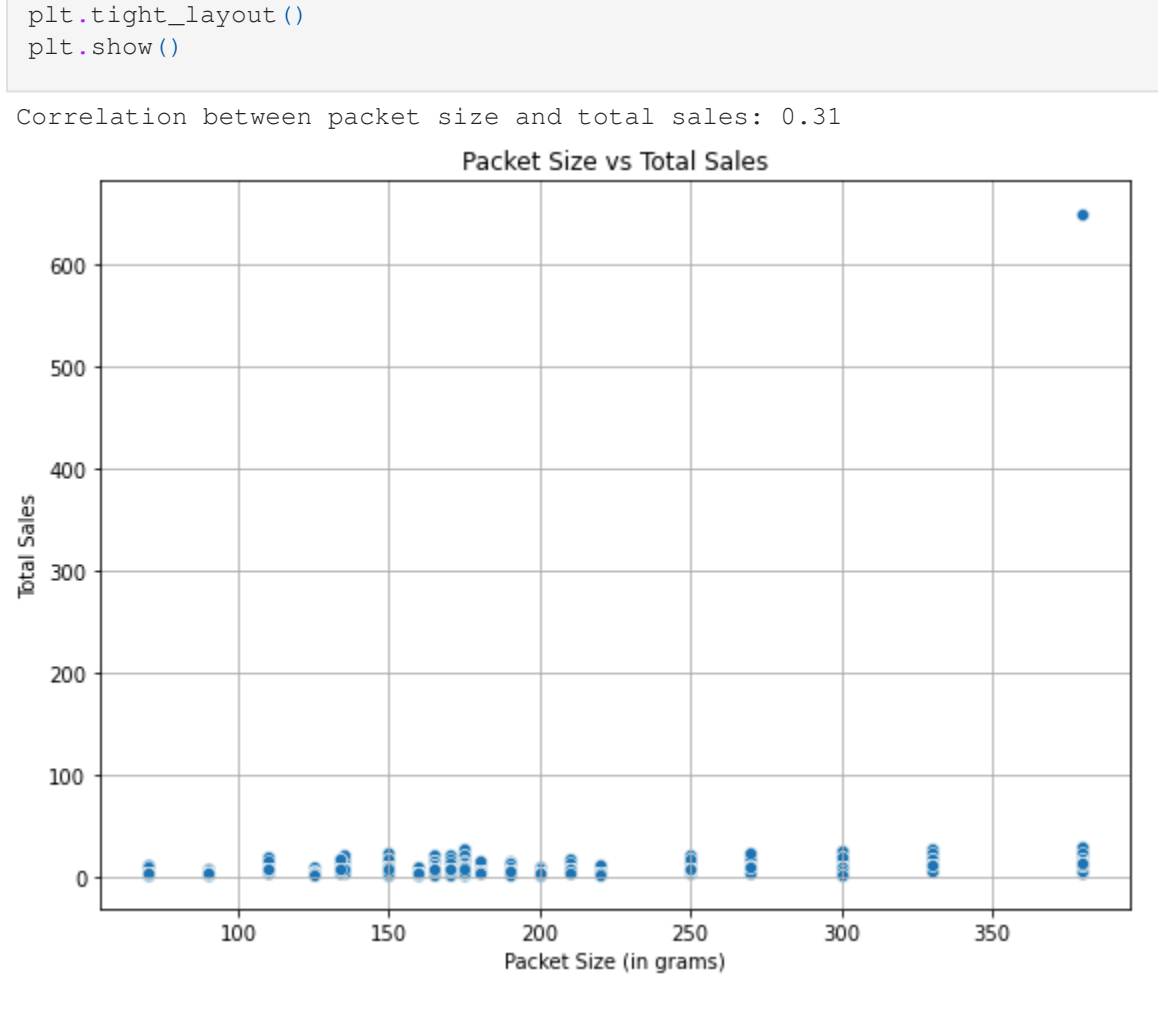
In [88]:

```
# Are there any relation between Total Sales and Packet size ?
```

In [89]:

```
correlation = merged_df['PACKET_SIZE'].corr(merged_df['TOT_SALES'])
print(f"Correlation between packet size and total sales: {correlation:.2f}")

# Step 4: Visualize
plt.figure(figsize=(8, 6))
sns.scatterplot(data=merged_df, x='PACKET_SIZE', y='TOT_SALES')
plt.title('Packet Size vs Total Sales')
plt.xlabel('Packet Size (in grams)')
plt.ylabel('Total Sales')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
Correlation between packet size and total sales: 0.31
```



In [93]:

```
#Which Product sizes are most popular overall

# Convert PACKET_SIZE to string to handle it as a category
merged_df['PACKET_SIZE'] = merged_df['PACKET_SIZE'].astype(str)

plt.figure(figsize=(10, 6))
sns.countplot(data=merged_df, x='PACKET_SIZE', order=merged_df['PACKET_SIZE'].value_counts().index[:10], color = 'blue')
plt.title('Top 10 Most Popular Pack Sizes')
plt.xlabel('Pack Size')
plt.ylabel('Number of Transactions')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

In [91]:

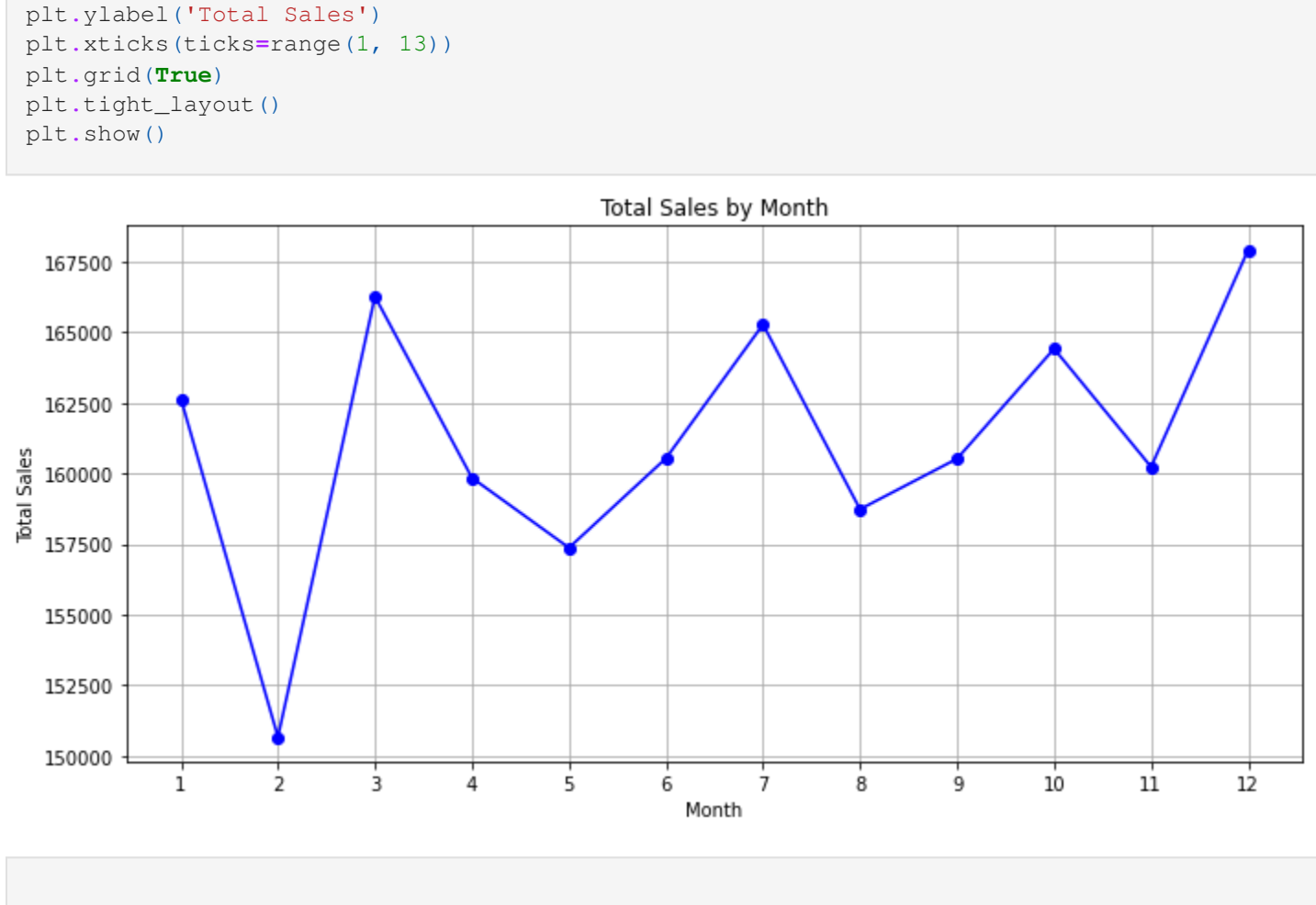
```
# Which store generates the most sales ?

merged_df.groupby('STORE_NBR')['TOT_SALES'].sum().sort_values(ascending=False).head(1)
```

Out[91]:

```
STORE_NBR
226    18905.45
Name: TOT_SALES, dtype: float64
```

In [92]:

```
# Total Sales by Month

# Extract month from date
merged_df['MONTH'] = merged_df['DATE'].dt.month

# Group sales by month
monthly_sales = merged_df.groupby('MONTH')['TOT_SALES'].sum().reset_index()

# Create a DataFrame with all 12 months
all_months = pd.DataFrame({'MONTH': range(1, 13)})

# Merge and fill missing months with 0
monthly_sales_full = pd.merge(all_months, monthly_sales, on='MONTH', how='left').fillna(0)

# Plot
plt.figure(figsize=(10, 5))
plt.plot(monthly_sales_full['MONTH'], monthly_sales_full['TOT_SALES'], marker='o', color='blue', linestyle='-')
plt.title('Total Sales by Month')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.xticks(ticks=range(1, 13))
plt.grid(True)
plt.tight_layout()
plt.show()
```



In [ ]: