In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import datetime as dt
import nltk
import re
sns.set(rc={'figure.figsize':(16,9)})
```

Importing data

Customer data

In [2]:
```
df_cust = pd.read_csv('QVI_purchase_behaviour.csv')
```

In [3]:
```
df_cust_copy = df_cust.copy()
```

In [4]:
```
df_cust.head()
```

Out[4]:

	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 [5]:
```
df_cust.isnull().sum()
```

Out[5]:
```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

In [6]:
```
df_cust.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 [7]:
```
df_cust.describe()
```

Out[7]:

	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 [8]:
```
df_cust.drop(df_cust[df_cust['LYLTY_CARD_NBR'] == 226000].index,inplace=T
```

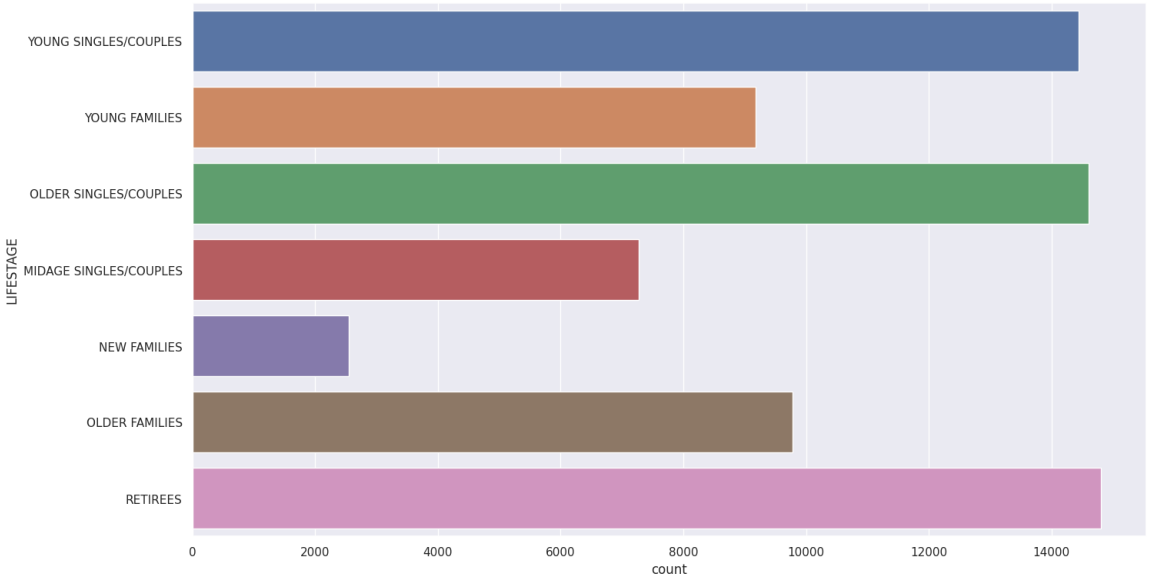
In [9]:
```
df_cust['LIFESTAGE'].value_counts()
```

Out[9]:
```
RETIREES                  14805
OLDER SINGLES/COUPLES     14609
YOUNG SINGLES/COUPLES     14441
OLDER FAMILIES             9779
YOUNG FAMILIES             9178
MIDAGE SINGLES/COUPLES     7275
NEW FAMILIES               2549
Name: LIFESTAGE, dtype: int64
```

In [10]:
```
prem_custs = pd.DataFrame(df_cust['PREMIUM_CUSTOMER'].value_counts().reset
```

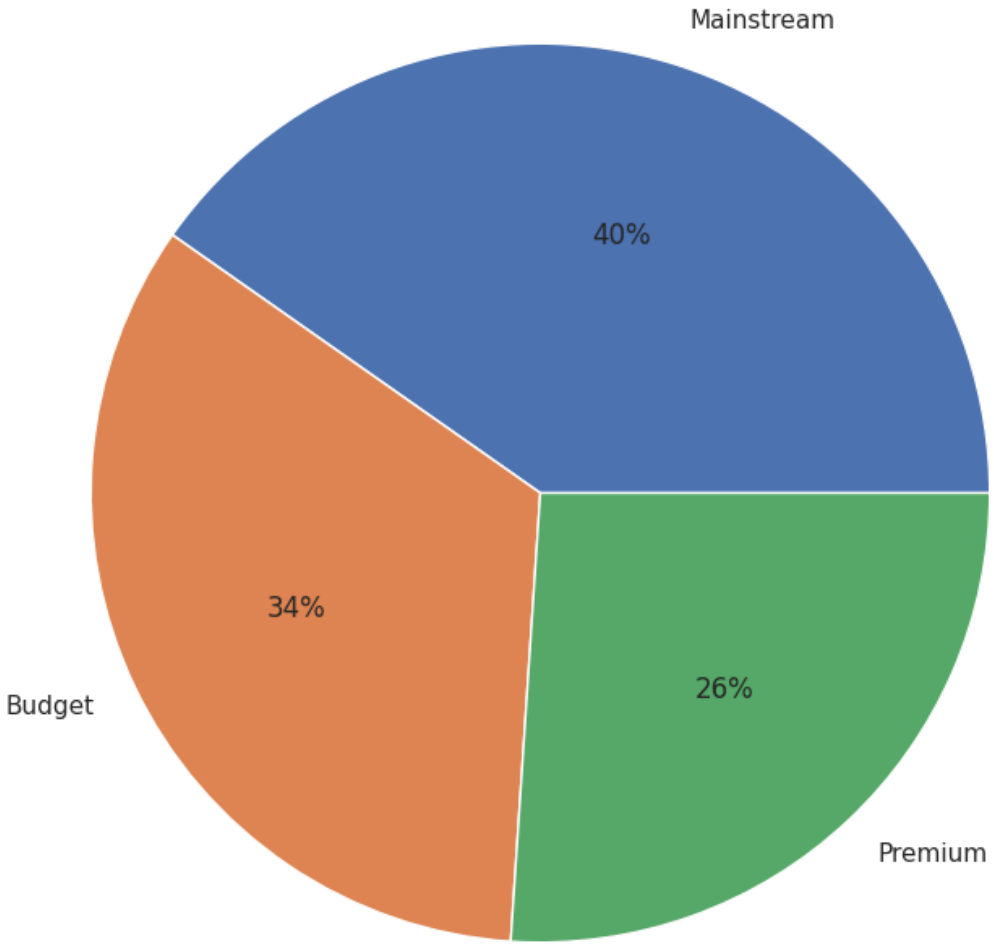
In [12]:
```
sns.countplot(y = 'LIFESTAGE',data = df_cust)
```

Out[12]:
```
<AxesSubplot: xlabel='count', ylabel='LIFESTAGE'>
```



In [13]:
```
plt.pie(prem_custs['Count'],labels=prem_custs['PREMIUM_CUSTOMER'],autopct=
```

Out[13]:
```
([<matplotlib.patches.Wedge at 0x7f90f4491930>,
  <matplotlib.patches.Wedge at 0x7f90f4491ea0>,
  <matplotlib.patches.Wedge at 0x7f90f44923b0>],
 [Text(0.3312830284596076, 1.0489287654815416, 'Mainstream'),
  Text(-0.9921483905603237, -0.47501744295189796, 'Budget'),
  Text(0.7517648941126204, -0.8030252449206319, 'Premium')],
 [Text(0.1806998337052405, 0.5721429629899317, '40%'),
  Text(-0.5411718493965401, -0.25910042342830797, '34%'),
  Text(0.4100535786068838, -0.43801376995670827, '26%')])
```



- Most customers are from Retirees group, followed by Young singles/couples and Old singles/couples.

Transaction data

In [14]:
```
df_trns = pd.read_excel('QVI_transaction_data.xlsx')
```

In [15]:
```
df_trns_copy = df_trns.copy()
```

In [16]:
```
df_trns.head()
```

Out[16]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY
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 [17]:
```
df_trns.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
dtype: int64
```

In [18]:
```
df_trns.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 [19]:
```
df_trns.describe()
```

Out[19]:

	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	1
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	1
25%	43373.000000	70.00000	7.002100e+04	6.760150e+04	28.000000	2
50%	43464.000000	130.00000	1.303575e+05	1.351375e+05	56.000000	2
75%	43555.000000	203.00000	2.030942e+05	2.027012e+05	85.000000	2
max	43646.000000	272.00000	2.373711e+06	2.415841e+06	114.000000	20

In [20]:
```
df_trns['PROD_NAME'].value_counts()
```

Out[20]:
```
Kettle Mozzarella   Basil & Pesto 175g      3304
Kettle Tortilla ChpsHny&Jlpno Chili 150g    3296
Cobs Popd Swt/Chlli &Sr/Cream Chips 110g    3269
Tyrrells Crisps     Ched & Chives 165g      3268
Cobs Popd Sea Salt  Chips 110g              3265
                                            ...
RRD Pc Sea Salt     165g                    1431
Woolworths Medium   Salsa 300g              1430
NCC Sour Cream &    Garden Chives 175g      1419
French Fries Potato Chips 175g              1418
WW Crinkle Cut      Original 175g           1410
Name: PROD_NAME, Length: 114, dtype: int64
```

In [21]:
```
df_trns['PROD_QTY'].value_counts()
```

Out[21]:
```
2      236039
1       27518
5         450
3         430
4         397
200         2
Name: PROD_QTY, dtype: int64
```

In [22]:
```
df_trns.loc[df_trns['PROD_QTY'] == 200,:]
```

Out[22]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_Q
69762	43331	226	226000	226201	4	Dorito Corn Chp Supreme 380g	2
69763	43605	226	226000	226210	4	Dorito Corn Chp Supreme 380g	2

- Probably a commercial purchase

In [23]:
```
df_trns.drop(df_trns[df_trns['PROD_QTY'] == 200].index,inplace=True)
```

In [24]:
```
df_trns.loc[df_trns['PROD_QTY'] == 200,:]
```

Out[24]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	T

In [25]:
```
def date_format(date):
    anc = dt.datetime(1900,1,1)
    if date < 60:
        dd = dt.timedelta(days=date-1)
    else:
        dd = dt.timedelta(days=date-2)
    return anc + dd
```

In [26]:
```
df_trns['DATE'] = df_trns['DATE'].apply(date_format)
```

In [27]:
```
len(df_trns) == df_trns.TXN_ID.nunique()
```

Out[27]:
```
False
```

In [28]:
```
df_trns[df_trns.duplicated('TXN_ID')].head()
```

Out[28]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD
42	2019-05-20	55	55073	48887	113	Twisties Chicken270g	
377	2019-01-10	7	7364	7739	20	Doritos Cheese Supreme 330g	
419	2018-10-18	12	12301	10982	93	Doritos Corn Chip Southern Chicken 150g	
476	2018-09-08	16	16427	14546	81	Pringles Original Crisps 134g	
511	2018-08-03	19	19272	16683	31	Infzns Crn Crnchers Tangy Gcamole 110g	

In [29]:
```
df_trns.loc[df_trns['TXN_ID'] == 7739,:]
```

Out[29]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD
376	2019-01-10	7	7364	7739	50	Tostitos Lightly Salted 175g	
377	2019-01-10	7	7364	7739	20	Doritos Cheese Supreme 330g	

In [30]:
```
df_trns['PROD_SIZE'] = df_trns['PROD_NAME'].str.extract('(\d+)')
df_trns['PROD_SIZE'] = pd.to_numeric(df_trns['PROD_SIZE'])
```

In [31]:
```
def process_txt(txt):
    txt = re.sub('\d\w*',' ',txt)
    txt = re.sub('[&/]',' ',txt)
    return txt
```

In [32]:
```
df_trns['PROD_NAME'] = df_trns['PROD_NAME'].apply(process_txt)
```

In [33]:
```
df_trns['BRAND'] = df_trns['PROD_NAME'].str.partition()[0]
```

In [34]:
```
prod_name = ''.join(df_trns['PROD_NAME'])
prod_names = nltk.word_tokenize(prod_name)
```

In [35]:
```
pd.DataFrame(list(nltk.probability.FreqDist(prod_names).items()),columns=
```

Out[35]:

	Word	Frequency
10	Chips	49770
16	Kettle	40739
7	Smiths	28572
6	Cheese	27890
66	Pringles	24743

In [36]:
```
grp_sales = df_trns.groupby('DATE')[['TOT_SALES']].sum()
```

In [37]:
```
grp_sales.loc['2018-12-25'] = 0
```

In [38]:
```
grp_sales.reset_index(inplace=True)
```

```
/tmp/ipykernel_9812/949551136.py:1: FutureWarning: Inferring datetime64[n
s] from data containing strings is deprecated and will be removed in a fu
ture version. To retain the old behavior explicitly pass Series(data, dty
pe=datetime64[ns])
  grp_sales.reset_index(inplace=True)
```

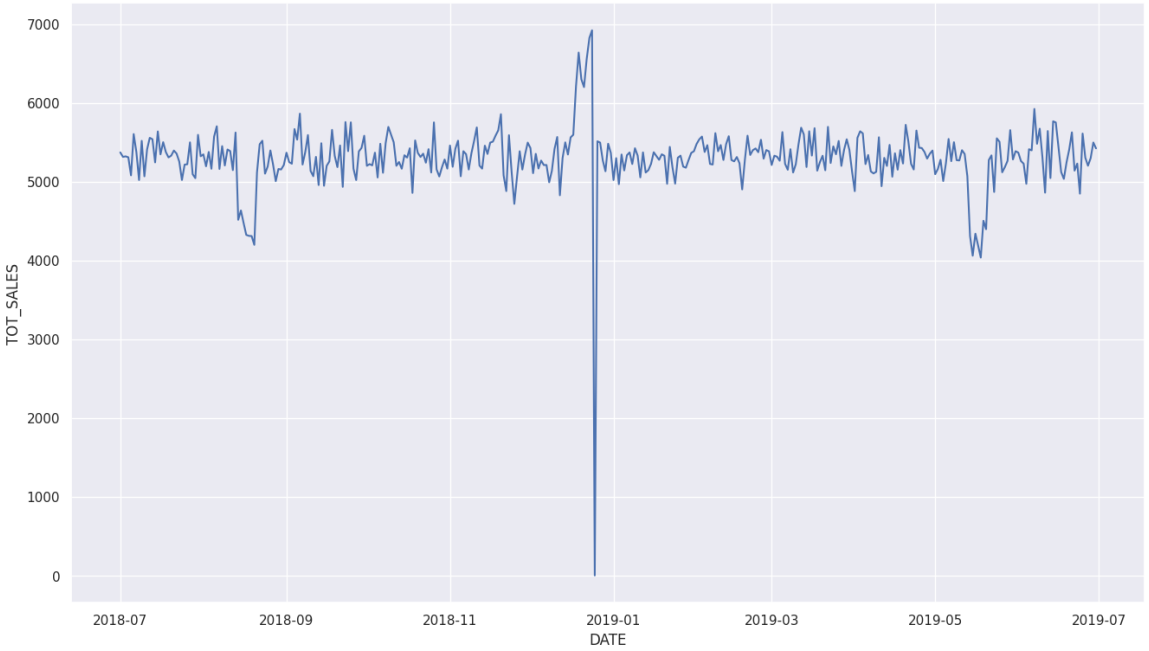
In [39]:
```
grp_sales.head()
```

Out[39]:

	DATE	TOT_SALES
0	2018-07-01	5372.2
1	2018-07-02	5315.4
2	2018-07-03	5321.8
3	2018-07-04	5309.9
4	2018-07-05	5080.9

In [40]:
```
sns.lineplot(data=grp_sales,x='DATE',y='TOT_SALES')
```

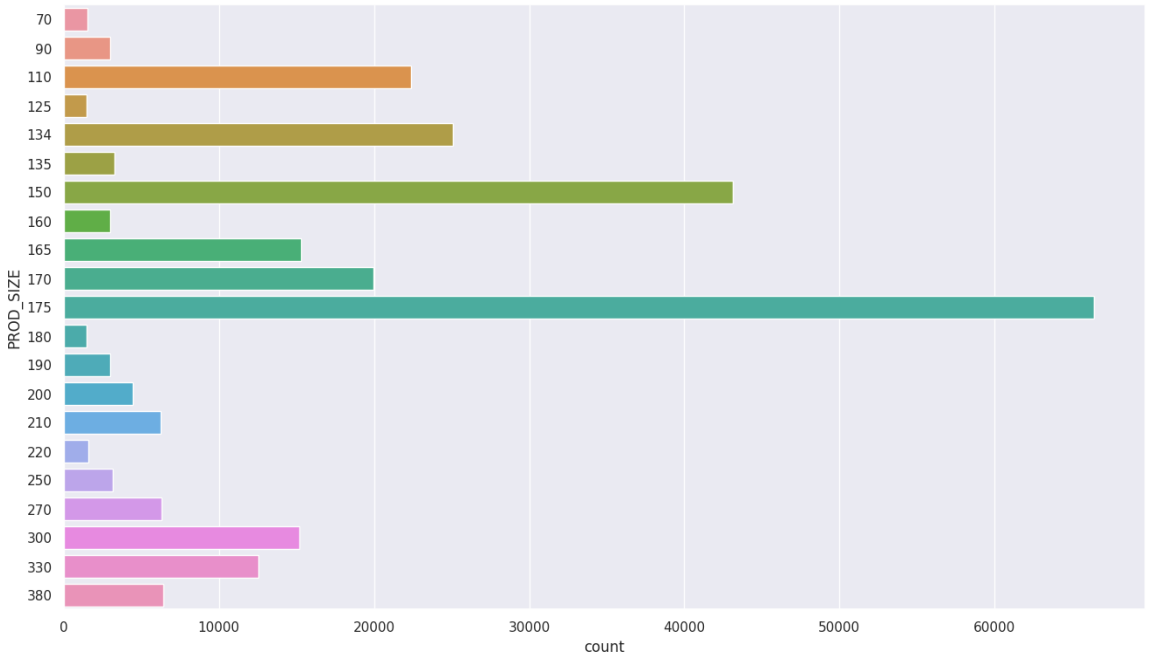
Out[40]:
```
<AxesSubplot: xlabel='DATE', ylabel='TOT_SALES'>
```



- Stores closed on Christmas, hence sales are zero on 2018-12-25

In [41]:
```
sns.countplot(data=df_trns,y='PROD_SIZE')
```

Out[41]:
```
<AxesSubplot: xlabel='count', ylabel='PROD_SIZE'>
```



In [42]:
```
brands = {'Ncc':'Natural',
          'NCC':'Natural',
          'Ccs':'CCS',
          'CCs':'CCS',
          'Smith':'Smiths',
          'Grain':'Grainwaves',
          'Grnwves':'Grainwaves',
          'GrnWves':'Grainwaves',
          'Ww':'Woolworths',
          'WW':'Woolworths',
          'Infzns':'Infuzions',
          'Red':'Red Rock Deli',
          'Rrd':'Red Rock Deli',
          'RRD':'Red Rock Deli',
          'Snbts':'Sunbites'}
df_trns['BRAND'] = df_trns['BRAND'].map(brands).fillna(df_trns['BRAND'])
```

In [43]:
```
df_trns['BRAND'].unique()
```

Out[43]:
```
array(['Natural', 'CCS', 'Smiths', 'Kettle', 'Old', 'Grainwaves',
       'Doritos', 'Twisties', 'Woolworths', 'Thins', 'Burger', 'Cheezels
',
       'Infuzions', 'Red Rock Deli', 'Pringles', 'Dorito', 'Tyrrells',
       'Cobs', 'French', 'Tostitos', 'Cheetos', 'Sunbites'], dtype=objec
t)
```

In [44]:
```
brand_df = df_trns.groupby('BRAND')['TOT_SALES'].sum().reset_index().sort_
```

In [45]:
```
brand_df.head()
```

Out[45]:

	BRAND	TOT_SALES
10	Kettle	390239.8
15	Smiths	224660.2
6	Doritos	201538.9
13	Pringles	177655.5
9	Infuzions	99047.6

Customer segments

In [46]:
```
merged_df = pd.merge(df_cust,df_trns)
```

In [110…
```
segment_sales = merged_df.groupby(['PREMIUM_CUSTOMER','LIFESTAGE'])['TOT_S
```

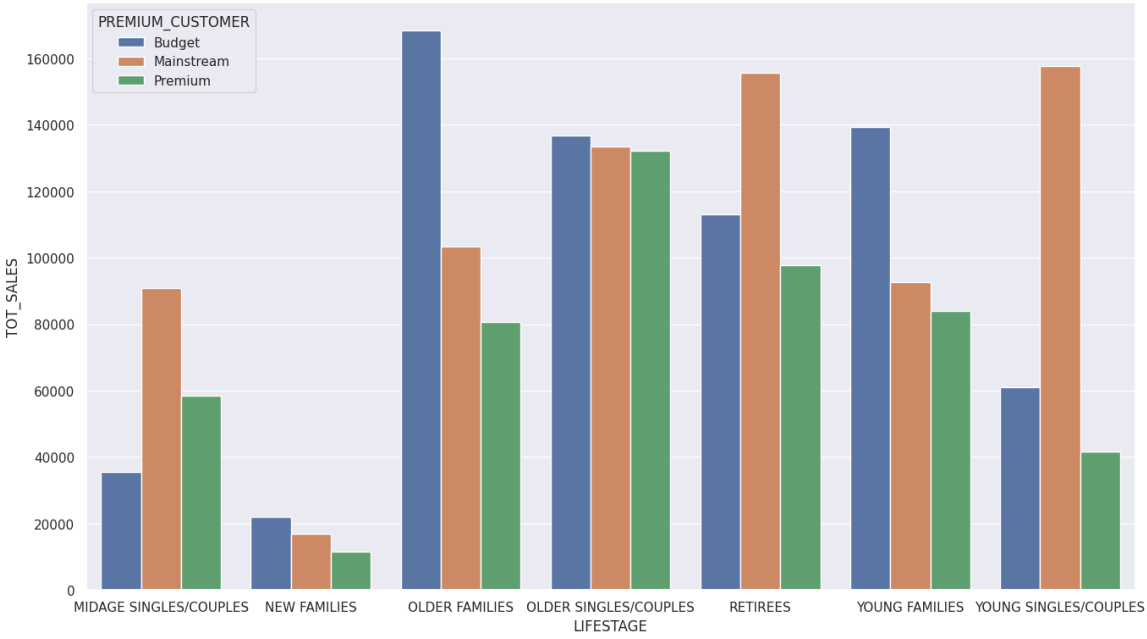
In [120…
```
segment_sales.head()
```

Out[120]:

	PREMIUM_CUSTOMER	LIFESTAGE	TOT_SALES
0	Budget	MIDAGE SINGLES/COUPLES	35514.80
1	Budget	NEW FAMILIES	21928.45
2	Budget	OLDER FAMILIES	168363.25
3	Budget	OLDER SINGLES/COUPLES	136769.80
4	Budget	RETIREES	113147.80

In [121…
```
sns.barplot(data=segment_sales,x='LIFESTAGE', y='TOT_SALES', hue='PREMIUM_
```

Out[121]:
```
<AxesSubplot: xlabel='LIFESTAGE', ylabel='TOT_SALES'>
```



Most sales come from

- Budget Older families
- Mainstream young singles/couples
- Mainstream retirees

In [125…
```
segment_qty = merged_df.groupby(['PREMIUM_CUSTOMER','LIFESTAGE'])['PROD_Q
```

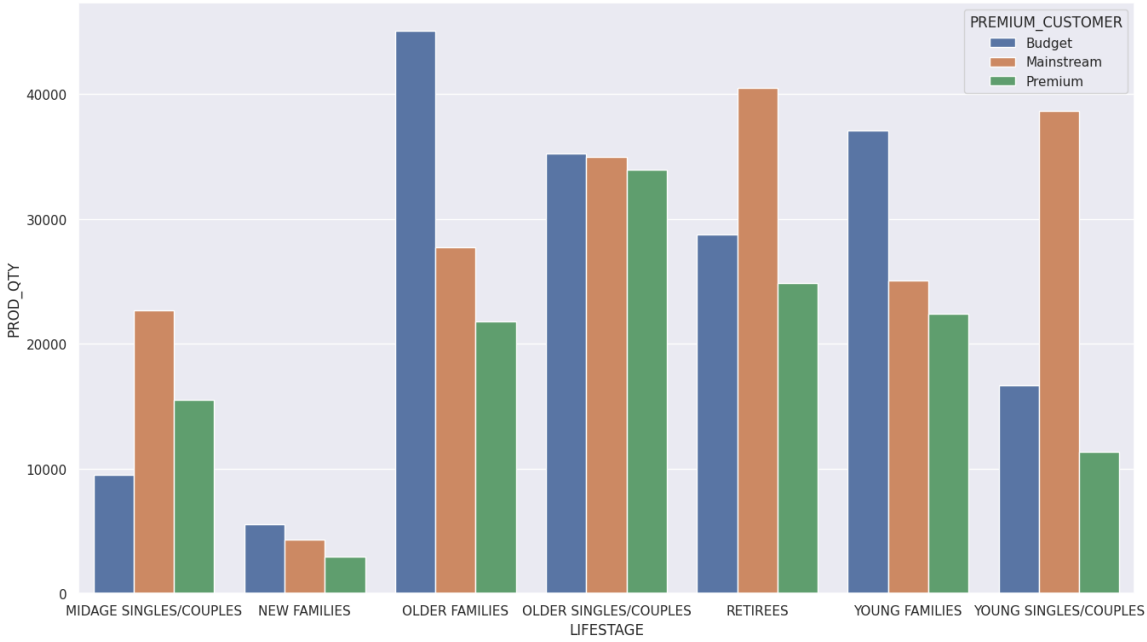
In [127…
```
segment_qty.head()
```

Out[127]:

	PREMIUM_CUSTOMER	LIFESTAGE	PROD_QTY
0	Budget	MIDAGE SINGLES/COUPLES	9496
1	Budget	NEW FAMILIES	5571
2	Budget	OLDER FAMILIES	45065
3	Budget	OLDER SINGLES/COUPLES	35220
4	Budget	RETIREES	28764

In [129…
```
sns.barplot(data=segment_qty,x='LIFESTAGE', y='PROD_QTY', hue='PREMIUM_CUS
```

Out[129]:
```
<AxesSubplot: xlabel='LIFESTAGE', ylabel='PROD_QTY'>
```



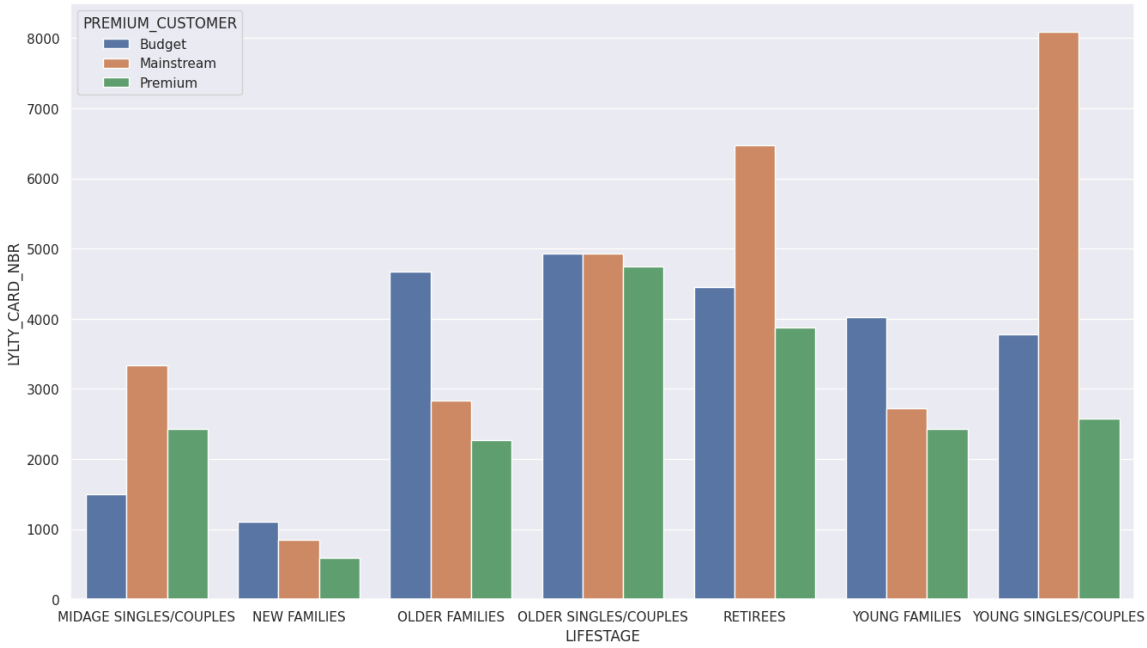
Most quantity sold

- Budget older families
- Mainstream retirees
- Mainstream young singles/couples

In [130…
```
segment_custs = merged_df.groupby(['PREMIUM_CUSTOMER','LIFESTAGE'])['LYLTY
```

In [133…
```
sns.barplot(data=segment_custs,x='LIFESTAGE',y='LYLTY_CARD_NBR',hue='PREMI
```

Out[133]: <AxesSubplot: xlabel='LIFESTAGE', ylabel='LYLTY_CARD_NBR'>



Most number of customers

- Mainstream young singles/couples
- Mainstream retirees

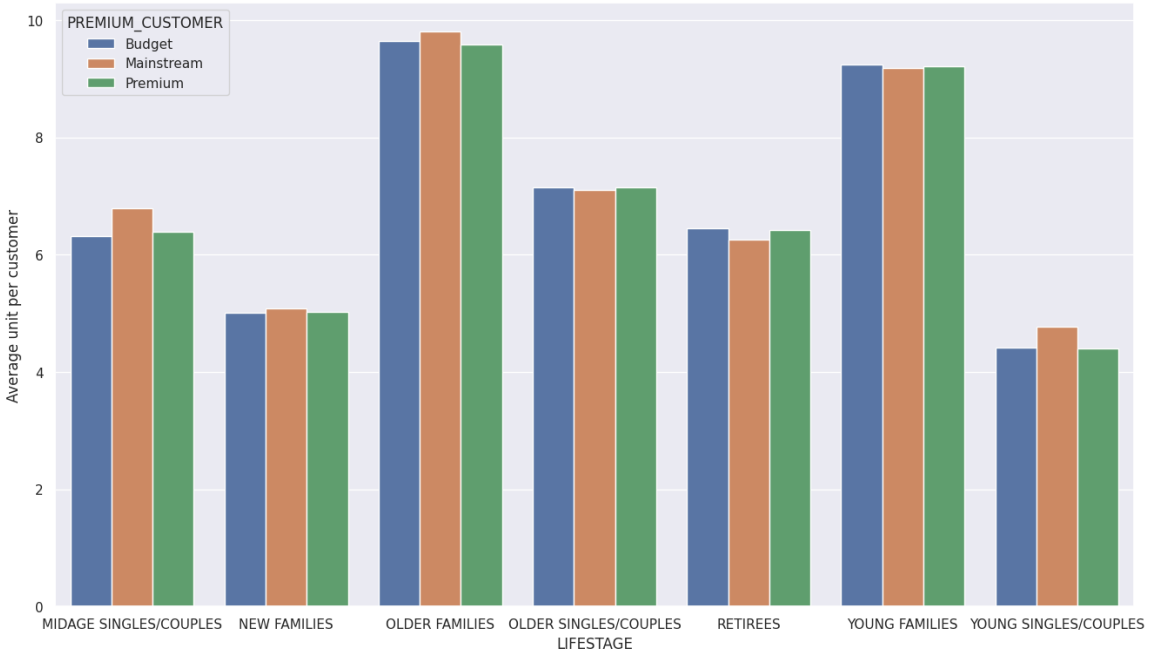
In [142…
```
segment_avg_units = merged_df.groupby(['PREMIUM_CUSTOMER','LIFESTAGE'])['
```

In [146…
```
segment_avg_units = pd.DataFrame(segment_avg_units,columns=['Average unit
```

In [148…
```
segment_avg_units.reset_index(inplace=True)
```

In [149…
```
sns.barplot(data=segment_avg_units,x='LIFESTAGE',y='Average unit per custo
```

Out[149]: <AxesSubplot: xlabel='LIFESTAGE', ylabel='Average unit per customer'>



Old and young families buy more per customer

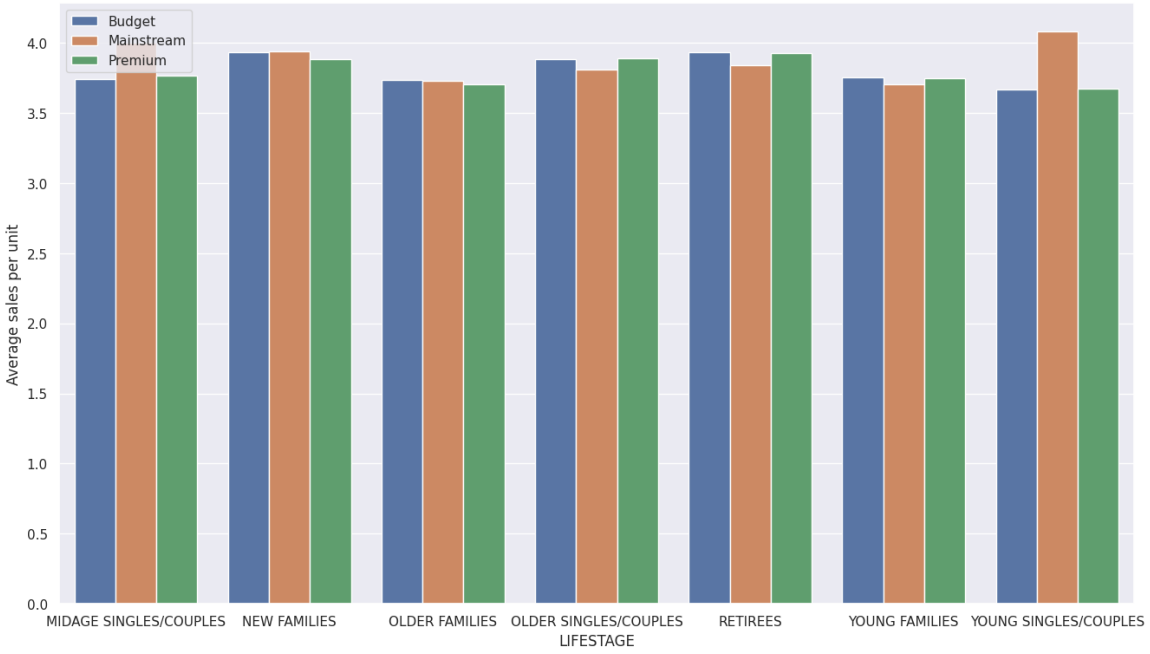
In [154…
```
segment_avg_price = merged_df.groupby(['PREMIUM_CUSTOMER','LIFESTAGE'])['
```

In [155…
```
segment_avg_price = pd.DataFrame(segment_avg_price,columns=['Average sales
```

In [156…
```
segment_avg_price.reset_index(inplace=True)
```

In [159…
```
sns.barplot(data=segment_avg_price,x='LIFESTAGE',y='Average sales per unit
plt.legend(loc='upper left')
```

Out[159]: <matplotlib.legend.Legend at 0x7efc3484c880>



Mainstream midage and young singles/couples are more willing to pay more per pack.

Affinity

In [47]:
```
tgt_custs = merged_df.loc[(merged_df['LIFESTAGE'] == 'YOUNG SINGLES/COUPL
ntgt_custs = merged_df.loc[(merged_df['LIFESTAGE'] != 'YOUNG SINGLES/COUP
```

Brand

In [61]:
```
tgt_brand = tgt_custs.loc[:,['BRAND','PROD_QTY']]
tgt_prodqty  = tgt_brand['PROD_QTY'].sum()
tgt_brand = tgt_brand.groupby('BRAND')['PROD_QTY'].sum().reset_index()
tgt_brand.loc[:,'TGT_Brand_Affinity'] = tgt_brand['PROD_QTY'] / tgt_prodqt
```

In [65]:
```
ntgt_brand = ntgt_custs.loc[:,['BRAND','PROD_QTY']]
ntgt_prodqty  = ntgt_brand['PROD_QTY'].sum()
ntgt_brand = ntgt_brand.groupby('BRAND')['PROD_QTY'].sum().reset_index()
ntgt_brand.loc[:,'NTGT_Brand_Affinity'] = ntgt_brand['PROD_QTY'] / ntgt_p
```

In [71]:
```
merged_brand_affinity = pd.merge(tgt_brand,ntgt_brand,left_index=True,righ
merged_brand_affinity = merged_brand_affinity[['BRAND_x','TGT_Brand_Affin
merged_brand_affinity.rename(columns={'BRAND_x':'BRAND'},inplace=True)
```

In [73]:
```
merged_brand_affinity['Brand Affinity'] = merged_brand_affinity['TGT_Bran
```

In [75]:
```
merged_brand_affinity.sort_values(by='Brand Affinity',ascending=False).hea
```

Out[75]:

	BRAND	TGT_Brand_Affinity	NTGT_Brand_Affinity	Brand Affinity
20	Tyrrells	0.029587	0.023968	1.234454
5	Dorito	0.014729	0.011986	1.228873
19	Twisties	0.043306	0.035355	1.224877
10	Kettle	0.185649	0.155243	1.195863
18	Tostitos	0.042581	0.035744	1.191269

Mainstream young couples/singles are more likely to buy Tyrrells compared to other brand.

Product size

In [76]:
```
tgt_size = tgt_custs.loc[:,['PROD_SIZE','PROD_QTY']]
tgt_prodqty = tgt_size['PROD_QTY'].sum()
tgt_size = tgt_size.groupby('PROD_SIZE')['PROD_QTY'].sum().reset_index()
tgt_size['TGT_PckSize_Affinity'] = tgt_size['PROD_QTY'] / tgt_prodqty
```

In [80]:
```
ntgt_size = ntgt_custs.loc[:,['PROD_SIZE','PROD_QTY']]
ntgt_prodqty = ntgt_size['PROD_QTY'].sum()
ntgt_size = ntgt_size.groupby('PROD_SIZE')['PROD_QTY'].sum().reset_index()
ntgt_size['NTGT_PckSize_Affinity'] = ntgt_size['PROD_QTY'] / ntgt_prodqty
```

In [83]:
```
merged_pcksize_affinity = pd.merge(tgt_size,ntgt_size,left_index=True,righ
merged_pcksize_affinity = merged_pcksize_affinity[['PROD_SIZE_x','TGT_Pck
merged_pcksize_affinity.rename(columns={'PROD_SIZE_x':'PROD_SIZE'},inplace
```

In [86]:
```
merged_pcksize_affinity['Product Size Affinity'] = merged_pcksize_affinity
```

In [88]:
```
merged_pcksize_affinity.sort_values(by='Product Size Affinity',ascending=F
```

Out[88]:

	PROD_SIZE	TGT_PckSize_Affinity	NTGT_PckSize_Affinity	Product Size Affinity
17	270	0.029846	0.023366	1.277295
20	380	0.030156	0.023964	1.258400
19	330	0.057465	0.047511	1.209522
2	110	0.099658	0.083489	1.193675
4	134	0.111980	0.094240	1.188241

More likely to purchase a product of 270 g

Twisties is the only brand that offers 270 g product size

Conclusion

- Sales are highest for: Budget older families, mainstream young singles/couples, mainstream retirees.
- Most of the customers are from mainstream young singles/couples and retirees.
- Mainstream young singles/couples are more likely to pay more per unit. Also more likely to purchase chips from **Tyrrells** and/or chips of size **270 g**.