In [1]:

```
#imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:

```
trans=pd.read_excel("QVI_transaction_data.xlsx_1")
trans.head()
```

Out[2]:

	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 [3]:

```
#transforming date column
trans["DATE"]=pd.to_datetime(trans["DATE"], origin = "1899-12-30",unit="D")
```

In [4]:

```
trans
```

Out[4]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	
...	...	...	...	...	...	...	
264831	2019-03-09	272	272319	270088	89	Kettle Sweet Chilli And Sour Cream 175g	
264832	2018-08-13	272	272358	270154	74	Tostitos Splash Of Lime 175g	
264833	2018-11-06	272	272379	270187	51	Doritos Mexicana 170g	
264834	2018-12-27	272	272379	270188	42	Doritos Corn Chip Mexican Jalapeno 150g	
264835	2018-09-22	272	272380	270189	74	Tostitos Splash Of Lime 175g	

264836 rows × 8 columns

In [5]:

```
trans.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  datetime64[ns]
 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: datetime64[ns](1), float64(1), int64(5), object(1)
memory usage: 16.2+ MB
```

In [6]:

```
trans.describe()
```

Out[6]:

	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_QTY	TOT
count	264836.00000	2.648360e+05	2.648360e+05	264836.000000	264836.000000	26483
mean	135.08011	1.355495e+05	1.351583e+05	56.583157	1.907309	
std	76.78418	8.057998e+04	7.813303e+04	32.826638	0.643654	
min	1.00000	1.000000e+03	1.000000e+00	1.000000	1.000000	
25%	70.00000	7.002100e+04	6.760150e+04	28.000000	2.000000	
50%	130.00000	1.303575e+05	1.351375e+05	56.000000	2.000000	
75%	203.00000	2.030942e+05	2.027012e+05	85.000000	2.000000	
max	272.00000	2.373711e+06	2.415841e+06	114.000000	200.000000	65

In [7]:

```
#number of nulls in each column
trans.isna().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]:

```
#Categorise Numeric and Categorical Data
```

In [9]:

```
trans_numerical = list(trans._get_numeric_data().columns)
trans_numerical
```

Out[9]:

```
['STORE_NBR', 'LYLTY_CARD_NBR', 'TXN_ID', 'PROD_NBR', 'PROD_QTY', 'TOT_SAL
ES']
```

In [10]:

```
trans_cat= set (trans.columns)-set(trans_numerical)
trans_cat
```

Out[10]:

```
{'DATE', 'PROD_NAME'}
```

In [11]:

```
print('numerical data :\n', list (trans_numerical))
print('categorical data :\n', (trans_cat))
```

```
numerical data :
 ['STORE_NBR', 'LYLTY_CARD_NBR', 'TXN_ID', 'PROD_NBR', 'PROD_QTY', 'TOT_SA
LES']
categorical data :
 {'DATE', 'PROD_NAME'}
```

In [12]:

```
# Extracting pack size from the Product
import re
def find_number(text):
    num = re.findall(r'[0-9]+',text)
    return " ".join(num)
trans['pack_size']=trans['PROD_NAME'].apply(lambda x: find_number(x))
trans
```

Out[12]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	
...	...	...	...	...	...	...	
264831	2019-03-09	272	272319	270088	89	Kettle Sweet Chilli And Sour Cream 175g	
264832	2018-08-13	272	272358	270154	74	Tostitos Splash Of Lime 175g	
264833	2018-11-06	272	272379	270187	51	Doritos Mexicana 170g	
264834	2018-12-27	272	272379	270188	42	Doritos Corn Chip Mexican Jalapeno 150g	
264835	2018-09-22	272	272380	270189	74	Tostitos Splash Of Lime 175g	

264836 rows × 9 columns

In [13]:

```
# Create column for brand names
trans['Brand Name'] = trans['PROD_NAME'].str.split(' ').str[0]
```

In [14]:

```
# Check for any duplication or similar bran
trans['Brand Name'].value_counts()
```

Out[14]:

```
Kettle        41288
Smiths        28860
Pringles      25102
Doritos       24962
Thins         14075
RRD           11894
Infuzions     11057
WW            10320
Cobs           9693
Tostitos       9471
Twisties       9454
Old            9324
Tyrrells       6442
Grain          6272
Natural        6050
Red            5885
Cheezels       4603
CCs            4551
Woolworths     4437
Dorito         3185
Infzns         3144
Smith          2963
Cheetos        2927
Snbts          1576
Burger         1564
GrnWves        1468
Sunbites       1432
NCC            1419
French         1418
Name: Brand Name, dtype: int64
```

In [15]:

```
#Some Brands are similar like RRD and Red Rock Deli , Let's combine them together as the

trans['Brand Name'] = trans['Brand Name'].str.replace('Red','RRD')
trans['Brand Name'] = trans['Brand Name'].str.replace('Woolworths','WW')
trans['Brand Name'] = trans['Brand Name'].str.replace('INFUZIONS','INFZNS')
trans['Brand Name'] = trans['Brand Name'].str.replace('SMITHS','SMITH')
trans['Brand Name'] = trans['Brand Name'].str.replace('SUNBITES','SNBTS')
trans['Brand Name'] = trans['Brand Name'].str.replace('DORITOS','DORITO')
trans['Brand Name'] = trans['Brand Name'].str.replace('GRNWVES','GRAIN')
```

In [16]:

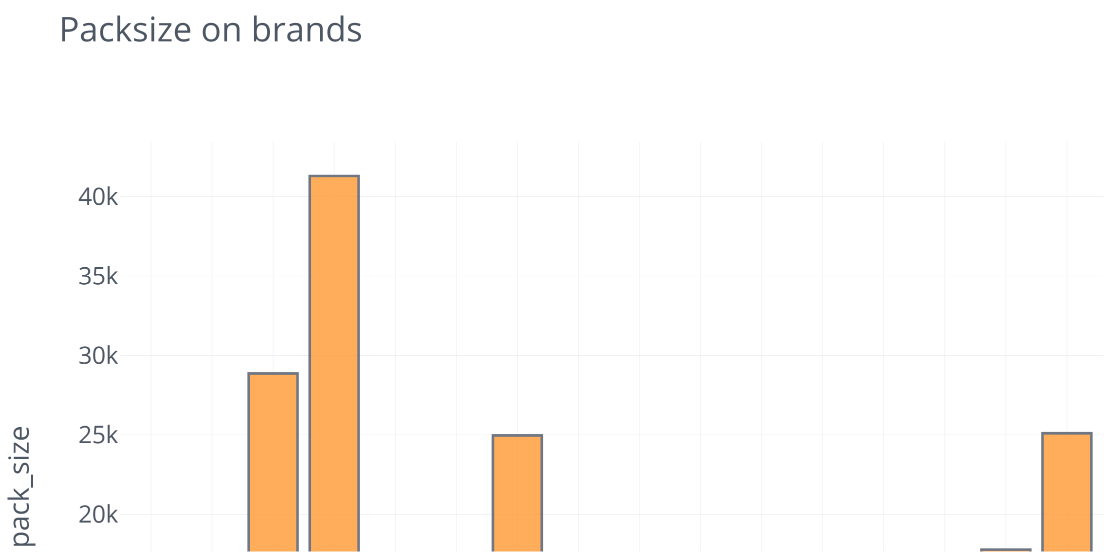
```
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
import plotly.offline as offline
offline.init_notebook_mode()
```

In [17]:

```
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
import plotly.offline as offline
offline.init_notebook_mode()
import cufflinks as cf
cf.go_offline()
```

In [18]:

```
#Histogram for brands
trans['Brand Name'].iplot(kind='hist',xTitle='Brand',yTitle='pack_size',title='Packsize
```

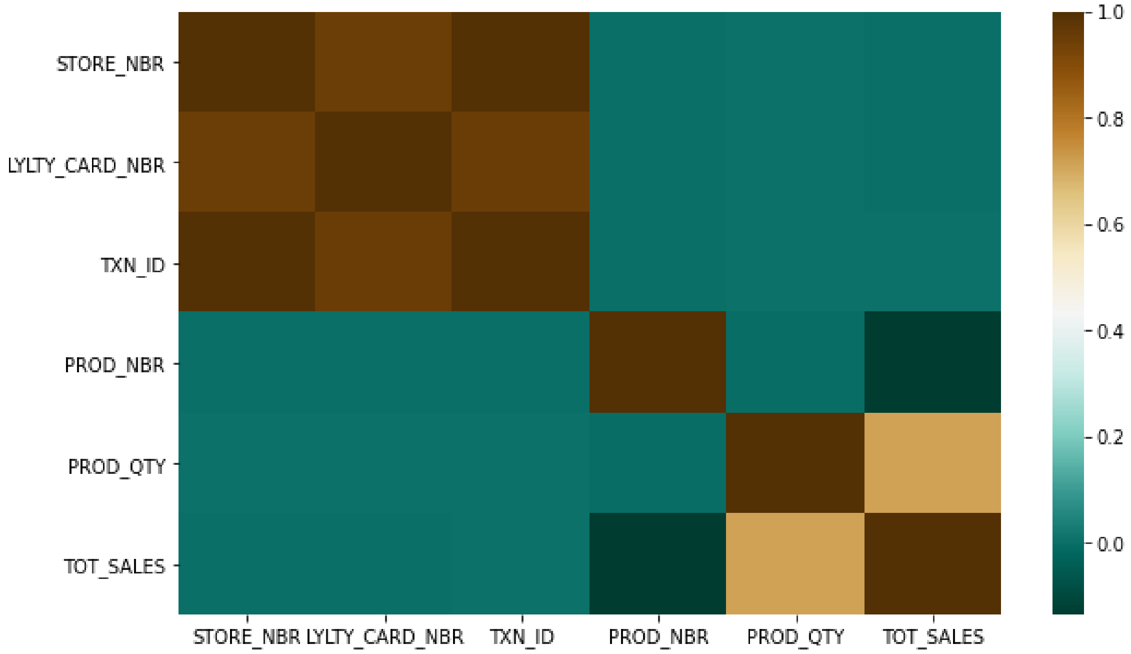


In [19]:

```
#correlation heatmap
plt.figure(figsize=(10,6))
sns.heatmap(trans.corr(),cmap='BrBG_r')
```

Out[19]:

```
<AxesSubplot:>
```



Examining customer data

In [20]:

```
purchase=pd.read_csv("QVI_purchase_behaviour.csv")
purchase.head()
```

Out[20]:

	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 [21]:

```
purchase.isna().sum()
```

Out[21]:

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

In [22]:

```
purchase.describe()
```

Out[22]:

	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

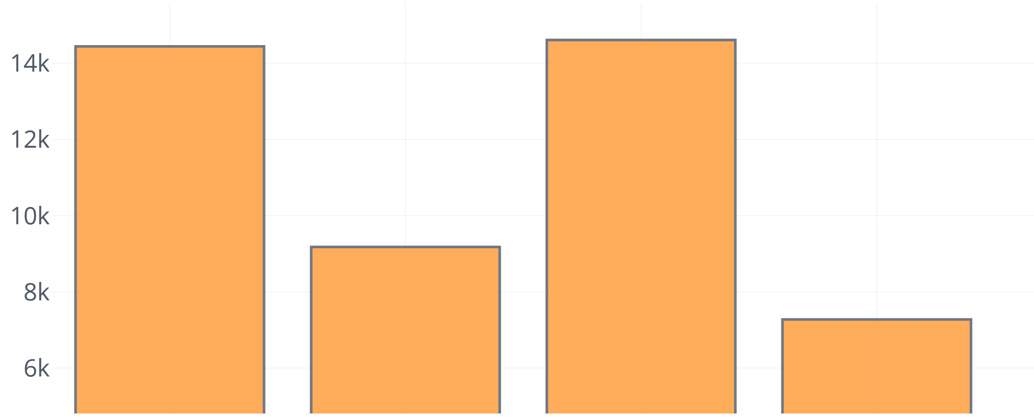
In [23]:

```
purchase.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 [24]:

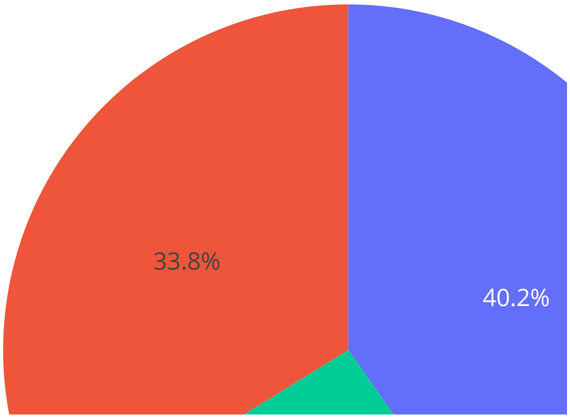
```
purchase['LIFESTAGE'].iplot(kind='hist')
```



In [25]:

```
#Premium customer distribution among customers
import plotly.express as px
fig = px.pie(purchase, values='LYLTY_CARD_NBR', names='PREMIUM_CUSTOMER', title="PREMIUM_
fig.show()
```

PREMIUM_CUSTOMER



joining datasets

In [26]:

```
finaldf=pd.merge(trans,purchase)
finaldf.head(5)
```

Out[26]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	2
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	3
2	2018-11-10	1	1307	346	96	WW Original Stacked Chips 160g	2
3	2019-03-09	1	1307	347	54	CCs Original 175g	1
4	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2

In [27]:

```
# convert to csv file
finaldf.to_csv('Final.csv')
```

In [28]:

```
finaldf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 264836 entries, 0 to 264835
Data columns (total 12 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   DATE              264836 non-null  datetime64[ns]
 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
 8   pack_size         264836 non-null  object
 9   Brand Name        264836 non-null  object
 10  LIFESTAGE         264836 non-null  object
 11  PREMIUM_CUSTOMER  264836 non-null  object
dtypes: datetime64[ns](1), float64(1), int64(5), object(5)
memory usage: 26.3+ MB
```

In [29]:

```
#find the total sales based on premium customer
finaldf[['TOT_SALES','PREMIUM_CUSTOMER']].groupby('PREMIUM_CUSTOMER').sum().sort_values(
```

Out[29]:

PREMIUM_CUSTOMER	TOT_SALES
Mainstream	750744.50
Budget	676211.55
Premium	507458.95

Data analysis on customer segments

In [30]:

```
# How many customers are there in each segment?
#How many chips are bought per customer by segment?
#sales based on brand
```

In [31]:

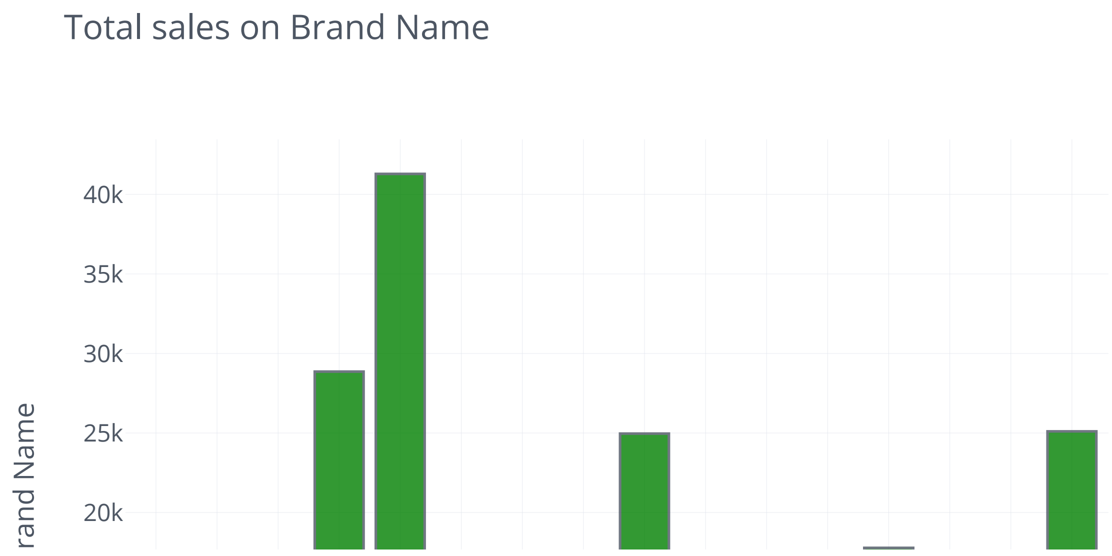
```
#find the total sales based on premium customer
finaldf[['TOT_SALES','Brand Name']].groupby('Brand Name').sum().sort_values('TOT_SALES',
```

Out[31]:

Brand Name	TOT_SALES
Kettle	390239.8
Smiths	210076.8
Doritos	201538.9
Pringles	177655.5
RRD	95046.0
Old	90785.1
Thins	88852.5
Twisties	81522.1
Tostitos	79789.6
Infuzions	76247.6
Cobs	70569.8
Tyrrells	51647.4
WW	49343.6
Grain	43048.8
Dorito	40352.0
Cheezels	40029.9
Natural	34272.0
Infzns	22800.0
CCs	18078.9
Cheetos	16884.5
Smith	14583.4
GrnWves	8568.4
NCC	8046.0
French	7929.0
Burger	6831.0
Snbts	5076.2
Sunbites	4600.2

In [32]:

```
finaldf['Brand Name'].iplot(kind='hist',xTitle='TOT_SALES',yTitle='Brand Name',title='To
```



In [33]:

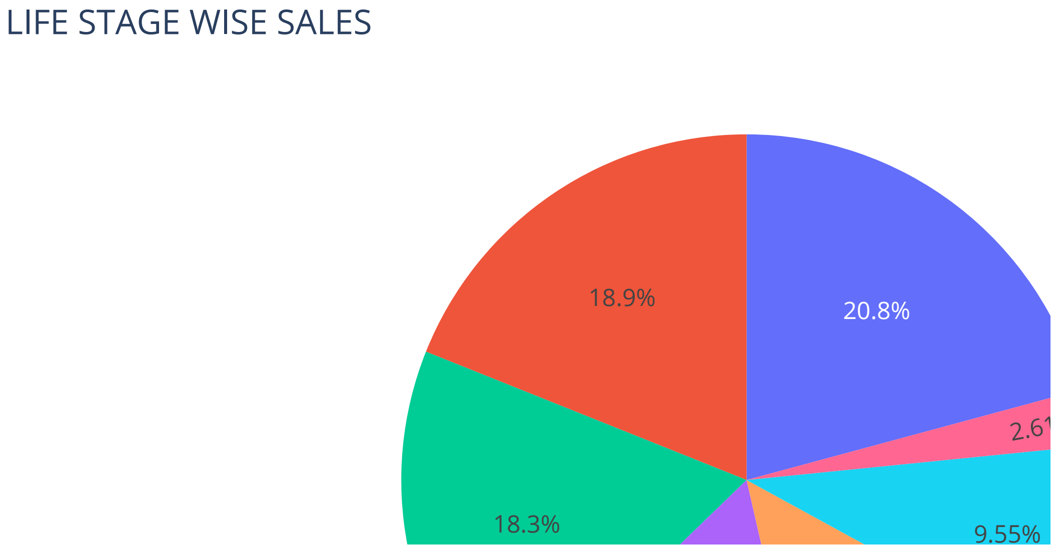
```
finaldf[['TOT_SALES','LIFESTAGE']].groupby('LIFESTAGE').sum().sort_values('TOT_SALES',as
```

Out[33]:

	TOT_SALES
LIFESTAGE	
OLDER SINGLES/COUPLES	402426.75
RETIREES	366470.90
OLDER FAMILIES	353767.20
YOUNG FAMILIES	316160.10
YOUNG SINGLES/COUPLES	260405.30
MIDAGE SINGLES/COUPLES	184751.30
NEW FAMILIES	50433.45

In [34]:

```
fig = px.pie(finaldf, values='TOT_SALES', names='LIFESTAGE', title="LIFE STAGE WISE SALE
fig.show()
```



In [40]:

```
#number of customers are from lifestage and premium customer
NumofCus= pd.DataFrame(finaldf.groupby(['PREMIUM_CUSTOMER', 'LIFESTAGE']).LYLTY_CARD_NBR
NumofCus.rename(columns = {'LYLTY_CARD_NBR': 'Number of Customers'}, inplace = True)
```

In [41]:

```
NumofCus
```

Out[41]:

PREMIUM_CUSTOMER	LIFESTAGE	Number of Customers
Budget	MIDAGE SINGLES/COUPLES	1504
	NEW FAMILIES	1112
	OLDER FAMILIES	4675
	OLDER SINGLES/COUPLES	4929
	RETIREES	4454
	YOUNG FAMILIES	4017
	YOUNG SINGLES/COUPLES	3779
Mainstream	MIDAGE SINGLES/COUPLES	3340
	NEW FAMILIES	849
	OLDER FAMILIES	2831
	OLDER SINGLES/COUPLES	4930
	RETIREES	6479
	YOUNG FAMILIES	2728
	YOUNG SINGLES/COUPLES	8088
Premium	MIDAGE SINGLES/COUPLES	2431
	NEW FAMILIES	588
	OLDER FAMILIES	2274
	OLDER SINGLES/COUPLES	4750
	RETIREES	3872
	YOUNG FAMILIES	2433
	YOUNG SINGLES/COUPLES	2574

In [37]:

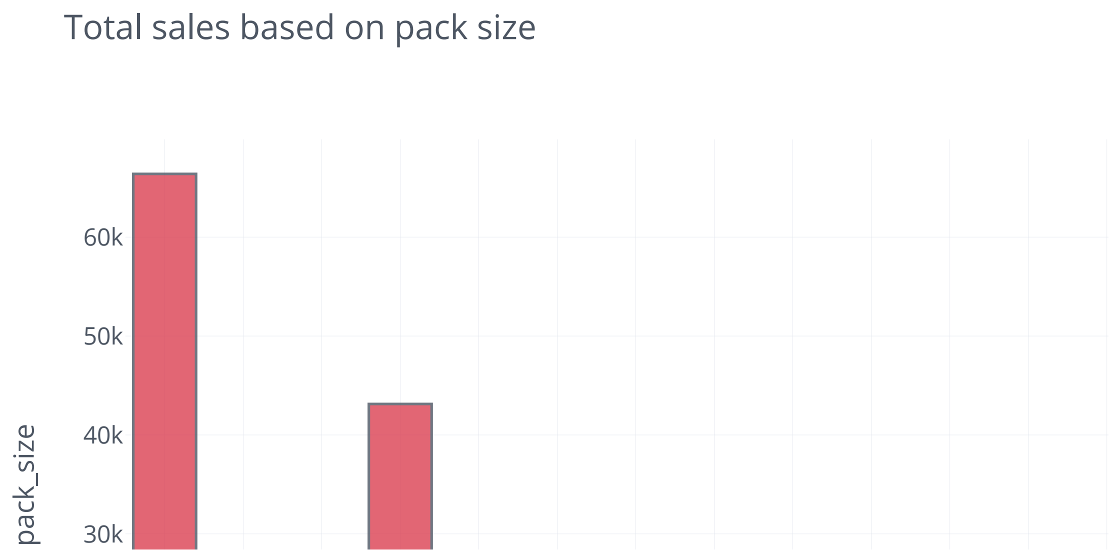
```
finaldf[['TOT_SALES','pack_size']].groupby('pack_size').sum().sort_values('TOT_SALES',as
```

Out[37]:

	TOT_SALES
pack_size	
125	5733.0
220	6831.0
70	6852.0
180	8568.4
90	9676.4
160	10647.6
190	14412.9
200	16007.5
135	26090.4
250	26096.7
210	43048.8
270	55425.4
380	76719.6
165	101360.6
300	113330.6
330	136794.3
170	146673.0
110	162765.4
134	177655.5
150	304288.5
175	485437.4

In [38]:

```
finaldf['pack_size'].iplot(kind='hist',xTitle='TOT_SALES',yTitle='pack_size',title='Tota
```



In [39]:

```
# convert to csv file
finaldf.to_csv('Final.csv')
```

CONCLUSION

After analyzing the data, we found that certain product categories and time periods drove higher sales. We identified distinct customer segments with unique preferences and behaviors. We recommend tailoring marketing strategies to these segments. Packet sizes appear to align with customer preferences. In summary, data-driven decisions can improve business performance, and ongoing monitoring is essential.

In [ ]:

```

```