In [146…
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
```

Data Validation and Cleaning

- Find outliers and remove
- Reformat data when applicable

Data

- Loyalty Card Number: Identification method, likely used for merging data
- Lifestate: Can be used to identify spending patterns
- Premium Customer: Way to identify wealth, could be combined wiht lifestage (renamed to make clearer)

In [147…
```
pd.set_option('display.max_columns', None)
customer_info = pd.read_csv("Dataset/QVI_purchase_behaviour.csv")
customer_info
```

Out[147…

	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
...	...	...	...
72632	2370651	MIDAGE SINGLES/COUPLES	Mainstream
72633	2370701	YOUNG FAMILIES	Mainstream
72634	2370751	YOUNG FAMILIES	Premium
72635	2370961	OLDER FAMILIES	Budget
72636	2373711	YOUNG SINGLES/COUPLES	Mainstream

72637 rows × 3 columns

In [148…
```
customer_info.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 [149…

```
print(f' Customer Status: {customer_info['PREMIUM_CUSTOMER'].unique()}')
print(f' Lifestages: {customer_info['LIFESTAGE'].unique()}')
```

```
Customer Status: ['Premium' 'Mainstream' 'Budget']
Lifestages: ['YOUNG SINGLES/COUPLES' 'YOUNG FAMILIES' 'OLDER SINGLES/COUPLES'
'MIDAGE SINGLES/COUPLES' 'NEW FAMILIES' 'OLDER FAMILIES' 'RETIREES']
```

Data

- Date: Needs to change type, but can be used for sales over time or which items are popular during certain times
- Store_Number: Identification for store, important when evaluating store profit
- Loyalty Card Number: Used for merging
- TXN_ID: Transaction ID
- Product Number/Product Name: For item identification
- Product quantity: Quantity of sale
- Tot Sales: Total Profit

Missing Data

- Product_Price: Calculate average price of product
- Product_Weight: Extract value from product_name

In [150…

```
transaction_df = pd.read_excel("Dataset/QVI_transaction_data.xlsx")
transaction_df
```

Out[150…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD
0	43390	1	1000	1	5	Natural Chip Compny SeaSalt175g	
1	43599	1	1307	348	66	CCs Nacho Cheese 175g	
2	43605	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	
3	43329	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	
4	43330	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	
...	...	...	...	...	...	...	
264831	43533	272	272319	270088	89	Kettle Sweet Chilli And Sour Cream 175g	
264832	43325	272	272358	270154	74	Tostitos Splash Of Lime 175g	
264833	43410	272	272379	270187	51	Doritos Mexicana 170g	
264834	43461	272	272379	270188	42	Doritos Corn Chip Mexican Jalapeno 150g	
264835	43365	272	272380	270189	74	Tostitos Splash Of Lime 175g	

264836 rows × 8 columns

In [151…

```
transaction_df.describe()
```

Out[151…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PR
count	264836.000000	264836.00000	2.648360e+05	2.648360e+05	264836.000000	26483
mean	43464.036260	135.08011	1.355495e+05	1.351583e+05	56.583157	
std	105.389282	76.78418	8.057998e+04	7.813303e+04	32.826638	
min	43282.000000	1.00000	1.000000e+03	1.000000e+00	1.000000	
25%	43373.000000	70.00000	7.002100e+04	6.760150e+04	28.000000	
50%	43464.000000	130.00000	1.303575e+05	1.351375e+05	56.000000	
75%	43555.000000	203.00000	2.030942e+05	2.027012e+05	85.000000	
max	43646.000000	272.00000	2.373711e+06	2.415841e+06	114.000000	20

In [152…

```
transaction_df.sort_values(by="PROD_QTY", ascending=False).head(10)
```

Out[152…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_C
69762	43331	226	226000	226201	4	Dorito Corn Chp Supreme 380g	
69763	43605	226	226000	226210	4	Dorito Corn Chp Supreme 380g	
217237	43603	201	201060	200202	26	Pringles Sweet&Spcy BBQ 134g	
238333	43326	219	219004	218018	25	Pringles SourCream Onion 134g	
238471	43604	261	261331	261111	87	Infuzions BBQ Rib Prawn Crackers 110g	
228749	43604	232	232138	235978	109	Pringles Barbeque 134g	
117802	43604	176	176471	177469	17	Kettle Sensations BBQ&Maple 150g	
228711	43329	205	205149	204215	1	Smiths Crinkle Cut Chips Barbecue 170g	
238397	43603	238	238337	243243	28	Thins Potato Chips Hot & Spicy 175g	
238395	43604	238	238250	242874	88	Kettle Honey Soy Chicken 175g	

The Two orders of Dorito Corn Chip seems to be outliers due to the Production Quantity and Total Sale being much larger than intended. These two orders are removed to eliminate bias.

In [153…
```
transaction_df.drop(index=[69762,69763], inplace=True)
```

In [154…
```
transaction_df.sort_values(by="PROD_QTY", ascending=False).head(10)
```

Out[154…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PI
5415	43332	236	236116	239252	12	Natural Chip Co Tmato Hrb&Spce 175g	
32796	43603	236	236033	238735	59	Old El Paso Salsa Dip Tomato Med 300g	
5107	43329	54	54225	48172	46	Kettle Original 175g	
80732	43603	49	49309	45816	30	Doritos Corn Chips Cheese Supreme 170g	
32762	43331	227	227046	228561	100	Smiths Crinkle Cut Chips Chs&Onion170g	
32765	43328	227	227134	229083	95	Sunbites Whlegrn Crisps Frch/Onin 90g	
135336	43604	95	95317	95544	46	Kettle Original 175g	
5109	43330	57	57122	51950	9	Kettle Tortilla ChpsBtroot&Ricotta 150g	
150322	43599	6	6465	6276	64	Red Rock Deli SR Salsa & Mzzrlla 150g	
17145	43328	202	202289	202104	42	Doritos Corn Chip Mexican Jalapeno 150g	

In [155…

```
transaction_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264834 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   DATE            264834 non-null  int64  
 1   STORE_NBR       264834 non-null  int64  
 2   LYLTY_CARD_NBR  264834 non-null  int64  
 3   TXN_ID          264834 non-null  int64  
 4   PROD_NBR        264834 non-null  int64  
 5   PROD_NAME       264834 non-null  object 
 6   PROD_QTY        264834 non-null  int64  
 7   TOT_SALES       264834 non-null  float64
dtypes: float64(1), int64(6), object(1)
memory usage: 18.2+ MB
```

In [156…
```
transaction_df_cleaned = transaction_df.copy()
transaction_df_cleaned['PROD_PRICE_$'] = transaction_df_cleaned['TOT_SALES']/transa
transaction_df_cleaned['DATE'] = pd.to_datetime(transaction_df_cleaned['DATE'], ori
transaction_df_cleaned['YEAR'] = pd.to_datetime(transaction_df_cleaned['DATE']).dt.
transaction_df_cleaned['MONTH'] = pd.to_datetime(transaction_df_cleaned['DATE']).dt
transaction_df_cleaned['PROD_WEIGHT_G'] = transaction_df_cleaned['PROD_NAME'].str.e
transaction_df_cleaned['PROD_WEIGHT_G'] = transaction_df_cleaned['PROD_WEIGHT_G'].s
transaction_df_cleaned['PROD_WEIGHT_G'] = transaction_df_cleaned['PROD_WEIGHT_G'].a
transaction_df_cleaned.sample(10)
```

Out[156…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_C
248472	2018-08-22	133	133211	137404	24	Grain Waves Sweet Chilli 210g	
209801	2019-04-07	114	114026	117048	10	RRD SR Slow Rst Pork Belly 150g	
258790	2019-05-18	266	266108	263944	104	Infuzions Thai SweetChili PotatoMix 110g	
106242	2018-09-29	95	95022	94210	3	Kettle Sensations Camembert & Fig 150g	
133385	2018-07-04	238	238050	242079	89	Kettle Sweet Chilli And Sour Cream 175g	
235485	2019-01-17	207	207002	204596	83	WW D/Style Chip Sea Salt 200g	
113635	2019-03-26	213	213069	212302	63	Kettle 135g Swt Pot Sea Salt	
173974	2018-10-04	39	39087	35156	34	Pringles Slt Vingar 134g	
154021	2019-02-21	41	41450	38432	15	Twisties Cheese 270g	
227530	2019-05-01	239	239175	243723	62	Pringles Mystery Flavour 134g	

In [157…
```
customer_transaction_merged = customer_info.merge(transaction_df_cleaned, on="LYLTY
customer_transaction_merged
```

Out[157…

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER	DATE	STORE_NBR	TX
0	1000	YOUNG SINGLES/COUPLES	Premium	2018-10-17	1	
1	1002	YOUNG SINGLES/COUPLES	Mainstream	2018-09-16	1	
2	1003	YOUNG FAMILIES	Budget	2019-03-07	1	
3	1003	YOUNG FAMILIES	Budget	2019-03-08	1	
4	1004	OLDER SINGLES/COUPLES	Mainstream	2018-11-02	1	
…	…	…	…	…	…	
264829	2370701	YOUNG FAMILIES	Mainstream	2018-12-08	88	2
264830	2370751	YOUNG FAMILIES	Premium	2018-10-01	88	2
264831	2370961	OLDER FAMILIES	Budget	2018-10-24	88	2
264832	2370961	OLDER FAMILIES	Budget	2018-10-27	88	2
264833	2373711	YOUNG SINGLES/COUPLES	Mainstream	2018-12-14	88	2

264834 rows × 14 columns

In [158…

```
title_order = ["LYLTY_CARD_NBR", "LIFESTAGE", "PREMIUM_CUSTOMER", "DATE", "YEAR", "
customer_transaction_merged = customer_transaction_merged[title_order]
customer_transaction_merged
```

Out[158…

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER	DATE	YEAR	MONTH
0	1000	YOUNG SINGLES/COUPLES	Premium	2018-10-17	2018	10
1	1002	YOUNG SINGLES/COUPLES	Mainstream	2018-09-16	2018	9
2	1003	YOUNG FAMILIES	Budget	2019-03-07	2019	3
3	1003	YOUNG FAMILIES	Budget	2019-03-08	2019	3
4	1004	OLDER SINGLES/COUPLES	Mainstream	2018-11-02	2018	11
…	…	…	…	…	…	…
264829	2370701	YOUNG FAMILIES	Mainstream	2018-12-08	2018	12
264830	2370751	YOUNG FAMILIES	Premium	2018-10-01	2018	10
264831	2370961	OLDER FAMILIES	Budget	2018-10-24	2018	10
264832	2370961	OLDER FAMILIES	Budget	2018-10-27	2018	10
264833	2373711	YOUNG SINGLES/COUPLES	Mainstream	2018-12-14	2018	12

264834 rows × 14 columns

In [159…

```
customer_transaction_merged.isnull().sum() # No missing values, can begin analysis
```

Out[159…
```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
DATE                0
YEAR                0
MONTH               0
STORE_NBR           0
TXN_ID              0
PROD_NBR            0
PROD_NAME           0
PROD_QTY            0
TOT_SALES           0
PROD_PRICE_$        0
PROD_WEIGHT_G       0
dtype: int64
```

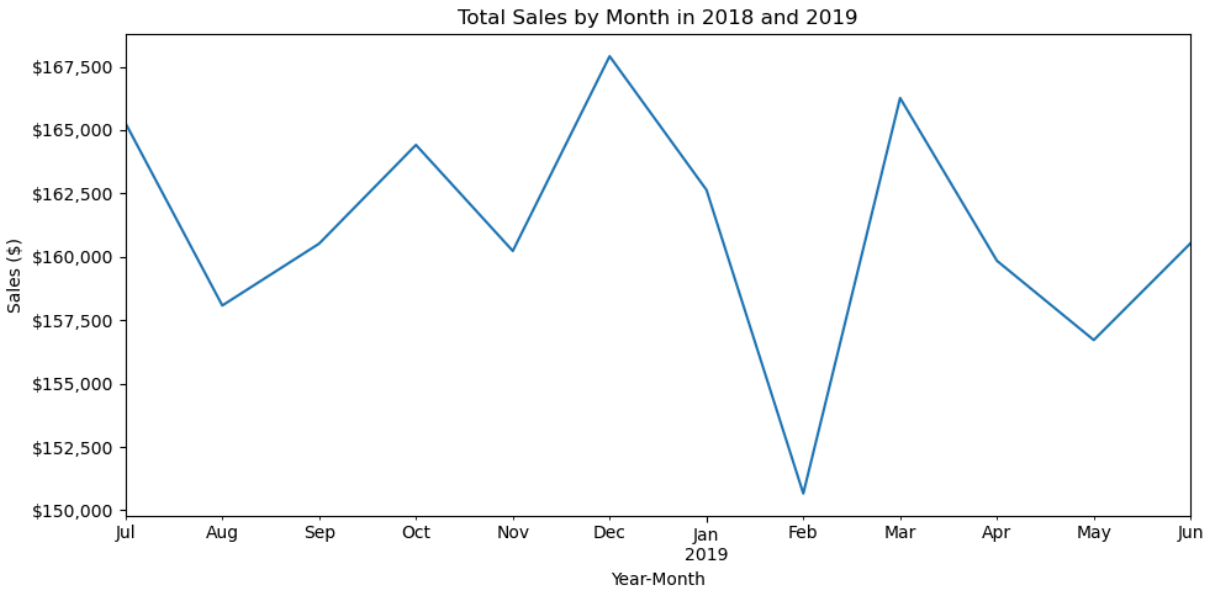
Topics to answer

1. Has the total sales of the store improved over the years, to evaluate the financial status of the company. (only 2 years so not useful)
2. Which item is the most popular based on each customer status and lifestage
3. Do premium customer tend to buy in a higher quantity or higher average price

1. Has the total sales of the store improved over the years, to evaluate the financial status of the company. (only 2 years so not useful)

In [160…
```
monthly_sales = customer_transaction_merged.groupby(
    customer_transaction_merged['DATE'].dt.to_period('M') # group by month
)['TOT_SALES'].sum()
monthly_sales.plot(kind='line', figsize=(10, 5))
plt.title('Total Sales by Month in 2018 and 2019')
plt.ylabel('Sales ($)')
plt.xlabel('Year-Month')
plt.gca().yaxis.set_major_formatter(ticker.StrMethodFormatter('${x:,.0f}'))
plt.tight_layout()
plt.plot()
```

Out[160…
```
[]
```



Topic 1 Analysis

- Monthly sales from July 2018 to June 2019 fluctuated between $151,000 and 167,500, an approximate 10% variation. This reflects a stable financial performance throughout the year. Sales peaked in December, likely due to seasonal demand, which presents an opportunity for holiday promotions.
- The lowest sales occurred in February, potentially due to reduced demand following high volume holiday purchases. To avoid overstocking, bundling strategies or targeted promotions during slower months could help drive additional revenue.
- Overall, the consistent monthly range indicates steady and predictable market demand, offering a strong foundation for category planning and future promotional strategies.

2. How does customer status and lifestage affect their purchasing pattern of the top 5 items.

In [161…
```
top_5_items = (
    customer_transaction_merged['PROD_NAME'].value_counts().head(5).index
    )
top_5_items
```

Out[161…
```
Index(['Kettle Mozzarella   Basil & Pesto 175g',
       'Kettle Tortilla ChpsHny&Jlpno Chili 150g',
       'Cobs Popd Swt/Chlli &Sr/Cream Chips 110g',
       'Tyrrells Crisps     Ched & Chives 165g',
       'Cobs Popd Sea Salt  Chips 110g'],
      dtype='object', name='PROD_NAME')
```

In [162…
```
top_5_df = customer_transaction_merged[customer_transaction_merged["PROD_NAME"].isi
top_5_df
```

Out[162…

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER	DATE	YEAR	MONTH
37	1034	RETIREES	Premium	2019-03-24	2019	3
47	1046	YOUNG SINGLES/COUPLES	Budget	2019-02-13	2019	2
92	1086	YOUNG SINGLES/COUPLES	Budget	2019-02-13	2019	2
106	1097	RETIREES	Budget	2019-01-24	2019	1
116	1104	RETIREES	Mainstream	2019-05-23	2019	5
…	…	…	…	…	…	…
264784	272390	NEW FAMILIES	Budget	2019-05-11	2019	5
264787	272392	MIDAGE SINGLES/COUPLES	Premium	2018-09-29	2018	9
264801	883791	OLDER SINGLES/COUPLES	Mainstream	2018-09-29	2018	9
264804	2330031	MIDAGE SINGLES/COUPLES	Premium	2018-07-07	2018	7
264823	2370001	OLDER SINGLES/COUPLES	Premium	2018-08-10	2018	8

16402 rows × 14 columns

In [163…

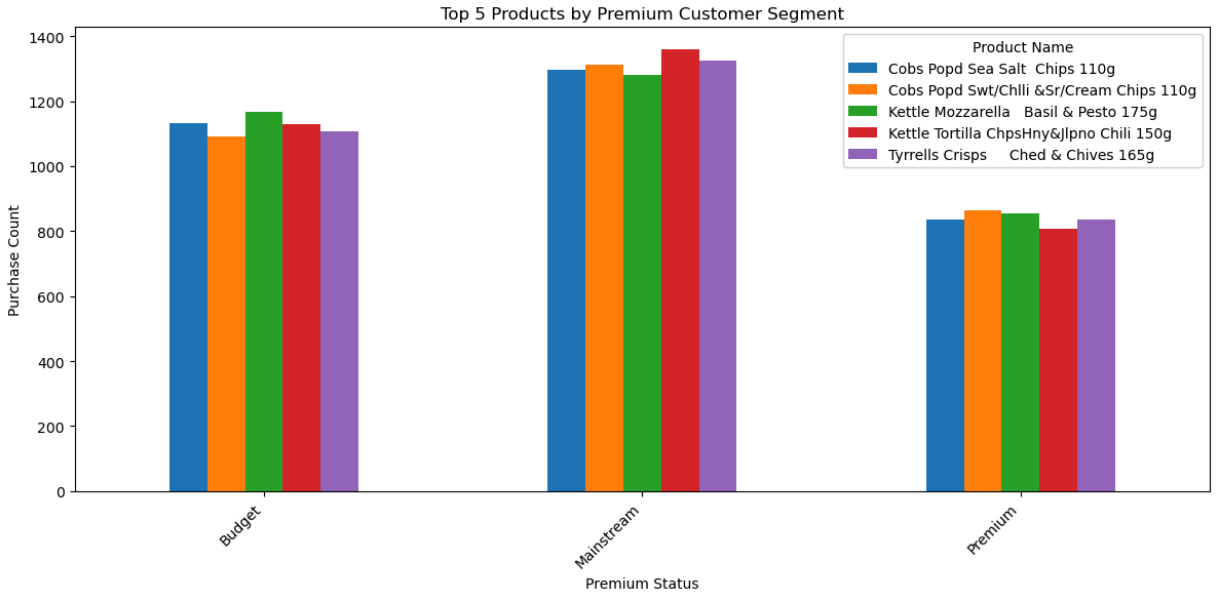
```
top_5_premium = top_5_df.groupby("PREMIUM_CUSTOMER")["PROD_NAME"].value_counts()
top_5_premium
```

Out[163…

```
PREMIUM_CUSTOMER  PROD_NAME
Budget            Kettle Mozzarella   Basil & Pesto 175g      1166
                  Cobs Popd Sea Salt  Chips 110g              1132
                  Kettle Tortilla ChpsHny&Jlpno Chili 150g    1128
                  Tyrrells Crisps     Ched & Chives 165g      1108
                  Cobs Popd Swt/Chlli &Sr/Cream Chips 110g    1091
Mainstream        Kettle Tortilla ChpsHny&Jlpno Chili 150g    1360
                  Tyrrells Crisps     Ched & Chives 165g      1324
                  Cobs Popd Swt/Chlli &Sr/Cream Chips 110g    1313
                  Cobs Popd Sea Salt  Chips 110g              1298
                  Kettle Mozzarella   Basil & Pesto 175g      1282
Premium           Cobs Popd Swt/Chlli &Sr/Cream Chips 110g     865
                  Kettle Mozzarella   Basil & Pesto 175g       856
                  Tyrrells Crisps     Ched & Chives 165g       836
                  Cobs Popd Sea Salt  Chips 110g               835
                  Kettle Tortilla ChpsHny&Jlpno Chili 150g     808
Name: count, dtype: int64
```

In [164…

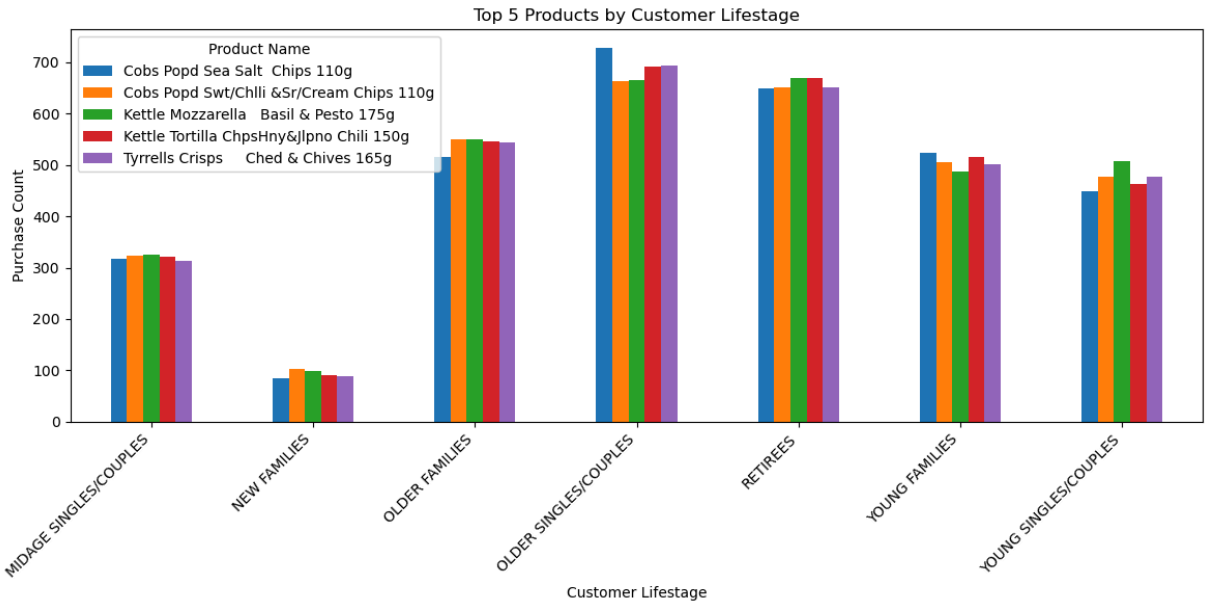
```
top_5_premium_unstacked = top_5_premium.unstack() # unstack makes PROD_NAME first r
top_5_premium_unstacked.plot(kind='bar', figsize=(12, 6))
plt.title('Top 5 Products by Premium Customer Segment')
plt.xlabel('Premium Status')
plt.ylabel('Purchase Count')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Product Name')
plt.tight_layout()
plt.show()
```



In [165…

```
top_5_lifestage = top_5_df.groupby("LIFESTAGE")["PROD_NAME"].value_counts()
top_5_lifestage_unstacked = top_5_lifestage.unstack()
top_5_lifestage_unstacked.plot(kind="bar", figsize=(12,6))
plt.title('Top 5 Products by Customer Lifestage')
plt.xlabel('Customer Lifestage')
plt.ylabel('Purchase Count')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Product Name')
```

```
plt.tight_layout()
plt.show()
```



Topic 2 Anaysis: Top 5 Chips:

- Two of the top selling chips are from the brands Cobs Popped and Kettle, suggesting these two brands are key drivers of chip sales.
- Two of the top chips have Chili flavor, suggesting a potential trend towards spicy chips.
- The store manager could inquire about more Spicy chips from these top brands to boost sale while the flavor is trending.

Premium Customer

- Purchase distribution across all premium customer segments is relatively even, suggesting the top five chips have broad appeal regardless of price sensitivity.
- The Mainstream segment accounts for the highest number of purchases, followed by Budget customers, implying that the majority of buyers fall into middle or lower-income tiers.
- While price cuts may not be feasible, promotional bundles or markdowns on soon-to-expire products could appeal to these cost-conscious segments and help drive volume.

Customer Lifestage

- Similar to Premium Customer, the purchase pattern for the top 5 chips are distributed evenly.
- However, there is notable polarization in customer ages, with most purchases coming from either younger or older lifestage groups.

3. Do customers tend to buy in a higher quantity or higher average price

In [166…
```
customer_quartile_df = customer_transaction_merged.copy()
customer_quartile_df["PRICE_QUARTILE"] = pd.qcut(customer_quartile_df["PROD_PRICE_$
customer_quartile_df["WEIGHT_QUARTILE"] = pd.qcut(customer_quartile_df["PROD_WEIGHT
customer_quartile_df
```

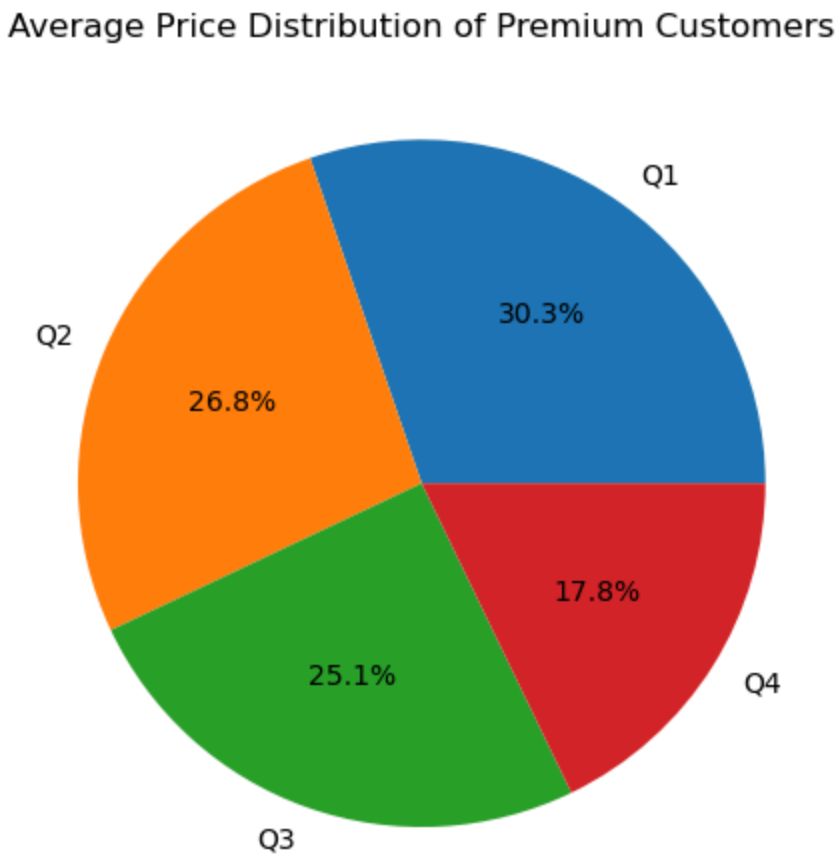
Out[166…

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER	DATE	YEAR	MONTH
0	1000	YOUNG SINGLES/COUPLES	Premium	2018-10-17	2018	10
1	1002	YOUNG SINGLES/COUPLES	Mainstream	2018-09-16	2018	9
2	1003	YOUNG FAMILIES	Budget	2019-03-07	2019	3
3	1003	YOUNG FAMILIES	Budget	2019-03-08	2019	3
4	1004	OLDER SINGLES/COUPLES	Mainstream	2018-11-02	2018	11
…	…	…	…	…	…	…
264829	2370701	YOUNG FAMILIES	Mainstream	2018-12-08	2018	12
264830	2370751	YOUNG FAMILIES	Premium	2018-10-01	2018	10
264831	2370961	OLDER FAMILIES	Budget	2018-10-24	2018	10
264832	2370961	OLDER FAMILIES	Budget	2018-10-27	2018	10
264833	2373711	YOUNG SINGLES/COUPLES	Mainstream	2018-12-14	2018	12

264834 rows × 16 columns

In [167…
```
customer_quartile_df_average_price = customer_quartile_df["PRICE_QUARTILE"].value_c
customer_quartile_df_average_price.plot(
    kind="pie",
    autopct='%1.1f%%',
```

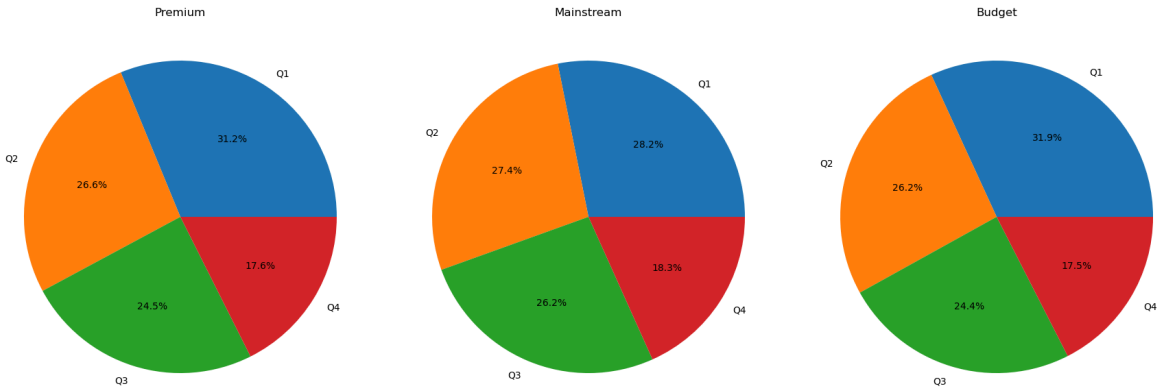
```
    ylabel=''
)
plt.title("Average Price Distribution of Premium Customers")
plt.tight_layout()
plt.show()
```



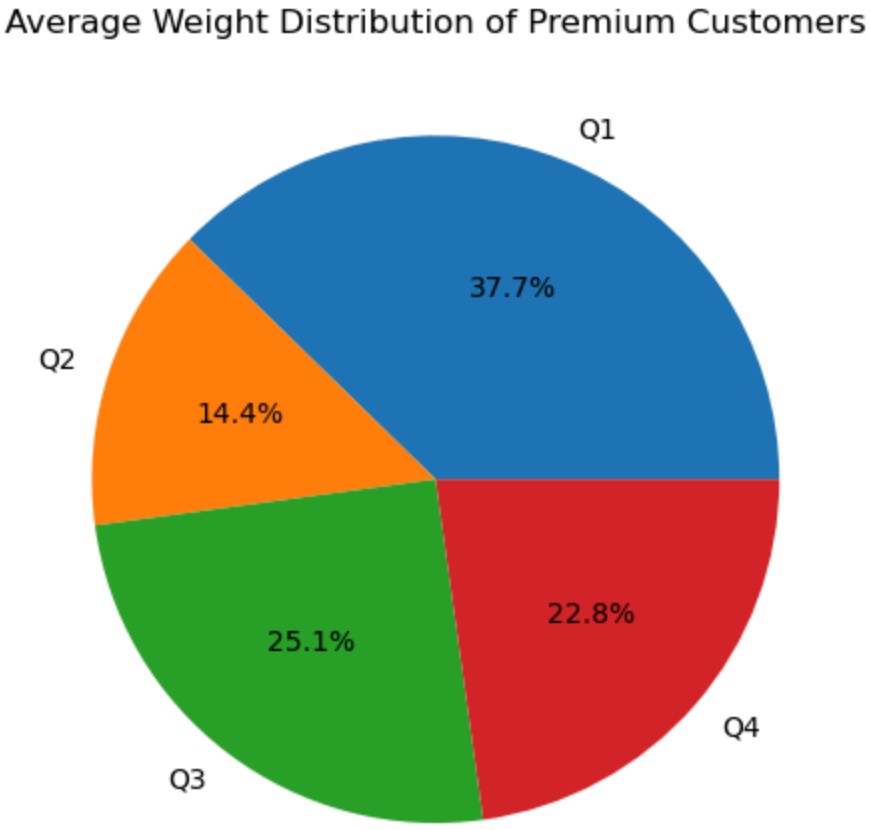
In [168…

```
customer_quartile_df_customers = customer_quartile_df.groupby("PREMIUM_CUSTOMER")
customer_quartile_df_customers = customer_quartile_df_customers["PRICE_QUARTILE"].v
customer_quartile_df_customers_unstack = customer_quartile_df_customers.unstack()

fig, axes = plt.subplots(1, 3, figsize=(18, 6))
segments = customer_quartile_df["PREMIUM_CUSTOMER"].unique()
for i, segment in enumerate(segments):
    data = customer_quartile_df[customer_quartile_df["PREMIUM_CUSTOMER"] == segment
    data.plot(
        kind='pie',
        autopct='%1.1f%%',
        ylabel='',
        ax=axes[i],
        title=segment
    )
plt.tight_layout()
plt.show()
```

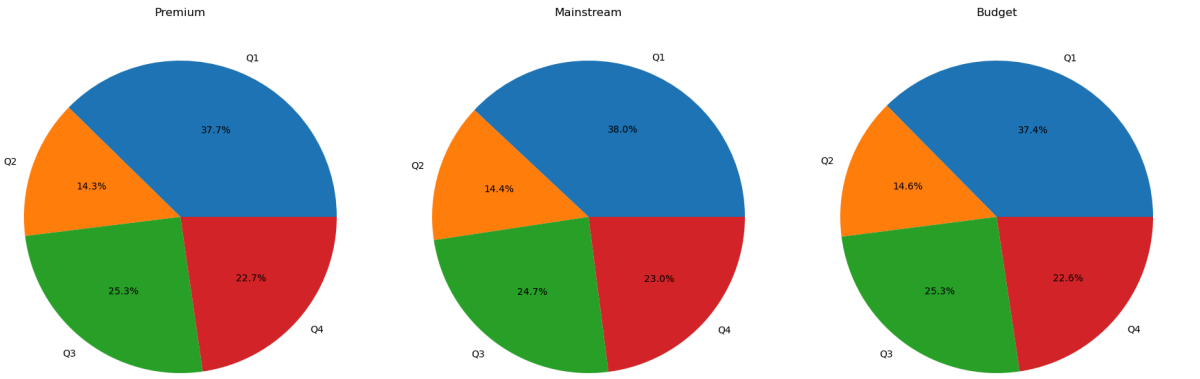


In [169…
```
customer_quartile_df_average_weight = customer_quartile_df["WEIGHT_QUARTILE"].value
customer_quartile_df_average_weight.plot(
    kind="pie",
    autopct='%1.1f%%',
    ylabel=''
)
plt.title("Average Weight Distribution of Premium Customers")
plt.tight_layout()
plt.show()
```



In [170…
```
customer_quartile_df_customers = customer_quartile_df.groupby("PREMIUM_CUSTOMER")
customer_quartile_df_customers = customer_quartile_df_customers["WEIGHT_QUARTILE"].
customer_quartile_df_customers_unstack = customer_quartile_df_customers.unstack()
```

```
fig, axes = plt.subplots(1, 3, figsize=(18, 6))
segments = customer_quartile_df["PREMIUM_CUSTOMER"].unique()
for i, segment in enumerate(segments):
    data = customer_quartile_df[customer_quartile_df["PREMIUM_CUSTOMER"] == segment
    data.plot(
        kind='pie',
        autopct='%1.1f%%',
        ylabel='',
        ax=axes[i],
        title=segment
    )
plt.tight_layout()
plt.show()
```



Topic 3 Analysis:

Price Distribution

- The price distribution is evenly distributed among customer's premium status. All customers show similar purchasing behavior when segmented by price quartiles.
- Even premium customers, typically assumed to prioritize quality over cost, demonstrate similarly toward value.
- These findings imply that price sensitivity is a general trend among all customer types, not just budget conscious ones. Future promotions could focus on highlighting value, even for higher tier items.

Weight Distribution

- The weight distribution is evenly distributed among customer's premium status. All customers have similar purchasing behavior when segmented by weight quartiles.
- The largest distribution of items sold are the lightest items. From previous observations, these items are smaller snacks which are cheaper and easily consumable.
- Products that are slightly below average (Q2) are the least purchased. This may be caused by higher price, too small to share with a group, or too much for a single person to purchase. This trend suggests that the store manager should either purchase items that are smaller in size or larger.