In [1]:
```
import pandas as pd
import numpy  as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [24]:
```
td=pd.read_csv("QVI_data.csv")
```

In [25]:
```
td.head()
```

Out[25]:

	LYLTY_CARD_NBR	DATE	STORE_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY
0	1000	2018-10-17	1	1	5	Natural Chip Compny SeaSalt175g	2
1	1002	2018-09-16	1	2	58	Red Rock Deli Chikn&Garlic Aioli 150g	1
2	1003	2019-03-07	1	3	52	Grain Waves Sour Cream&Chives 210G	1
3	1003	2019-03-08	1	4	106	Natural ChipCo Hony Soy Chckn175g	1
4	1004	2018-11-02	1	5	96	WW Original Stacked Chips 160g	1

In [26]:
```
td.describe()
```

Out[26]:

	LYLTY_CARD_NBR	STORE_NBR	TXN_ID	PROD_NBR	PROD_QTY	TO
count	2.648340e+05	264834.000000	2.648340e+05	264834.000000	264834.000000	2648
mean	1.355488e+05	135.079423	1.351576e+05	56.583554	1.905813	
std	8.057990e+04	76.784063	7.813292e+04	32.826444	0.343436	
min	1.000000e+03	1.000000	1.000000e+00	1.000000	1.000000	
25%	7.002100e+04	70.000000	6.760050e+04	28.000000	2.000000	
50%	1.303570e+05	130.000000	1.351365e+05	56.000000	2.000000	
75%	2.030940e+05	203.000000	2.026998e+05	85.000000	2.000000	
max	2.373711e+06	272.000000	2.415841e+06	114.000000	5.000000	2

In [27]:
```
td.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264834 entries, 0 to 264833
Data columns (total 12 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   LYLTY_CARD_NBR    264834 non-null  int64  
 1   DATE              264834 non-null  object 
 2   STORE_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
 8   PACK_SIZE         264834 non-null  int64  
 9   BRAND             264834 non-null  object 
 10  LIFESTAGE         264834 non-null  object 
 11  PREMIUM_CUSTOMER  264834 non-null  object 
dtypes: float64(1), int64(6), object(5)
memory usage: 24.2+ MB
```

CHANGING THE DATA TYPE

In [57]:
```
td.DATE=pd.to_datetime(td.DATE)
td["YEARMONTH"] = td["DATE"].dt.strftime("%Y%m").astype("int")
```

In [58]:
```
td.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264834 entries, 0 to 264833
Data columns (total 13 columns):
 #   Column            Non-Null Count   Dtype         
---  ------            --------------   -----         
 0   LYLTY_CARD_NBR    264834 non-null  int64         
 1   DATE              264834 non-null  datetime64[ns]
 2   STORE_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       
 8   PACK_SIZE         264834 non-null  int64         
 9   BRAND             264834 non-null  object        
 10  LIFESTAGE         264834 non-null  object        
 11  PREMIUM_CUSTOMER  264834 non-null  object        
 12  YEARMONTH         264834 non-null  int32         
dtypes: datetime64[ns](1), float64(1), int32(1), int64(6), object(4)
memory usage: 25.3+ MB
```

In [31]:
```
df1=td[td.STORE_NBR==77]
```

In [32]:
```
df2=td[td.STORE_NBR==88]
```

In [33]:
```
df3=td[td.STORE_NBR==86]
```

ANALYSIS FOR STORE NUMBER 77

In [34]:
```
df1.describe()
```

Out[34]:

	LYLTY_CARD_NBR	STORE_NBR	TXN_ID	PROD_NBR	PROD_QTY	TOT_SALES
count	5.630000e+02	563.0	563.000000	563.000000	563.000000	563.000000
mean	1.572879e+05	77.0	80924.701599	57.481350	1.548845	5.399645
std	4.173978e+05	0.0	29931.642675	32.618701	0.532580	2.619975
min	7.700000e+04	77.0	74910.000000	1.000000	1.000000	1.500000
25%	7.712750e+04	77.0	75053.500000	29.000000	1.000000	3.400000
50%	7.726300e+04	77.0	75196.000000	59.000000	2.000000	5.200000
75%	7.739450e+04	77.0	75337.500000	86.000000	2.000000	7.200000
max	2.330501e+06	77.0	236780.000000	114.000000	5.000000	25.500000

In [35]:
```
Total_Sales_77=sum(df1.TOT_SALES)
```

In [36]:
```
Total_Sales_77
```

Out[36]: 3040.0000000000005

In [37]:
```
Total_Cust_77=df1.TXN_ID.count()
```

In [38]:
```
Total_Cust_77
```

Out[38]: 563

In [39]:
```
Avg_Trx_77=Total_Sales_77/Total_Cust_77
```

In [40]:
```
Avg_Trx_77
```

Out[40]: 5.399644760213144

ANALYSIS FOR STORE NUMBER 88

In [41]:
```
df2.describe()
```

Out[41]:

	LYLTY_CARD_NBR	STORE_NBR	TXN_ID	PROD_NBR	PROD_QTY	TOT_SALES
count	1.873000e+03	1873.0	1.873000e+03	1873.000000	1873.000000	1873.00000
mean	1.016763e+05	88.0	8.931120e+04	53.515216	1.985051	8.72036
std	1.744924e+05	0.0	5.505014e+04	32.668791	0.203576	1.76394
min	8.800000e+04	88.0	8.622000e+04	2.000000	1.000000	3.25000
25%	8.809600e+04	88.0	8.669600e+04	26.000000	2.000000	7.40000
50%	8.819300e+04	88.0	8.717700e+04	49.000000	2.000000	8.60000
75%	8.828500e+04	88.0	8.765000e+04	78.000000	2.000000	10.20000
max	2.373711e+06	88.0	2.415841e+06	114.000000	5.000000	22.80000

In [42]:
```
Total_Sales_88=sum(df2.TOT_SALES)
```

In [43]:
```
Total_Sales_88
```

Out[43]:
```
16333.249999999965
```

In [45]:
```
Total_Cust_88=df2.TXN_ID.count()
```

In [46]:
```
Total_Cust_88
```

Out[46]:
```
1873
```

In [47]:
```
Avg_Trx_88=Total_Sales_88/Total_Cust_88
```

In [48]:
```
Avg_Trx_88
```

Out[48]:
```
8.720368392952464
```

ANALYSIS FOR STORE NUMBER 86

In [49]:
```
df3.describe()
```

Out[49]:

	LYLTY_CARD_NBR	STORE_NBR	TXN_ID	PROD_NBR	PROD_QTY	TOT_SALE
count	1538.000000	1538.0	1538.000000	1538.000000	1538.000000	1538.0000
mean	87561.135891	86.0	86368.061769	56.516905	1.993498	6.9150
std	9864.423642	0.0	10047.016520	33.248048	0.239190	2.3035
min	86000.000000	86.0	84137.000000	1.000000	1.000000	1.8000
25%	86063.000000	86.0	84533.250000	28.000000	2.000000	5.4000
50%	86126.000000	86.0	84918.500000	56.000000	2.000000	6.6000
75%	86189.000000	86.0	85313.750000	85.000000	2.000000	8.8000
max	155510.000000	86.0	155718.000000	114.000000	5.000000	16.8000

In [51]:
```
Total_Sales_86=sum(df3.TOT_SALES)
```

In [52]:
```
Total_Sales_86
```

Out[52]:
```
10635.349999999973
```

In [53]:
```
Total_Cust_86=df3.TXN_ID.count()
```

In [54]:
```
Total_Cust_86
```

Out[54]:
```
1538
```

In [55]:
```
Avg_Trx_86=Total_Sales_86/Total_Cust_86
```

In [56]:
```
Avg_Trx_86
```

Out[56]:
```
6.9150520156046635
```

In [77]:
```
metrics=td.groupby(['STORE_NBR','YEARMONTH']).agg({'TOT_SALES':'sum','LYLTY
metrics['PRICE_PER_UNIT']=metrics['TOT_SALES']/metrics['PROD_QTY']
metrics['CHIP_PER_TXN']=metrics['PROD_QTY']/metrics['TXN_ID']
metrics=metrics.rename(columns={'LYLTY_CARD_NBR':'CUSTOMERS'})
metrics['TXN_PER_CUST']=metrics['TXN_ID']/metrics['CUSTOMERS']
metrics.drop(['TXN_ID'],axis=1,inplace=True)
```

In [78]:
```
full=metrics.copy()
```

In [79]:
```
trial=[]
for i in metrics.index:
    if(i[1]>=201902):
        if(i[1]<=201904):
            trial.append(metrics.loc[i])
        metrics.drop(i,inplace=True)
trial=pd.DataFrame(trial)
```

In [83]:
```
metrics
```

Out[83]:

STORE_NBR	YEARMONTH	TOT_SALES	CUSTOMERS	PROD_QTY	PRICE_PER_UNIT	CHIP_
	201807	206.9	49	62	3.337097	
	201808	176.1	42	54	3.261111	
1	201809	278.8	59	75	3.717333	
	201810	188.1	44	58	3.243103	
	201811	192.6	46	57	3.378947	
...	...	...	...	...	...	
	201809	304.7	32	71	4.291549	
	201810	430.6	44	99	4.349495	
272	201811	376.2	41	87	4.324138	
	201812	403.9	47	89	4.538202	
	201901	423.0	46	96	4.406250	

1848 rows × 6 columns

In [84]:
```
def calcCorr(store):
    a=[]
    metrix=metrics[['TOT_SALES','CUSTOMERS']]#add metrics as required e.g.
    for i in metrix.index:
        a.append(metrix.loc[store].corrwith(metrix.loc[i[0]]))
    df= pd.DataFrame(a)
    df.index=metrix.index
    df=df.drop_duplicates()
    df.index=[s[0] for s in df.index]
    df.index.name="STORE_NBR"
    return df
```

In [85]:
```
def standardizer(df):
    df=df.abs()
    df['MAGNITUDE']=df.mean(axis=1)
    return df
```

FINDING CONTROL STORE FOR 77

In [86]:
```
corr77=calcCorr(77)
```

In [87]:
```
corr77
```

Out[87]:

STORE_NBR	TOT_SALES	CUSTOMERS
1	0.075218	0.322168
2	-0.263079	-0.572051
3	0.806644	0.834207
4	-0.263300	-0.295639
5	-0.110652	0.370659
...	...	...
268	0.344757	0.369517
269	-0.315730	-0.474293
270	0.315430	-0.131259
271	0.355487	0.019629
272	0.117622	0.223217

266 rows × 2 columns

In [89]:
```
corr77=standardizer(corr77)
corr77
```

Out[89]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
1	0.075218	0.322168	0.198693
2	0.263079	0.572051	0.417565
3	0.806644	0.834207	0.820426
4	0.263300	0.295639	0.279469
5	0.110652	0.370659	0.240655
...	...	...	...
268	0.344757	0.369517	0.357137
269	0.315730	0.474293	0.395011
270	0.315430	0.131259	0.223345
271	0.355487	0.019629	0.187558
272	0.117622	0.223217	0.170420

266 rows × 3 columns

In [90]:
```
corr77=corr77.sort_values(['MAGNITUDE'],ascending=False).dropna()
```

In [91]:
```
corr77
```

Out[91]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
77	1.000000	1.000000	1.000000
233	0.903774	0.990358	0.947066
119	0.867664	0.983267	0.925466
71	0.914106	0.754817	0.834461
3	0.806644	0.834207	0.820426
...	...	...	...
256	0.014245	0.047863	0.031054
159	0.001655	0.054404	0.028030
260	0.016618	0.027446	0.022032
194	0.010182	0.032053	0.021117
166	0.005875	0.012896	0.009386

263 rows × 3 columns

LET US ASSUME THAT STORE 233 IS CONTROL STORE OF 77

In [113]:
```
from scipy.stats import ks_2samp,ttest_ind,ttest_rel,t
```

In [116]:
```
a=[]
for x in metrics.columns:
    a.append(ks_2samp(metrics.loc[77][x], metrics.loc[233][x]))
a=pd.DataFrame(a,index=metrics.columns)
```

```
C:\Users\Gaurang\AppData\Local\Temp\ipykernel_4176\2415427399.py:3: Runtim
eWarning: ks_2samp: Exact calculation unsuccessful. Switching to method=as
ymp.
  a.append(ks_2samp(metrics.loc[77][x], metrics.loc[233][x]))
```

In [117]:
```
a
```

Out[117]:

	statistic	pvalue
TOT_SALES	0.285714	0.962704
CUSTOMERS	0.142857	0.999961
PROD_QTY	0.285714	0.962704
PRICE_PER_UNIT	0.285714	0.962704
CHIP_PER_TXN	0.285714	0.962704
TXN_PER_CUST	0.428571	0.575175

SINCE P-VALUE IS GREATER THAN 0.05. THEREFORE,STORE 233 CAN BE TERMED AS CONTROL STORE FOR STORE 77

In [ ]:

FINDING CONTROL STORE FOR 88

In [99]:
```
corr88=calcCorr(88)
```

In [100]:
```
corr88
```

Out[100]:

STORE_NBR	TOT_SALES	CUSTOMERS
1	0.813636	0.305334
2	-0.067927	-0.452379
3	-0.507847	0.522884
4	-0.745566	-0.361503
5	0.190330	-0.025320
...	...	...
268	-0.021429	0.672672
269	-0.172578	-0.274781
270	-0.723272	-0.103032
271	-0.103037	-0.018831
272	-0.772772	0.026909

265 rows × 2 columns

In [101]:
```
corr88=standardizer(corr88)
corr88
```

Out[101]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
1	0.813636	0.305334	0.559485
2	0.067927	0.452379	0.260153
3	0.507847	0.522884	0.515365
4	0.745566	0.361503	0.553534
5	0.190330	0.025320	0.107825
...	...	...	...
268	0.021429	0.672672	0.347050
269	0.172578	0.274781	0.223679
270	0.723272	0.103032	0.413152
271	0.103037	0.018831	0.060934
272	0.772772	0.026909	0.399841

265 rows × 3 columns

In [102]:
```
corr88
```

Out[102]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
1	0.813636	0.305334	0.559485
2	0.067927	0.452379	0.260153
3	0.507847	0.522884	0.515365
4	0.745566	0.361503	0.553534
5	0.190330	0.025320	0.107825
...	...	...	...
268	0.021429	0.672672	0.347050
269	0.172578	0.274781	0.223679
270	0.723272	0.103032	0.413152
271	0.103037	0.018831	0.060934
272	0.772772	0.026909	0.399841

265 rows × 3 columns

In [103]:
```
corr88=corr88.sort_values(['MAGNITUDE'],ascending=False).dropna()
```

In [104]:
```
corr88
```

Out[104]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
88	1.000000	1.000000	1.000000
178	0.731857	0.939466	0.835661
14	0.698557	0.942976	0.820767
133	0.735407	0.835426	0.785417
204	0.885774	0.550263	0.718018
...	...	...	...
271	0.103037	0.018831	0.060934
177	0.084074	0.005568	0.044821
170	0.027262	0.028583	0.027923
137	0.005058	0.039985	0.022521
194	0.008321	0.009504	0.008912

263 rows × 3 columns

LET US ASSUME THAT STORE 178 IS CONTROL STORE OF 88

In [124]:
```
b=[]
for x in metrics.columns:
    b.append(ks_2samp(metrics.loc[88][x], metrics.loc[178][x]))
b=pd.DataFrame(c,index=metrics.columns)
```

In [122]:
```
b
```

Out[122]:

	statistic	pvalue
TOT_SALES	0.285714	0.962704
CUSTOMERS	0.285714	0.962704
PROD_QTY	0.285714	0.962704
PRICE_PER_UNIT	0.428571	0.575175
CHIP_PER_TXN	0.428571	0.575175
TXN_PER_CUST	0.428571	0.575175

SINCE P-VALUE IS GREATER THAN 0.05. THEREFORE,STORE 178 CAN BE TERMED AS CONTROL STORE FOR STORE 88

In [ ]:

FINDING CONTROL STORE FOR 86

In [108]:
```
corr86=calcCorr(86)
```

In [109]:
```
corr86
```

Out[109]:

STORE_NBR	TOT_SALES	CUSTOMERS
1	0.445632	0.485831
2	-0.403835	-0.086161
3	-0.261284	-0.353786
4	-0.039035	-0.169608
5	0.235159	-0.253229
...	...	...
268	-0.452182	-0.034273
269	0.697055	-0.098587
270	-0.730679	-0.767267
271	0.527637	0.267393
272	0.004926	-0.353815

266 rows × 2 columns

In [110]:
```
corr86=standardizer(corr86)
corr86
```

Out[110]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
1	0.445632	0.485831	0.465731
2	0.403835	0.086161	0.244998
3	0.261284	0.353786	0.307535
4	0.039035	0.169608	0.104322
5	0.235159	0.253229	0.244194
...	...	...	...
268	0.452182	0.034273	0.243228
269	0.697055	0.098587	0.397821
270	0.730679	0.767267	0.748973
271	0.527637	0.267393	0.397515
272	0.004926	0.353815	0.179371

266 rows × 3 columns

In [111]:
```
corr86=corr86.sort_values(['MAGNITUDE'],ascending=False).dropna()
corr86
```

Out[111]:

STORE_NBR	TOT_SALES	CUSTOMERS	MAGNITUDE
86	1.000000	1.000000	1.000000
155	0.877882	0.942876	0.910379
23	0.784698	0.943559	0.864128
120	0.872693	0.815097	0.843895
114	0.734415	0.855339	0.794877
...	...	...	...
91	0.019027	0.041271	0.030149
17	0.029793	0.030039	0.029916
131	0.028487	0.031142	0.029815
219	0.046653	0.004999	0.025826
234	0.010509	0.040306	0.025407

263 rows × 3 columns

LET US ASSUME THAT STORE 155 IS CONTROL STORE OF 86

In [119]:
```
c=[]
for x in metrics.columns:
    c.append(ks_2samp(metrics.loc[86][x], metrics.loc[155][x]))
c=pd.DataFrame(c,index=metrics.columns)
```

In [120]:
```
c
```

Out[120]:

	statistic	pvalue
TOT_SALES	0.285714	0.962704
CUSTOMERS	0.285714	0.962704
PROD_QTY	0.285714	0.962704
PRICE_PER_UNIT	0.428571	0.575175
CHIP_PER_TXN	0.428571	0.575175
TXN_PER_CUST	0.428571	0.575175

SINCE P-VALUE IS GREATER THAN 0.05. THEREFORE,STORE 155 CAN BE TERMED AS CONTROL STORE FOR STORE 86