Task 2: Experimentation and Uplift Testing

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

In [2]:
```
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 [3]:
```
#reading data
data=pd.read_csv("QVI_data.csv");
data.head()
```

Out[3]:

	LYLTY_CARD_NBR	DATE	STORE_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT
0	1000	2018-10-17	1	1	5	Natural Chip Compny SeaSalt175g	2.0	
1	1002	2018-09-16	1	2	58	Red Rock Deli Chikn&Garlic Aioli 150g	1.0	
2	1003	2019-03-07	1	3	52	Grain Waves Sour Cream&Chives 210G	1.0	
3	1003	2019-03-08	1	4	106	Natural ChipCo Hony Soy Chckn175g	1.0	
4	1004	2018-11-02	1	5	96	WW Original Stacked Chips 160g	1.0	

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 135412 entries, 0 to 135411
Data columns (total 12 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   LYLTY_CARD_NBR    135412 non-null  int64  
 1   DATE              135412 non-null  object 
 2   STORE_NBR         135412 non-null  int64  
 3   TXN_ID            135412 non-null  int64  
 4   PROD_NBR          135412 non-null  int64  
 5   PROD_NAME         135412 non-null  object 
 6   PROD_QTY          135411 non-null  float64
 7   TOT_SALES         135411 non-null  float64
 8   PACK_SIZE         135411 non-null  float64
 9   BRAND             135411 non-null  object 
 10  LIFESTAGE         135411 non-null  object 
 11  PREMIUM_CUSTOMER  135411 non-null  object 
dtypes: float64(3), int64(4), object(5)
memory usage: 12.4+ MB
```

In [5]:
```
data['DATE']=pd.to_datetime(data['DATE'])
```

1. Select control stores

The client has selected store numbers 77, 86 and 88 as trial stores and want control stores to be established stores that are operational for the entire observation period.
We would want to match trial stores to control stores that are similar to the trial store prior to the trial period of Feb 2019 in terms of :

- Monthly overall sales revenue
- Monthly number of customers
- Monthly number of transactions per customer

In [8]:
```
# Add a new 'MONTH_ID' column in the data with the format yyyymm
data['MONTH_ID']=[s.year*100+s.month for s in data['DATE']]
```

In [9]:
```
data
```

Out[9]:

	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	2.0
1	1002	2018-09-16	1	2	58	Red Rock Deli Chikn&Garlic Aioli 150g	1.0
2	1003	2019-03-07	1	3	52	Grain Waves Sour Cream&Chives 210G	1.0
3	1003	2019-03-08	1	4	106	Natural ChipCo Hony Soy Chckn175g	1.0
4	1004	2018-11-02	1	5	96	WW Original Stacked Chips 160g	1.0
...	...	...	...	...	...	...	...
135407	134326	2018-09-06	134	138194	65	Old El Paso Salsa Dip Chnky Tom Ht300g	2.0
135408	134326	2018-09-07	134	138195	109	Pringles Barbeque 134g	2.0
135409	134326	2019-04-07	134	138196	25	Pringles SourCream Onion 134g	2.0
135410	134327	2018-08-30	134	138197	50	Tostitos Lightly Salted 175g	2.0
135411	134327	2019-01-23	134	138198	63	Kettle	NaN

135412 rows × 14 columns

In [55]:
```
data.drop('Month_id',axis=1)
```

Out[55]:

	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	2.0
1	1002	2018-09-16	1	2	58	Red Rock Deli Chikn&Garlic Aioli 150g	1.0
2	1003	2019-03-07	1	3	52	Grain Waves Sour Cream&Chives 210G	1.0
3	1003	2019-03-08	1	4	106	Natural ChipCo Hony Soy Chckn175g	1.0
4	1004	2018-11-02	1	5	96	WW Original Stacked Chips 160g	1.0
...	...	...	...	...	...	...	...
135407	134326	2018-09-06	134	138194	65	Old El Paso Salsa Dip Chnky Tom Ht300g	2.0
135408	134326	2018-09-07	134	138195	109	Pringles Barbeque 134g	2.0
135409	134326	2019-04-07	134	138196	25	Pringles SourCream Onion 134g	2.0
135410	134327	2018-08-30	134	138197	50	Tostitos Lightly Salted 175g	2.0
135411	134327	2019-01-23	134	138198	63	Kettle	NaN

135412 rows × 13 columns

1) Create the metrics

In [14]:
```
#metrics=data.groupby(['STORE_NBR','MONTH_ID']).agg({'TOT_SALES':'sum','LYLTY_C
#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 [19]:

```
# Monthly total sales
M_TOT_SALES = data.groupby(["STORE_NBR","MONTH_ID"])["TOT_SALES"].sum()

# Monthly customer counts
M_CUS_COUNTS = data.groupby(["STORE_NBR","MONTH_ID"])["LYLTY_CARD_NBR"].nunique

# Monthly transactions per customer
M_TXN_CUS = data.groupby(["STORE_NBR","MONTH_ID"])["TXN_ID"].nunique()/M_CUS_CO

# Monthly chips per customer
M_CHIP_CUS = data.groupby(["STORE_NBR","MONTH_ID"])["PROD_QTY"].sum()/M_CUS_COU

# Monthly average price per unit
M_AVG_PRICE_CHIP = M_TOT_SALES/data.groupby(["STORE_NBR","MONTH_ID"])["PROD_QTY

# Combining metrics together
measureOverTime = pd.concat([M_TOT_SALES, M_CUS_COUNTS, M_TXN_CUS, M_CHIP_CUS,
measureOverTime.columns = ["totSales", "Customers", "nTxnPerCust", "nChipsPerTx
measureOverTime = measureOverTime.reset_index()
measureOverTime
```

Out[19]:

	STORE_NBR	MONTH_ID	totSales	Customers	nTxnPerCust	nChipsPerTxn	avgPricePerUn
0	1	201807	206.9	49	1.061224	1.265306	3.33709
1	1	201808	176.1	42	1.023810	1.285714	3.26111
2	1	201809	278.8	59	1.050847	1.271186	3.71733
3	1	201810	188.1	44	1.022727	1.318182	3.24310
4	1	201811	192.6	46	1.021739	1.239130	3.37894
...	...	...	...	...	...	...	.
1549	134	201903	320.2	36	1.055556	2.111111	4.21315
1550	134	201904	462.6	50	1.020000	2.040000	4.53529
1551	134	201905	356.9	41	1.073171	2.024390	4.30000
1552	134	201906	385.8	42	1.071429	2.142857	4.28666
1553	155	201906	16.8	2	1.000000	2.000000	4.20000

1554 rows × 7 columns

2) Divide full observation periods

In [20]:
```
# Stores with full observation periods(12 month)
obs_counts = measureOverTime["STORE_NBR"].value_counts()
full_idx = obs_counts[obs_counts == 12].index
storesWithFullObs = measureOverTime[measureOverTime["STORE_NBR"].isin(full_idx)

# Filter to the pre-trial period (201807 - 201901)
preTrialMeasures = storesWithFullObs[storesWithFullObs["MONTH_ID"] < 201902]
preTrialMeasures.head(8)
```

Out[20]:

	STORE_NBR	MONTH_ID	totSales	Customers	nTxnPerCust	nChipsPerTxn	avgPricePerUnit
0	1	201807	206.9	49	1.061224	1.265306	3.337097
1	1	201808	176.1	42	1.023810	1.285714	3.261111
2	1	201809	278.8	59	1.050847	1.271186	3.717333
3	1	201810	188.1	44	1.022727	1.318182	3.243103
4	1	201811	192.6	46	1.021739	1.239130	3.378947
5	1	201812	189.6	42	1.119048	1.357143	3.326316
6	1	201901	154.8	35	1.028571	1.200000	3.685714
12	2	201807	150.8	39	1.051282	1.179487	3.278261

In [24]:
```
preTrialMeasures=pd.DataFrame(preTrialMeasures)
```

3) Select control stores

In [38]:
```
control_stores=preTrialMeasures[(preTrialMeasures.STORE_NBR!=77 ) & (preTrialMe
control_stores
```

Out[38]:

STORE_NBR	totSales	Customers	nTxnPerCust
1	1386.90	317	7.327967
2	1128.50	272	7.359700
3	7526.15	744	8.209829
4	9127.00	849	8.535253
5	5739.70	651	8.791906
...	...	...	...
130	8248.85	830	8.049786
131	1503.30	322	7.249771
132	271.80	45	7.200000
133	6942.90	782	9.040840
134	2575.05	280	7.447572

124 rows × 3 columns

In [39]:
```
trial_stores=preTrialMeasures[(preTrialMeasures.STORE_NBR==77 ) | (preTrialMeas
trial_stores
```

Out[39]:

STORE_NBR	totSales	Customers	nTxnPerCust
77	1617.80	285	7.401449
86	6026.45	686	8.811943
88	9267.40	870	8.524347

Store 77

In [40]:
```
difference=control_stores.loc[control_stores.corrwith(trial_stores.loc[77], met

difference=(trial_stores.loc[77]-difference).sort_values(by="totSales", ascend:
difference["DIFFERENCE"]=difference["totSales"]-difference["totSales"].mean()
difference.sort_values(by="DIFFERENCE", ascending=False)
```

Out[40]:

STORE_NBR	totSales	Customers	nTxnPerCust	DIFFERENCE
127	1382.5	240.0	0.401449	1280.6
90	-118.6	-6.0	0.081271	-220.5
46	-140.2	-17.0	0.090375	-242.1
50	-277.8	-58.0	-0.036278	-379.7
38	-336.4	-43.0	-0.078112	-438.3

Store 86

In [41]:
```
difference=control_stores.loc[control_stores.corrwith(trial_stores.loc[86], ax:
difference=(trial_stores.loc[86]-difference).sort_values(by="totSales", ascend:
difference["DIFFERENCE"]=difference["totSales"]-difference["totSales"].mean()
difference.sort_values(by="DIFFERENCE", ascending=False)
```

Out[41]:

STORE_NBR	totSales	Customers	nTxnPerCust	DIFFERENCE
19	581.85	68.0	0.844490	506.51
5	286.75	35.0	0.020038	211.41
70	189.05	23.0	-0.131763	113.71
57	-120.95	-13.0	0.045215	-196.29
106	-560.00	-60.0	-0.019804	-635.34

Store 88

In [42]:
```
difference=control_stores.loc[control_stores.corrwith(trial_stores.loc[88], ax

difference=(trial_stores.loc[88]-difference).sort_values(by="totSales", ascend
difference["DIFFERENCE"]=difference["totSales"]-difference["totSales"].mean()
difference.sort_values(by="DIFFERENCE", ascending=False)
```

Out[42]:

STORE_NBR	totSales	Customers	nTxnPerCust	DIFFERENCE
60	1580.90	144.0	0.053033	808.35
75	1303.90	119.0	0.079515	531.35
72	748.90	69.0	0.086400	-23.65
4	140.40	21.0	-0.010906	-632.15
58	88.65	-4.0	0.182870	-683.90

For STORE_NBR 88, we can see that STORE_NBR 165 would be the most suitable control store.

In [44]:
```
trial_stores_one=preTrialMeasures.loc[preTrialMeasures.STORE_NBR.isin([77])].re
trial_stores_two=preTrialMeasures.loc[preTrialMeasures.STORE_NBR.isin([86])].re
trial_stores_three=preTrialMeasures.loc[preTrialMeasures.STORE_NBR.isin([88])]
```

In [45]:
```
control_stores_one=preTrialMeasures.loc[preTrialMeasures.STORE_NBR.isin([46])]
control_stores_two=preTrialMeasures.loc[preTrialMeasures.STORE_NBR.isin([57])]
control_stores_three=preTrialMeasures.loc[preTrialMeasures.STORE_NBR.isin([165]
```

In [46]:
```
stores=pd.concat([trial_stores_one, trial_stores_two, trial_stores_three, contr
stores
```

Out[46]:

	index	STORE_NBR	MONTH_ID	totSales	Customers	nTxnPerCust	nChipsPerTxn	avgPricePe
0	880	77	201807	272.30	46	1.086957	1.695652	3.4
1	881	77	201808	255.50	47	1.021277	1.574468	3.4
2	882	77	201809	206.20	39	1.051282	1.641026	3.2
3	883	77	201810	204.50	37	1.027027	1.405405	3.9
4	884	77	201811	207.60	35	1.057143	1.628571	3.6
5	885	77	201812	267.30	46	1.043478	1.565217	3.7
6	886	77	201901	204.40	35	1.114286	1.857143	3.1
0	977	86	201807	892.20	99	1.272727	2.535354	3.5
1	978	86	201808	710.85	88	1.170455	2.284091	3.5
2	979	86	201809	914.60	103	1.242718	2.504854	3.5
3	980	86	201810	948.40	109	1.266055	2.532110	3.4
4	981	86	201811	877.80	95	1.263158	2.568421	3.5
5	982	86	201812	841.20	98	1.224490	2.448980	3.5
6	983	86	201901	841.40	94	1.372340	2.765957	3.2
0	1001	88	201807	1310.00	129	1.186047	2.372093	4.2
1	1002	88	201808	1289.20	128	1.210938	2.320312	4.3
2	1003	88	201809	1423.00	124	1.266129	2.564516	4.4
3	1004	88	201810	1317.20	120	1.258333	2.566667	4.2
4	1005	88	201811	1382.80	130	1.200000	2.415385	4.4
5	1006	88	201812	1278.80	122	1.172131	2.360656	4.4
6	1007	88	201901	1266.40	117	1.230769	2.495726	4.3
0	519	46	201807	253.00	45	1.066667	1.666667	3.3
1	520	46	201808	240.70	44	1.045455	1.522727	3.5
2	521	46	201809	233.00	41	1.048780	1.731707	3.2
3	522	46	201810	275.10	47	1.042553	1.723404	3.3
4	523	46	201811	273.10	42	1.047619	1.738095	3.7
5	524	46	201812	306.90	50	1.060000	1.700000	3.6
6	525	46	201901	176.20	33	1.000000	1.545455	3.4
0	651	57	201807	839.60	103	1.203883	2.427184	3.3
1	652	57	201808	915.40	102	1.274510	2.441176	3.6
2	653	57	201809	792.80	99	1.171717	2.383838	3.3
3	654	57	201810	965.80	104	1.307692	2.615385	3.5
4	655	57	201811	830.00	100	1.170000	2.340000	3.5
5	656	57	201812	951.00	104	1.259615	2.519231	3.6
6	657	57	201901	852.80	87	1.379310	2.758621	3.5

In [47]:
```
sns.set_style("darkgrid")
figure, axis=plt.subplots(1, 3, figsize=(20, 7))
sns.barplot(x="STORE_NBR", y="totSales", data=stores, ax=axis[0], palette="past
axis[0].set_title("Total Sales")
sns.barplot(x="STORE_NBR", y="Customers", data=stores, ax=axis[1], palette="pas
axis[1].set_title("Total Customers")
sns.barplot(x="STORE_NBR", y="nTxnPerCust", data=stores, ax=axis[2], palette="p
axis[2].set_title("Transactions per Customer")
figure.suptitle("Comparison of the Total Sales, Total Customers, and Transactic
plt.show()
```

```
<ipython-input-47-91cd33bcbeae>:3: FutureWarning:



Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.


<ipython-input-47-91cd33bcbeae>:5: FutureWarning:



Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.


<ipython-input-47-91cd33bcbeae>:7: FutureWarning:



Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.
```

While the other trial stores performed the same as their corresponding control stores, we can see, however, that STORE_NBR 88 slightly out-performed its control store in all attributes. We can also notice that STORE_NBR 86 and 88 show a significant difference in terms of the total sales, but this isn't the case with STORE_NBR 77, whose sales are considerably less.

In [50]:
```
TrialMeasures = storesWithFullObs[storesWithFullObs["MONTH_ID"] < 201902]
TrialMeasures.head(8)
```

Out[50]:

	STORE_NBR	MONTH_ID	totSales	Customers	nTxnPerCust	nChipsPerTxn	avgPricePerUnit
0	1	201807	206.9	49	1.061224	1.265306	3.337097
1	1	201808	176.1	42	1.023810	1.285714	3.261111
2	1	201809	278.8	59	1.050847	1.271186	3.717333
3	1	201810	188.1	44	1.022727	1.318182	3.243103
4	1	201811	192.6	46	1.021739	1.239130	3.378947
5	1	201812	189.6	42	1.119048	1.357143	3.326316
6	1	201901	154.8	35	1.028571	1.200000	3.685714
12	2	201807	150.8	39	1.051282	1.179487	3.278261

In [51]:
```
trial_stores_one=TrialMeasures.loc[TrialMeasures.STORE_NBR.isin([77])].reset_in
trial_stores_two=TrialMeasures.loc[TrialMeasures.STORE_NBR.isin([86])].reset_in
trial_stores_three=TrialMeasures.loc[TrialMeasures.STORE_NBR.isin([88])].reset_
```

In [52]:
```
control_stores_one=TrialMeasures.loc[TrialMeasures.STORE_NBR.isin([46])].reset_
control_stores_two=TrialMeasures.loc[TrialMeasures.STORE_NBR.isin([57])].reset_
control_stores_three=TrialMeasures.loc[TrialMeasures.STORE_NBR.isin([165])].res
```

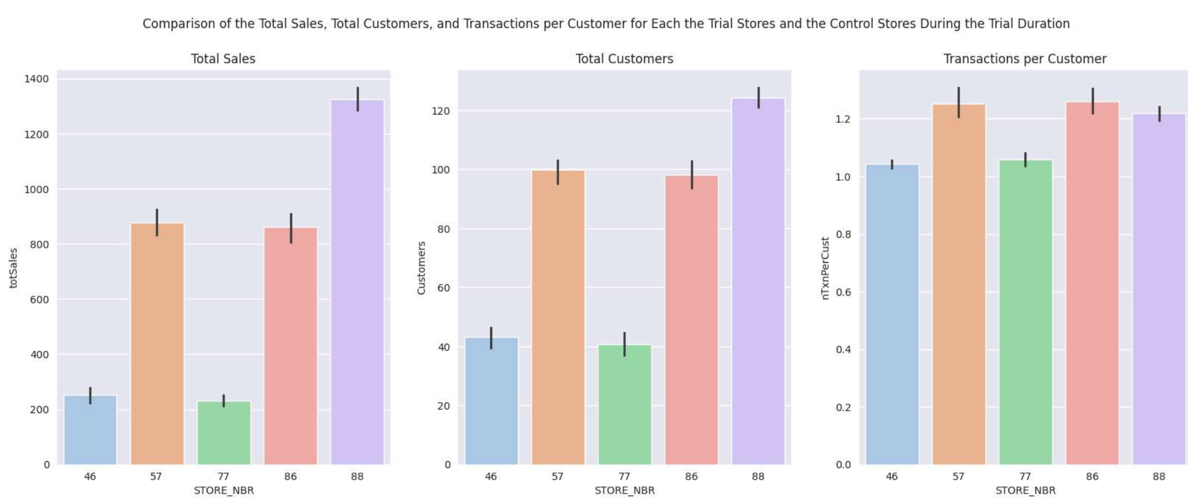
In [53]:
```
stores=pd.concat([trial_stores_one, trial_stores_two, trial_stores_three, contr
stores
```

Out[53]:

	index	STORE_NBR	MONTH_ID	totSales	Customers	nTxnPerCust	nChipsPerTxn	avgPricePe
0	880	77	201807	272.30	46	1.086957	1.695652	3.4
1	881	77	201808	255.50	47	1.021277	1.574468	3.4
2	882	77	201809	206.20	39	1.051282	1.641026	3.2
3	883	77	201810	204.50	37	1.027027	1.405405	3.9
4	884	77	201811	207.60	35	1.057143	1.628571	3.6
5	885	77	201812	267.30	46	1.043478	1.565217	3.7
6	886	77	201901	204.40	35	1.114286	1.857143	3.1
0	977	86	201807	892.20	99	1.272727	2.535354	3.5
1	978	86	201808	710.85	88	1.170455	2.284091	3.5
2	979	86	201809	914.60	103	1.242718	2.504854	3.5
3	980	86	201810	948.40	109	1.266055	2.532110	3.4
4	981	86	201811	877.80	95	1.263158	2.568421	3.5
5	982	86	201812	841.20	98	1.224490	2.448980	3.5
6	983	86	201901	841.40	94	1.372340	2.765957	3.2
0	1001	88	201807	1310.00	129	1.186047	2.372093	4.2
1	1002	88	201808	1289.20	128	1.210938	2.320312	4.3
2	1003	88	201809	1423.00	124	1.266129	2.564516	4.4
3	1004	88	201810	1317.20	120	1.258333	2.566667	4.2
4	1005	88	201811	1382.80	130	1.200000	2.415385	4.4
5	1006	88	201812	1278.80	122	1.172131	2.360656	4.4
6	1007	88	201901	1266.40	117	1.230769	2.495726	4.3
0	519	46	201807	253.00	45	1.066667	1.666667	3.3
1	520	46	201808	240.70	44	1.045455	1.522727	3.5
2	521	46	201809	233.00	41	1.048780	1.731707	3.2
3	522	46	201810	275.10	47	1.042553	1.723404	3.3
4	523	46	201811	273.10	42	1.047619	1.738095	3.7
5	524	46	201812	306.90	50	1.060000	1.700000	3.6
6	525	46	201901	176.20	33	1.000000	1.545455	3.4
0	651	57	201807	839.60	103	1.203883	2.427184	3.3
1	652	57	201808	915.40	102	1.274510	2.441176	3.6
2	653	57	201809	792.80	99	1.171717	2.383838	3.3
3	654	57	201810	965.80	104	1.307692	2.615385	3.5
4	655	57	201811	830.00	100	1.170000	2.340000	3.5
5	656	57	201812	951.00	104	1.259615	2.519231	3.6
6	657	57	201901	852.80	87	1.379310	2.758621	3.5

In [54]:
```
sns.set_style("darkgrid")
figure, axis=plt.subplots(1, 3, figsize=(20, 7))
sns.barplot(x="STORE_NBR", y="totSales", data=stores, ax=axis[0], palette="past
axis[0].set_title("Total Sales")
sns.barplot(x="STORE_NBR", y="Customers", data=stores, ax=axis[1], palette="pas
axis[1].set_title("Total Customers")
sns.barplot(x="STORE_NBR", y="nTxnPerCust", data=stores, ax=axis[2], palette="
axis[2].set_title("Transactions per Customer")
figure.suptitle("Comparison of the Total Sales, Total Customers, and Transactio
plt.show()
```

```
<ipython-input-54-a04d03e567b7>:3: FutureWarning:



Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.


<ipython-input-54-a04d03e567b7>:5: FutureWarning:



Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.


<ipython-input-54-a04d03e567b7>:7: FutureWarning:



Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.
```



We can, notice that STORE_NBR 88 slightly out-performs its control store, STORE_NBR 165, and still remains the best implementation of the trial of all the trial stores. The driver for this seems to be the purchasing customers rather than purchases per customer, as we can see that with the increase in the total customers, there's also an increase in the total sales almost identically, but the transactions per customer seem to be reasonably high for all the trial stores regardless of the total sales.

Conclusion

- While the other trial stores performed the same as their corresponding control stores, we can see, however, that STORE_NBR 88 slightly out-performed its control store, STORE_NBR 165, in all attributes.
- STORE_NBR 86 and 88 show a significant difference in terms of the total sales, but this isn't the case with STORE_NBR 77, which may be because of the way the trial was implemented for it.
- Due to the maximum difference in the total sales of all the trial stores, STORE_NBR 88 remains the best implementation of the trial.
- The driver for the increase in total sales seems to be the purchasing customers rather than purchases per customer — the more the customers, the higher the sales.

In [ ]: