In [1]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
from mpl_toolkits.mplot3d import axis3d
from IPython.display import FileLink

plt.rcParams['figure.figsize'] = (16, 12)
```

In [2]:
```
df = pd.read_csv('ready_dataframe.csv', parse_dates=['transaction_date', 'all_dates'])
```

In [3]:
```
df.drop('Unnamed: 0', axis=1, inplace=True)
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 246740 entries, 0 to 246739
Data columns (total 14 columns):
 #   Column               Non-Null Count   Dtype         
---  ------               --------------   -----         
 0   transaction_date     246740 non-null  datetime64[ns]
 1   store_number         246740 non-null  float64       
 2   loyalty_card_number  246740 non-null  int64         
 3   tax_ID               246740 non-null  float64       
 4   product_number       246740 non-null  float64       
 5   product_name         246740 non-null  object        
 6   product_quantity     246740 non-null  float64       
 7   total_sales          246740 non-null  float64       
 8   weights_of_chips     246740 non-null  int64         
 9   company_name         246740 non-null  object        
 10  lifestage            246740 non-null  object        
 11  premium_customer     246740 non-null  object        
 12  day_of_week          246740 non-null  object        
 13  all_dates            246740 non-null  datetime64[ns]
dtypes: datetime64[ns](2), float64(5), int64(2), object(5)
memory usage: 26.4+ MB
```

In [5]:
```
df['year'] = df.all_dates.dt.year
```

In [6]:
```
df['month'] = df.all_dates.dt.month
```

In [7]:
```
df.month = df.month.astype('str')
```

Let's filter to the pre-trial period and stores with full observation periods.

In [8]:
```
stores_full_observations = avg_price_per_unit = df.groupby(['store_number'])\
                        .agg({'month': 'nunique'})
stores_full_observations.head()
```

Out[8]:

store_number	month
1.0	12
2.0	12
3.0	12
4.0	12
5.0	12

In [9]:
```
stores_full_observations.month.unique()
```

Out[9]:
```
array([12,  2, 11,  1, 10,  3])
```

In [10]:
```
stores_with_full_period = stores_full_observations.query('month == 12')
```

In [11]:
```
stores_with_full_period.month.unique()
```

Out[11]:
```
array([12])
```

In [12]:
```
numbers_of_stores = stores_with_full_period.index
```

In [13]:
```
df_valid_period = df.query('store_number in @numbers_of_stores and all_dates<"2019-02-01
```

Checking our culculations

In [14]:
```
set(numbers_of_stores).difference(set(df_valid_period.store_number.unique()))
```

Out[14]:
```
set()
```

In [15]:
```
df_valid_period
```

Out[15]:

	transaction_date	store_number	loyalty_card_number	tax_ID	product_number	product_name	produ
0	2018-10-17	1.0	1000	1.0	5.0	Natural Chip Compny SeaSalt	
1	2019-05-14	1.0	1307	348.0	66.0	CCs Nacho Cheese	
2	2018-11-10	1.0	1307	346.0	96.0	WW Oriinal Stacked Chips	
3	2019-03-09	1.0	1307	347.0	54.0	CCs Oriinal	
4	2019-05-20	1.0	1343	383.0	61.0	Smiths Crinkle Cut Chips Chicken	
...	...	...	...	...	...	...	
246734	2018-11-12	272.0	272319	270087.0	44.0	Thins Chips Liht Tany	
246736	2018-08-13	272.0	272358	270154.0	74.0	Tostitos Splash Of Lime	
246737	2018-11-06	272.0	272379	270187.0	51.0	Doritos Mexicana	
246738	2018-12-27	272.0	272379	270188.0	42.0	Doritos Corn Chip Mexican Jalapeno	
246739	2018-09-22	272.0	272380	270189.0	74.0	Tostitos Splash Of Lime	

145003 rows × 16 columns

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

Let's first create the metrics of interest and filter to stores that are present throughout the pre-trial period. Calculate these measures over time for each store.

For each store and month we will calculate total sales.

In [16]:
```
total_sales = df_valid_period.groupby(['store_number', 'year', 'month'], as_index=False)
total_sales
```

Out[16]:

	store_number	year	month	total_sales
0	1.0	2018	10	169.40
1	1.0	2018	11	181.00
2	1.0	2018	12	160.60
3	1.0	2018	7	210.00
4	1.0	2018	8	168.40
...	...	...	...	...
1808	272.0	2018	12	363.10
1809	272.0	2018	7	392.30
1810	272.0	2018	8	326.95
1811	272.0	2018	9	294.50
1812	272.0	2019	1	392.40

1813 rows × 4 columns

For each store and month we will calculate number of customers.

In [17]:
```
number_of_customers = df_valid_period.groupby(['store_number', 'year', 'month'], as_inde
                      .agg({'loyalty_card_number': 'nunique'})
number_of_customers
```

Out[17]:

	store_number	year	month	loyalty_card_number
0	1.0	2018	10	38
1	1.0	2018	11	43
2	1.0	2018	12	37
3	1.0	2018	7	50
4	1.0	2018	8	41
...	...	...	...	...
1808	272.0	2018	12	43
1809	272.0	2018	7	47
1810	272.0	2018	8	39
1811	272.0	2018	9	31
1812	272.0	2019	1	44

1813 rows × 4 columns

For each store and month we will calculate transactions per customer.

In [18]:
```
transactions_per_customer = df_valid_period.groupby(['store_number', 'year', 'month'], a
                      .agg({'loyalty_card_number': 'count'})

transactions_per_customer.rename(columns = {'loyalty_card_number': 'loyalty_card_number_
```

```
transactions_per_customer['per_customer'] = transactions_per_customer.loyalty_card_numbe

transactions_per_customer
```

Out[18]:

	store_number	year	month	loyalty_card_number_count	per_customer
0	1.0	2018	10	39	1.026316
1	1.0	2018	11	44	1.023256
2	1.0	2018	12	40	1.081081
3	1.0	2018	7	54	1.080000
4	1.0	2018	8	41	1.000000
...	...	...	...	...	...
1808	272.0	2018	12	43	1.000000
1809	272.0	2018	7	48	1.021277
1810	272.0	2018	8	43	1.102564
1811	272.0	2018	9	35	1.129032
1812	272.0	2019	1	47	1.068182

1813 rows × 5 columns

For each store and month we will calculate chips per customer.

In [19]:
```
chips_per_customer = df_valid_period.groupby(['store_number', 'year', 'month'], as_index
                     .agg({'product_quantity': 'mean'})
chips_per_customer
```

Out[19]:

	store_number	year	month	product_quantity
0	1.0	2018	10	1.256410
1	1.0	2018	11	1.204545
2	1.0	2018	12	1.200000
3	1.0	2018	7	1.259259
4	1.0	2018	8	1.268293
...	...	...	...	...
1808	272.0	2018	12	1.883721
1809	272.0	2018	7	1.875000
1810	272.0	2018	8	1.767442
1811	272.0	2018	9	1.971429
1812	272.0	2019	1	1.914894

1813 rows × 4 columns

For each store and month we will calculate average price per unit.

In [20]:
```
avg_price_per_unit = df_valid_period.groupby(['store_number', 'year', 'month'], as_index
                        .agg({'product_quantity': 'sum', 'total_sales': 'sum'})
avg_price_per_unit['avg_price'] = avg_price_per_unit.total_sales/avg_price_per_unit.prod
avg_price_per_unit
```

Out[20]:

	store_number	year	month	product_quantity	total_sales	avg_price
0	1.0	2018	10	49.0	169.40	3.457143
1	1.0	2018	11	53.0	181.00	3.415094
2	1.0	2018	12	48.0	160.60	3.345833
3	1.0	2018	7	68.0	210.00	3.088235
4	1.0	2018	8	52.0	168.40	3.238462
...	...	...	...	...	...	...
1808	272.0	2018	12	81.0	363.10	4.482716
1809	272.0	2018	7	90.0	392.30	4.358889
1810	272.0	2018	8	76.0	326.95	4.301974
1811	272.0	2018	9	69.0	294.50	4.268116
1812	272.0	2019	1	90.0	392.40	4.360000

1813 rows × 6 columns

Now we need to work out a way of ranking how similar each potential control store is to the trial store. We can calculate how correlated the performance of each store is to the trial store.

Total sales correlation

In [21]:
```
total_sales
```

Out[21]:

	store_number	year	month	total_sales
0	1.0	2018	10	169.40
1	1.0	2018	11	181.00
2	1.0	2018	12	160.60
3	1.0	2018	7	210.00
4	1.0	2018	8	168.40
...	...	...	...	...
1808	272.0	2018	12	363.10
1809	272.0	2018	7	392.30
1810	272.0	2018	8	326.95
1811	272.0	2018	9	294.50
1812	272.0	2019	1	392.40

1813 rows × 4 columns

In [22]:
```
list_of_store_numbers = total_sales.store_number.unique()
list_of_store_numbers
```

Out[22]:
```
array([  1.,   2.,   3.,   4.,   5.,   6.,   7.,   8.,   9.,  10.,  12.,
        13.,  14.,  15.,  16.,  17.,  18.,  19.,  20.,  21.,  22.,  23.,
        24.,  25.,  26.,  27.,  28.,  29.,  30.,  32.,  33.,  34.,  35.,
        36.,  37.,  38.,  39.,  40.,  41.,  42.,  43.,  45.,  46.,  47.,
        48.,  49.,  50.,  51.,  52.,  53.,  54.,  55.,  56.,  57.,  58.,
        59.,  60.,  61.,  62.,  63.,  64.,  65.,  66.,  67.,  68.,  69.,
        70.,  71.,  72.,  73.,  74.,  75.,  77.,  78.,  79.,  80.,  81.,
        82.,  83.,  84.,  86.,  87.,  88.,  89.,  90.,  91.,  93.,  94.,
        95.,  96.,  97.,  98.,  99., 100., 101., 102., 103., 104., 105.,
       106., 107., 108., 109., 110., 111., 112., 113., 114., 115., 116.,
       118., 119., 120., 121., 122., 123., 124., 125., 126., 127., 128.,
       129., 130., 131., 132., 133., 134., 135., 136., 137., 138., 139.,
       140., 141., 142., 143., 144., 145., 146., 147., 148., 149., 150.,
       151., 152., 153., 154., 155., 156., 157., 158., 159., 160., 161.,
       162., 163., 164., 165., 166., 167., 168., 169., 170., 171., 172.,
       173., 174., 175., 176., 178., 179., 180., 181., 182., 183., 184.,
       185., 186., 187., 188., 189., 190., 191., 192., 194., 195., 196.,
       197., 198., 199., 200., 201., 202., 203., 204., 205., 207., 208.,
       209., 210., 212., 213., 214., 215., 216., 217., 219., 220., 221.,
       222., 223., 224., 225., 226., 227., 228., 229., 230., 231., 232.,
       233., 234., 235., 236., 237., 238., 239., 240., 241., 242., 243.,
       244., 245., 246., 247., 248., 249., 250., 251., 253., 254., 255.,
       256., 257., 258., 259., 260., 261., 262., 263., 264., 265., 266.,
       267., 268., 269., 270., 271., 272.])
```

Correlation function

In [23]:
```
from scipy import stats
```

In [24]:
```
def correlation(list_of_store_numbers, df, store_number, col):
    value = dict()
    for j in list_of_store_numbers:
        res = stats.pearsonr(df.query('store_number == @store_number')[col].values,\
                             df.query('store_number == @j')[col].values)
```

```
        value[j] = [res.statistic, res.pvalue]
    return value
```

In [25]:
```
# value = dict()
# for j in list_of_store_numbers:
#     res = total_sales.query('store_number == 77').\
#     merge(total_sales.query('store_number == @j'), on='month')[['total_sales_x', 'tota
#     if len(res.query('total_sales_x != 1').total_sales_x) > 0:
#         temp = res.query('total_sales_x != 1').total_sales_x.item()
#         value[j] = temp
```

In [26]:
```
cor_store_77 = correlation(list_of_store_numbers, df=total_sales,  store_number=77, col=
```

In [27]:
```
cor_store_86 = correlation(list_of_store_numbers, df=total_sales,  store_number=86, col=
```

In [28]:
```
cor_store_88 = correlation(list_of_store_numbers, df=total_sales,  store_number=88, col=
```

Number of customers correlation

In [29]:
```
number_of_customers
```

Out[29]:

	store_number	year	month	loyalty_card_number
0	1.0	2018	10	38
1	1.0	2018	11	43
2	1.0	2018	12	37
3	1.0	2018	7	50
4	1.0	2018	8	41
...	...	...	...	...
1808	272.0	2018	12	43
1809	272.0	2018	7	47
1810	272.0	2018	8	39
1811	272.0	2018	9	31
1812	272.0	2019	1	44

1813 rows × 4 columns

In [30]:
```
cor_store_numb_77 = correlation(list_of_store_numbers, df=number_of_customers,  store_nu
```

In [31]:
```
cor_store_numb_86 = correlation(list_of_store_numbers, df=number_of_customers,  store_nu
```

In [32]:
```
cor_store_numb_88 = correlation(list_of_store_numbers, df=number_of_customers,  store_nu
```

Transactions per customer correlation.

In [33]:
```
transactions_per_customer
```

Out[33]:

	store_number	year	month	loyalty_card_number_count	per_customer
0	1.0	2018	10	39	1.026316
1	1.0	2018	11	44	1.023256
2	1.0	2018	12	40	1.081081
3	1.0	2018	7	54	1.080000
4	1.0	2018	8	41	1.000000
...	...	...	...	...	...
1808	272.0	2018	12	43	1.000000
1809	272.0	2018	7	48	1.021277
1810	272.0	2018	8	43	1.102564
1811	272.0	2018	9	35	1.129032
1812	272.0	2019	1	47	1.068182

1813 rows × 5 columns

In [34]:
```
cor_store_trans_77 = correlation(list_of_store_numbers, df=transactions_per_customer,  s
```

```
/home/alberdinamariya/.local/lib/python3.8/site-packages/scipy/stats/_stats_py.py:4424:
ConstantInputWarning: An input array is constant; the correlation coefficient is not def
ined.
  warnings.warn(stats.ConstantInputWarning(msg))
```

In [35]:
```
cor_store_trans_86 = correlation(list_of_store_numbers, df=transactions_per_customer,  s
```

In [36]:
```
cor_store_trans_88 = correlation(list_of_store_numbers, df=transactions_per_customer,  s
```

Chips per customer correlation

In [37]:
```
chips_per_customer
```

Out[37]:

	store_number	year	month	product_quantity
0	1.0	2018	10	1.256410
1	1.0	2018	11	1.204545
2	1.0	2018	12	1.200000
3	1.0	2018	7	1.259259
4	1.0	2018	8	1.268293
...	...	...	...	...
1808	272.0	2018	12	1.883721
1809	272.0	2018	7	1.875000
1810	272.0	2018	8	1.767442
1811	272.0	2018	9	1.971429
1812	272.0	2019	1	1.914894

1813 rows × 4 columns

In [38]:
```
cor_store_chips_77 = correlation(list_of_store_numbers, df=chips_per_customer,  store_nu
```

In [39]:
```
cor_store_chips_86 = correlation(list_of_store_numbers, df=chips_per_customer,  store_nu
```

In [40]:
```
cor_store_chips_88 = correlation(list_of_store_numbers, df=chips_per_customer,  store_nu
```

Average price per unit correlation.

In [41]:
```
avg_price_per_unit
```

Out[41]:

	store_number	year	month	product_quantity	total_sales	avg_price
0	1.0	2018	10	49.0	169.40	3.457143
1	1.0	2018	11	53.0	181.00	3.415094
2	1.0	2018	12	48.0	160.60	3.345833
3	1.0	2018	7	68.0	210.00	3.088235
4	1.0	2018	8	52.0	168.40	3.238462
...	...	...	...	...	...	...
1808	272.0	2018	12	81.0	363.10	4.482716
1809	272.0	2018	7	90.0	392.30	4.358889
1810	272.0	2018	8	76.0	326.95	4.301974
1811	272.0	2018	9	69.0	294.50	4.268116
1812	272.0	2019	1	90.0	392.40	4.360000

1813 rows × 6 columns

In [42]:
```
cor_store_avg_77 = correlation(list_of_store_numbers, df=avg_price_per_unit,  store_numb
```

In [43]:
```
cor_store_avg_86 = correlation(list_of_store_numbers, df=avg_price_per_unit,  store_numb
```

In [44]:
```
cor_store_avg_88 = correlation(list_of_store_numbers, df=avg_price_per_unit,  store_numb
```

we may chose p_value<0,05 and correlation more than 0,7 to choose suitable stores.

In [45]:
```
def table_correlation(corr_metrics, p_value, correlation):
    res = dict()
    for key,value in corr_metrics.items():
        if (value[0] >= correlation or value[0] <= -correlation) and value[1] < p_value:
            res[key] = [value[0], value[1]]
    return res
```

In [46]:
```
stores_sales_77 = table_correlation(corr_metrics=cor_store_77, p_value=0.05, correlation
```

In [47]:
```
stores_numb_77 = table_correlation(corr_metrics=cor_store_numb_77, p_value=0.05, correla
```

In [48]:
```
similar_stores_to_77 = set(stores_sales_77.keys())& set(stores_numb_77.keys())
similar_stores_to_77
```

Out[48]:
```
{9.0, 77.0, 157.0, 162.0, 186.0, 233.0}
```

In [49]:
```
stores_sales_86 = table_correlation(corr_metrics=cor_store_86, p_value=0.05, correlation
```

In [50]:
```
stores_numb_86 = table_correlation(corr_metrics=cor_store_numb_86, p_value=0.05, correla
```

In [51]:
```
similar_stores_to_86 = set(stores_sales_86.keys())& set(stores_numb_86.keys())
similar_stores_to_86
```

Out[51]:
```
{86.0, 155.0}
```

In [52]:
```
stores_sales_88 = table_correlation(corr_metrics=cor_store_88, p_value=0.05, correlation
```

In [53]:
```
stores_numb_88 = table_correlation(corr_metrics=cor_store_numb_88, p_value=0.05, correla
```

In [54]:
```
similar_stores_to_88 = set(stores_sales_88.keys())& set(stores_numb_88.keys())
```

In [55]:
```
similar_stores_to_88
```

Out[55]:
```
{88.0}
```

In [56]:
```
cor_store_88
```

Out[56]: {1.0: [0.6551983956289403, 0.11016760861274533],
2.0: [-0.48781557578538753, 0.26676805075620336],
3.0: [-0.5585077331888858, 0.192523736487302],
4.0: [-0.5803015406475007, 0.17197105283568914],
5.0: [0.213003258473512, 0.6465413976750768],
6.0: [0.09903278661006223, 0.8326999394843351],
7.0: [0.4006335723499198, 0.37310623453684877],
8.0: [-0.6292483963150707, 0.1300072551178054],
9.0: [0.42512103736596196, 0.3417003172448471],
10.0: [0.1093156845633588, 0.8155267730561411],
12.0: [-0.16443635392403974, 0.7246029095298316],
13.0: [0.33436639943048596, 0.46355372716667526],
14.0: [0.45427539431541547, 0.30584537402847806],
15.0: [-0.42759891462555033, 0.3385869438328566],
16.0: [0.029165246295417682, 0.9505085953620381],
17.0: [-0.3995814156283062, 0.374481167286327],
18.0: [-0.04626449160983194, 0.9215429874398975],
19.0: [-0.12437677641094827, 0.7904808198498728],
20.0: [-0.17106649976223043, 0.7138190368673862],
21.0: [0.04328730230364738, 0.9265820233154166],
22.0: [0.08092533917567282, 0.8630662899552128],
23.0: [-0.8144946874781911, 0.025689875963589577],
24.0: [0.19315130393612318, 0.6781784033668646],
25.0: [0.18540097467246322, 0.6906349954644093],
26.0: [0.11494770047993783, 0.8061453620358301],
27.0: [-0.17567150580570393, 0.7063511564302405],
28.0: [0.10956681170352248, 0.815108082823559],
29.0: [0.2882216672037783, 0.5307673982296472],
30.0: [-0.40041477522462277, 0.3733919829917961],
32.0: [0.2231480832634472, 0.6305330567216495],
33.0: [-0.5223247852657276, 0.22910830686643238],
34.0: [-0.3773249121732257, 0.4040423079181692],
35.0: [-0.31198998088399615, 0.4957459998877815],
36.0: [0.49432839849235344, 0.25945921012381223],
37.0: [-0.6092556645606558, 0.14643718695009578],
38.0: [0.3417821683658219, 0.4530598415498739],
39.0: [0.46089954922558124, 0.29793997804802524],
40.0: [-0.20581768373442172, 0.6579463833123314],
41.0: [0.1493766199713979, 0.7492300992819622],
42.0: [-0.30878472526987516, 0.500420318795656],
43.0: [-0.2698543055434116, 0.5583777021952042],
45.0: [-0.5814542741208779, 0.17091560680133783],
46.0: [0.09307580084455672, 0.842673152892798],
47.0: [0.058047186766795045, 0.9016219719433276],
48.0: [-0.9417953108368134, 0.0015211456212283116],
49.0: [-0.11119584344571223, 0.8123929422622355],
50.0: [-0.3752347869291638, 0.40686436454882624],
51.0: [0.17146765887530427, 0.7131677515299771],
52.0: [-0.26911993483236674, 0.5594910950718509],
53.0: [-0.35646264940030803, 0.43255276839791773],
54.0: [0.02289999172905388, 0.961133959342139],
55.0: [0.4878112378295851, 0.26677294966088444],
56.0: [0.1459929476928637, 0.7547874886483892],
57.0: [-0.33966216824554674, 0.45605069680626176],
58.0: [-0.06234836744481563, 0.894359732349082],
59.0: [-0.3440868917516337, 0.44981672460887034],
60.0: [-0.07151113448697015, 0.8789091022938815],
61.0: [0.7306400711350398, 0.06216675621415794],
62.0: [-0.01255049823296362, 0.9786952904564474],
63.0: [-0.34782814867969214, 0.44457080527449117],
64.0: [-0.1622055131183909, 0.7282395874063866],

65.0: [-0.4339667908843851, 0.330641485848686],
66.0: [0.046715667171913056, 0.9207795298629781],
67.0: [-0.5464449560788514, 0.20438282377492983],
68.0: [-0.18094705536436295, 0.6978188474653967],
69.0: [0.2873767105021984, 0.5320273652812503],
70.0: [0.016200670358987063, 0.9725004969119286],
71.0: [-0.59272513689341, 0.16076512149285355],
72.0: [-0.5719610492082153, 0.17970272213562716],
73.0: [-0.4954145275883993, 0.25824933964585334],
74.0: [-0.46153601445777337, 0.2971852059738384],
75.0: [0.07860165483948939, 0.8669735097215298],
77.0: [-0.3411806444038769, 0.45390771457306],
78.0: [0.27050967463656966, 0.5573846877207681],
79.0: [0.7137670260388742, 0.07164574479170026],
80.0: [0.07672709416244325, 0.8701271120807175],
81.0: [0.05734731635417469, 0.9028041808120951],
82.0: [0.056826736268219824, 0.9036836283195827],
83.0: [-0.224501659776004, 0.628405652279448],
84.0: [-0.5925900974521249, 0.16088491479066624],
86.0: [0.33396921477881697, 0.4641182794579646],
87.0: [-0.3872045689729783, 0.3908087689272274],
88.0: [0.9999999999999999, 2.494476486799542e-40],
89.0: [0.05735056818223627, 0.9027986875493602],
90.0: [-0.5456694355329583, 0.20515689723139877],
91.0: [0.8991629536183667, 0.005869902902641812],
93.0: [0.04258952879968911, 0.9277633258247818],
94.0: [0.24126063932246142, 0.6022384657725373],
95.0: [0.252114724665908, 0.5854683977335698],
96.0: [-0.25901065727978356, 0.5748895322616572],
97.0: [-0.003916056465074827, 0.993351947041544],
98.0: [-0.33841766350006197, 0.4578098289477469],
99.0: [0.49471076322787444, 0.2590329883822199],
100.0: [-0.5173073001154658, 0.23441876241708803],
101.0: [0.3908568869252363, 0.38596131639866005],
102.0: [0.5942052260766936, 0.1594550435191832],
103.0: [0.45995591849409684, 0.299060564642298],
104.0: [-0.6361049205901735, 0.12460127401120026],
105.0: [-0.4950431666582188, 0.2586627185898644],
106.0: [0.5149035135083825, 0.23698303827250317],
107.0: [-0.5656692630947596, 0.18564528854068194],
108.0: [0.07140601661179788, 0.8790861908231193],
109.0: [-0.13792762465517033, 0.7680676905366054],
110.0: [-0.49921572393752467, 0.25403546120683335],
111.0: [-0.3798502588003367, 0.400642996531598],
112.0: [-0.47162934537199874, 0.28532919036433857],
113.0: [0.11955752548332255, 0.7984803318301635],
114.0: [-0.05161077364568868, 0.9124994744742788],
115.0: [-0.5040344189511488, 0.24873939113449253],
116.0: [-0.027115337366745172, 0.9539844942035454],
118.0: [-0.6584710985808431, 0.10778567886498507],
119.0: [-0.2609701923074157, 0.5718944513798377],
120.0: [-0.0552572780605373, 0.9063354802446585],
121.0: [-0.19400828559576902, 0.6768045523750046],
122.0: [-0.32123636603846284, 0.48234912659740486],
123.0: [0.42224233951326784, 0.34533241465637243],
124.0: [0.26384980914413236, 0.5675020069009621],
125.0: [0.4953659651650819, 0.25830337953906185],
126.0: [0.07814399904885534, 0.8677433030020689],
127.0: [0.1074440620484899, 0.8186483185407069],
128.0: [-0.02726741211582609, 0.9537266103641658],
129.0: [0.16713218463631413, 0.7202137039111521],
130.0: [-0.555653146083214, 0.19529926482644552],

131.0: [-0.19515690788270415, 0.6749642812497962],
132.0: [0.1542541362008297, 0.7412344048643339],
133.0: [-0.6506440437096528, 0.11352726810785835],
134.0: [0.6033898188883742, 0.15144488292692224],
135.0: [-0.5486445536354636, 0.202194960710575],
136.0: [0.3849536426853102, 0.39380834933802705],
137.0: [-0.1169316197961305 9, 0.802845038030687],
138.0: [0.41808947402317265, 0.3506005881855372],
139.0: [-0.21375005041635275, 0.6453591944278346],
140.0: [0.39697889477836473, 0.3778909406313201],
141.0: [-0.6893722989028507, 0.08663161380771339],
142.0: [-0.121595773051079, 0.7950952832603994],
143.0: [-0.33670538187918636, 0.4602342769253123],
144.0: [-0.5156329514886716, 0.23620352465913536],
145.0: [0.4031090618452434, 0.36987949588393365],
146.0: [-0.16025079892388203, 0.7314294649872369],
147.0: [-0.012607353957322084, 0.97859879208887],
148.0: [-0.3997410493394765, 0.37427242883897144],
149.0: [-0.12358760575113187, 0.7917897856505127],
150.0: [-0.019429429625419922, 0.9670218013283092],
151.0: [-0.49568930166702807, 0.25794367052448597],
152.0: [-0.31268238022605443, 0.4947382856558342],
153.0: [-0.26856732200275396, 0.5603293910071568],
154.0: [-0.5742533114815549, 0.1775611698107236],
155.0: [0.40407790559957407, 0.3686197639612455],
156.0: [-0.6220300304392042, 0.13582504276113502],
157.0: [-0.2044226780316272, 0.6601667386252089],
158.0: [0.33119800271570654, 0.46806426244705684],
159.0: [0.9080434092789186, 0.0046846165787381025],
160.0: [0.09457573251489435, 0.8401603408994657],
161.0: [0.5508425109358558, 0.20002001920068121],
162.0: [-0.20382861522970536, 0.661112879206082],
163.0: [0.7384483883806546, 0.05802510174673048],
164.0: [-0.6282699883120771, 0.13078823114835197],
165.0: [-0.35769527368481563, 0.43084737873114737],
166.0: [-0.2863965924803149, 0.5334901333498728],
167.0: [0.26617926398078706, 0.5639566149383184],
168.0: [-0.5926088404286626, 0.16086828527300126],
169.0: [0.2734146708435663, 0.5529899174205977],
170.0: [-0.07250168886357011, 0.8772405467784151],
171.0: [0.3704949493869972, 0.41329264042725194],
172.0: [-0.037960303831616034, 0.9356030076561141],
173.0: [-0.704301070971494, 0.0772808591133387],
174.0: [-0.35398743325999193, 0.435985114139169],
175.0: [-0.8104167349416412, 0.027061864216975 84],
176.0: [-0.41057083410493467, 0.3602231498317801],
178.0: [0.1438010849676779, 0.7583919403767095],
179.0: [0.1902832297156079, 0.6827814311378114],
180.0: [-0.33690392132688723, 0.4599529185860647],
181.0: [-0.38880990499135587, 0.3886751262222126],
182.0: [0.7184910851355408, 0.0689188358329382],
183.0: [-0.2583147931636789, 0.5759543151984479],
184.0: [-0.6139232022474762, 0.14251296015111226],
185.0: [-0.5152219099185142, 0.23664263690450088],
186.0: [0.37939589253532346, 0.40125376843597343],
187.0: [0.5454098255157376, 0.20541633580179897],
188.0: [0.28904728063196927, 0.529537244445543],
189.0: [-0.46363606690253356, 0.2947007899321705],
190.0: [0.6023309920812483, 0.15235780156943754],
191.0: [-0.18988751708955734, 0.6834171349179571],
192.0: [0.10689350990592823, 0.8195669093451821],
194.0: [-0.3469953217665713, 0.44573657546879397],

195.0: [-0.39204228017124454, 0.3843932758519849],
196.0: [-0.528109327330819, 0.22305689894996264],
197.0: [-0.7921259808539602, 0.03371276016886981],
198.0: [0.491062777699438, 0.2631124140956784],
199.0: [-0.8497546899893335, 0.015472983215015957],
200.0: [0.26087122517015315, 0.5720456013547174],
201.0: [0.5295684895570872, 0.2215424694713352],
202.0: [-0.16164008819922843, 0.7291619798398904],
203.0: [-0.03657793230250023, 0.9379449061050477],
204.0: [0.8233347176492806, 0.022852876633033738],
205.0: [-0.49258382816771584, 0.2614079393694549],
207.0: [-0.12750418874050234, 0.7852973792158884],
208.0: [0.23464128178190166, 0.6125350189494598],
209.0: [-0.34115820517240175, 0.4539393550425241],
210.0: [-0.3079208444919455, 0.5016827765944738],
212.0: [0.3372082285091034, 0.4595217960967695],
213.0: [0.39660797862792135, 0.37837793277807913],
214.0: [-0.20452469451784436, 0.6600042971164439],
215.0: [-0.47698401756824704, 0.2791267763028116],
216.0: [-0.43135482884888615, 0.3338908217789003],
217.0: [0.3413815986923499, 0.45362439536768745],
219.0: [-0.23292242843747174, 0.6152171113509113],
220.0: [0.23680267050274703, 0.6091672700979752],
221.0: [-0.7556790403402811, 0.04943219730139123],
222.0: [0.22802218237033048, 0.6228820660255382],
223.0: [0.5669232372544049, 0.1844533935238934],
224.0: [0.01404315578415357, 0.9761619490129291],
225.0: [0.6803616905624359, 0.09254983606919287],
226.0: [-0.006049376129175499, 0.9897304480310947],
227.0: [-0.4152358273739288, 0.3542399729332345],
228.0: [0.44463156181938457, 0.3175159219034449],
229.0: [0.09880689000464446, 0.8330778184518552],
230.0: [-0.7810587824465648, 0.03813610206511609],
231.0: [0.07068937865679253, 0.8802935930549329],
232.0: [0.0907524422887761, 0.8465675291379874],
233.0: [-0.2095177341800162, 0.6520669160908082],
234.0: [-0.015407365313688998, 0.9738467487475331],
235.0: [-0.8593533026258688, 0.013190009590299177],
236.0: [-0.5989792694056073, 0.15526574087264897],
237.0: [0.04233532390075459, 0.9281937108185718],
238.0: [0.3960612326634705, 0.37909624473911424],
239.0: [-0.39668700085768815, 0.37827415962121314],
240.0: [0.7638893267323539, 0.045601288259359565],
241.0: [-0.29784457656811536, 0.5164895308568336],
242.0: [0.3085978419583327, 0.500693331739794],
243.0: [0.4364553578110003, 0.3275582923157596],
244.0: [0.16620716369189031, 0.721719090416731],
245.0: [0.2830110179740941, 0.5385532041605375],
246.0: [0.14669796609709929, 0.7536288559316293],
247.0: [-0.1880073140951824, 0.6864396668180742],
248.0: [0.3413346160780154, 0.4536906287857243],
249.0: [-0.1725815491821253, 0.7113600677834544],
250.0: [-0.5207404827803875, 0.2307789636243391],
251.0: [-0.19716743317290192, 0.6717462079446037],
253.0: [0.5412455163514426, 0.20959931114019492],
254.0: [0.0049976238073215495, 0.9915158762701082],
255.0: [0.06716906573473108, 0.8862273113138879],
256.0: [-0.40064633946940464, 0.37308956347639166],
257.0: [0.29806856760736344, 0.5161587687823076],
258.0: [0.23438966165126887, 0.612927432202983],
259.0: [0.19166719667217064, 0.6805592917445817],
260.0: [0.026297898226474353, 0.955370737456004],

```
 261.0: [-0.6559538927529264, 0.10961534652090696],
 262.0: [-0.2963183521479198, 0.5187452005337626],
 263.0: [-0.5330732157093205, 0.2179249146400788],
 264.0: [-0.673283634446904, 0.09734335151185128],
 265.0: [-0.147780146909857, 0.7518511068667277],
 266.0: [-0.08376038746723809, 0.8583022341899285],
 267.0: [0.39889437192539967, 0.37538009408539025],
 268.0: [-0.19914461059855396, 0.6685853923680791],
 269.0: [-0.15715082647459847, 0.7364945744972032],
 270.0: [-0.6290472934578166, 0.13016758299897582],
 271.0: [-0.22012260297630243, 0.6352954990774607],
 272.0: [-0.672398539110579, 0.09795170928219551]}
```

In [57]:
```
stores_numb_88
```

Out[57]:
```
{14.0: [0.9559752052403879, 0.000762740476892847],
 35.0: [0.7715692253424996, 0.042171059527333965],
 57.0: [0.7698186806082465, 0.04293994344305898],
 77.0: [0.8539667858464666, 0.014445509716088845],
 88.0: [0.9999999999999998, 1.411088991461081e-39],
 178.0: [0.9287152141473084, 0.0025070102233084274],
 208.0: [-0.7943228176446844, 0.03287069047607607],
 233.0: [0.7863939199352599, 0.03596592786225097],
 237.0: [0.9857550954123973, 4.616155537089377e-05],
 266.0: [-0.7940157920136566, 0.03298766299506544]}
```

The store with the highest score is then selected as the control store since it is most similar to the trial store.

In [58]:
```
for key,value in stores_sales_77.items():
    if key in similar_stores_to_77:
        print(key, stores_sales_77[key])
```

```
9.0 [-0.770266756106569, 0.04274240736139248]
77.0 [1.0, 0.0]
157.0 [0.7765451462605506, 0.04002731411945841]
162.0 [0.8575839389643507, 0.013595218588036553]
186.0 [-0.9171305971313111, 0.0036297903107149576]
233.0 [0.9736429414543201, 0.0002135680496620952]
```

In [59]:
```
final_stores_similar_to_77 = [233, 186, 50, 162, 71]
```

In [60]:
```
for key,value in stores_sales_86.items():
    if key in similar_stores_to_86:
        print(key, stores_sales_86[key])
```

```
86.0 [0.9999999999999999, 2.494476486799542e-40]
155.0 [0.869532475367053, 0.010994112545311251]
```

In [61]:
```
final_stores_similar_to_86 = [260, 155, 6, 138]
```

In [62]:
```
for key,value in stores_sales_88.items():
    if key in similar_stores_to_88:
        print(key, stores_sales_88[key])
```

```
88.0 [0.9999999999999999, 2.494476486799542e-40]
```

Vizualisations

In [63]:
```
df_valid_period.groupby('store_number', as_index=False).\
agg({'total_sales': 'sum'}).query('store_number == 77 or store_number == 50 or store_num
```

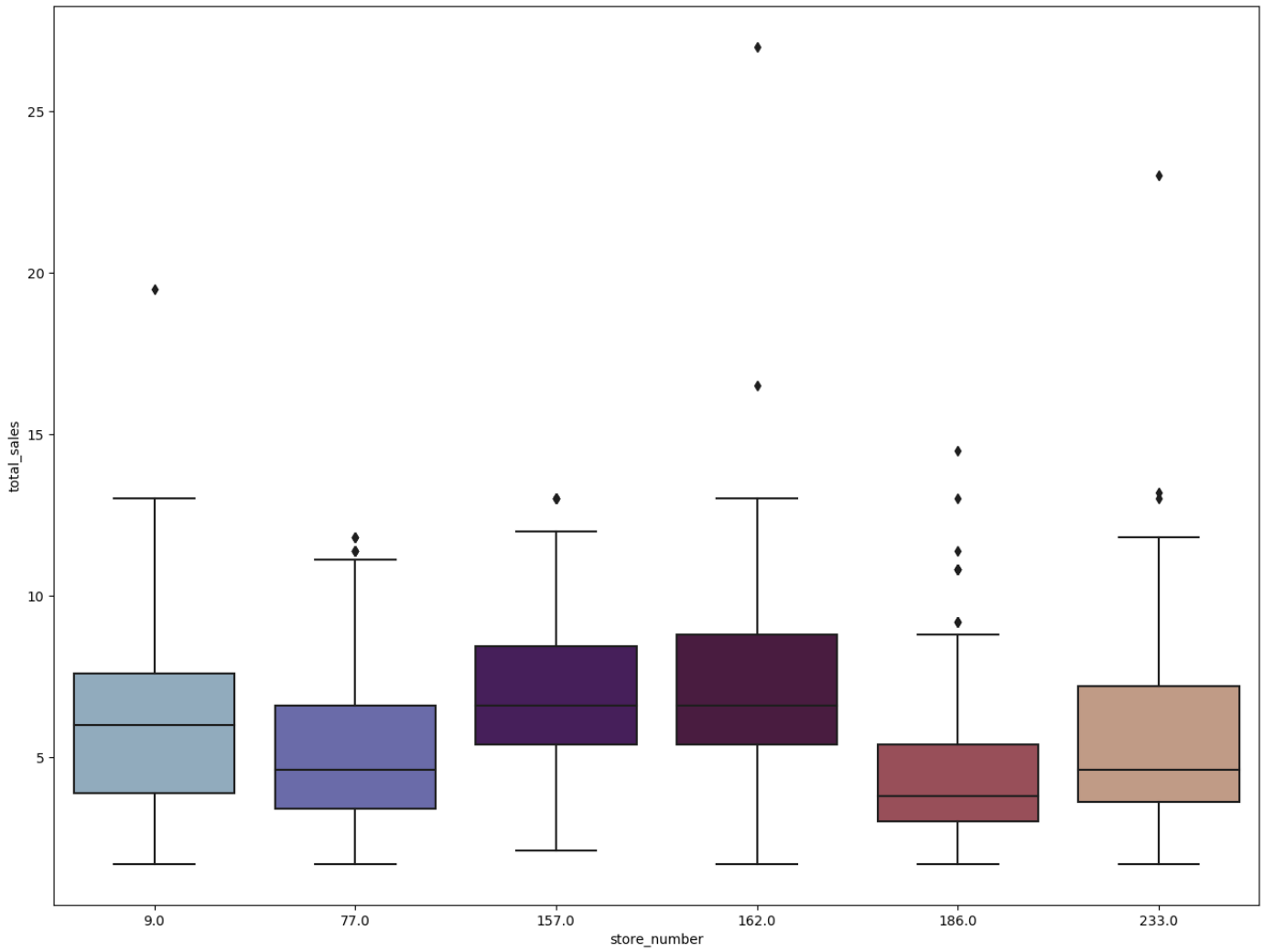
Out[63]:

	store_number	total_sales
46	50.0	1788.9
72	77.0	1595.5
195	205.0	1700.3
220	233.0	1534.5

Total_sales boxplot

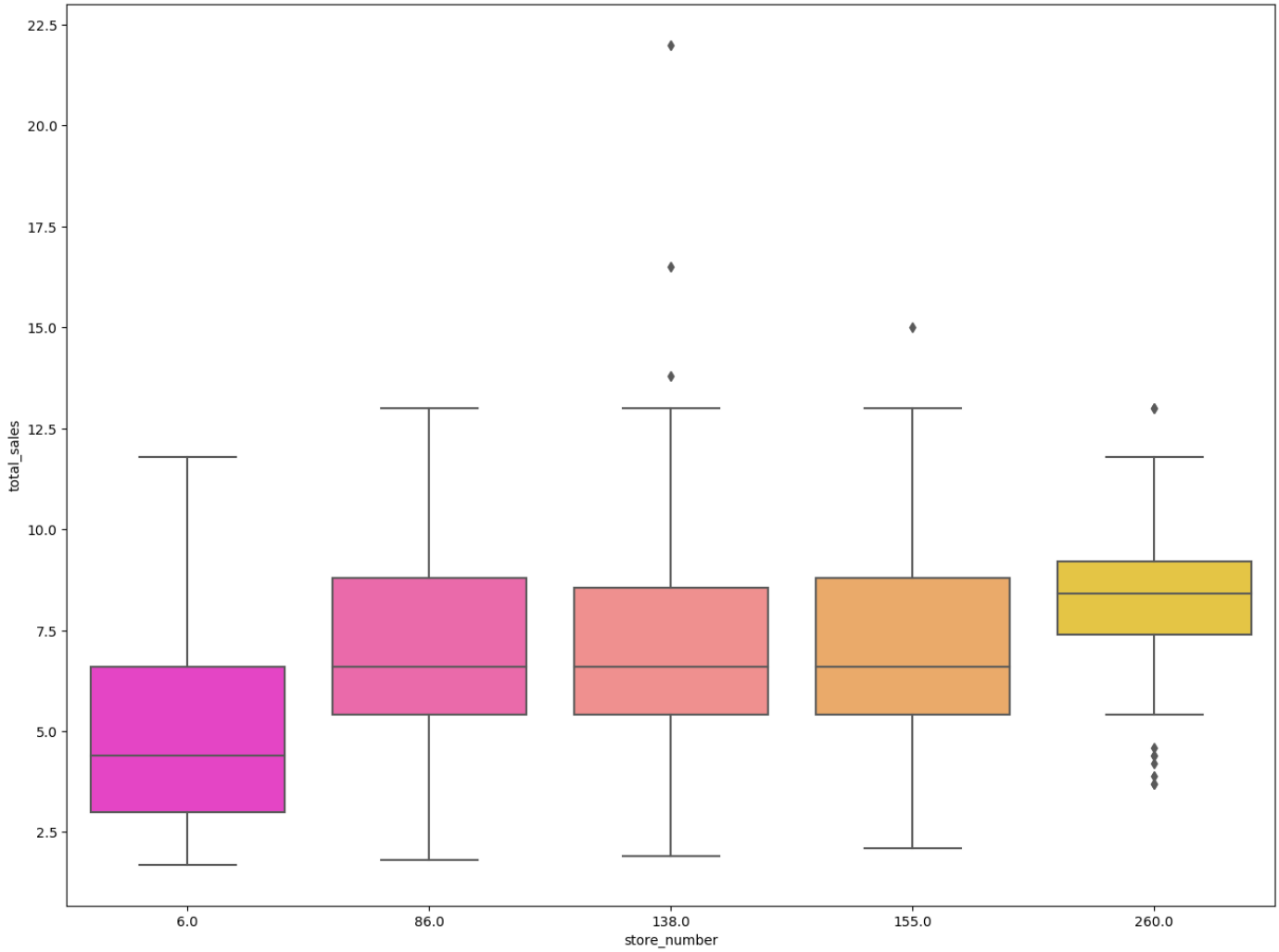
In [64]:
```
# 77 store
sns.boxplot(data = df_valid_period.\
        query('store_number == 77 or store_number == 9 or store_number == 162 or store_n
        x='store_number', y='total_sales', palette='twilight')
```

Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x7fa000b234c0>



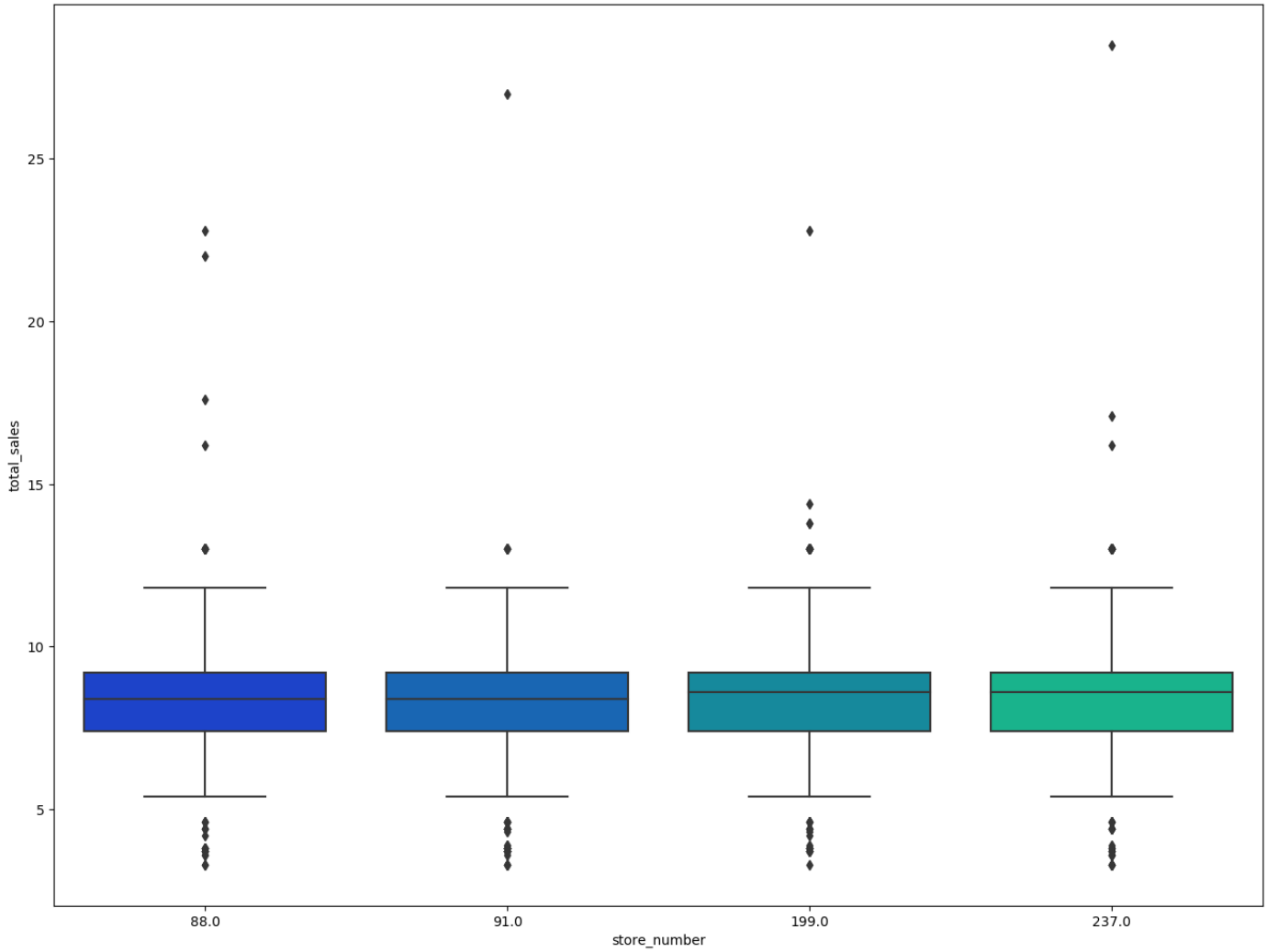
In [65]:
```
# 86 store
sns.boxplot(data = df_valid_period.\
        query('store_number == 86 or store_number == 6 \
        or store_number == 155 or store_number == 138 or store_number == 260'),\
        x='store_number', y='total_sales', palette='spring')
```

Out[65]: <matplotlib.axes._subplots.AxesSubplot at 0x7fa000b23700>



In [66]:
```
# 88
sns.boxplot(data = df_valid_period.\
        query('store_number == 88 or store_number == 91\
         or store_number == 237\
         or store_number == 199'),
        x='store_number', y='total_sales', palette='winter')
```

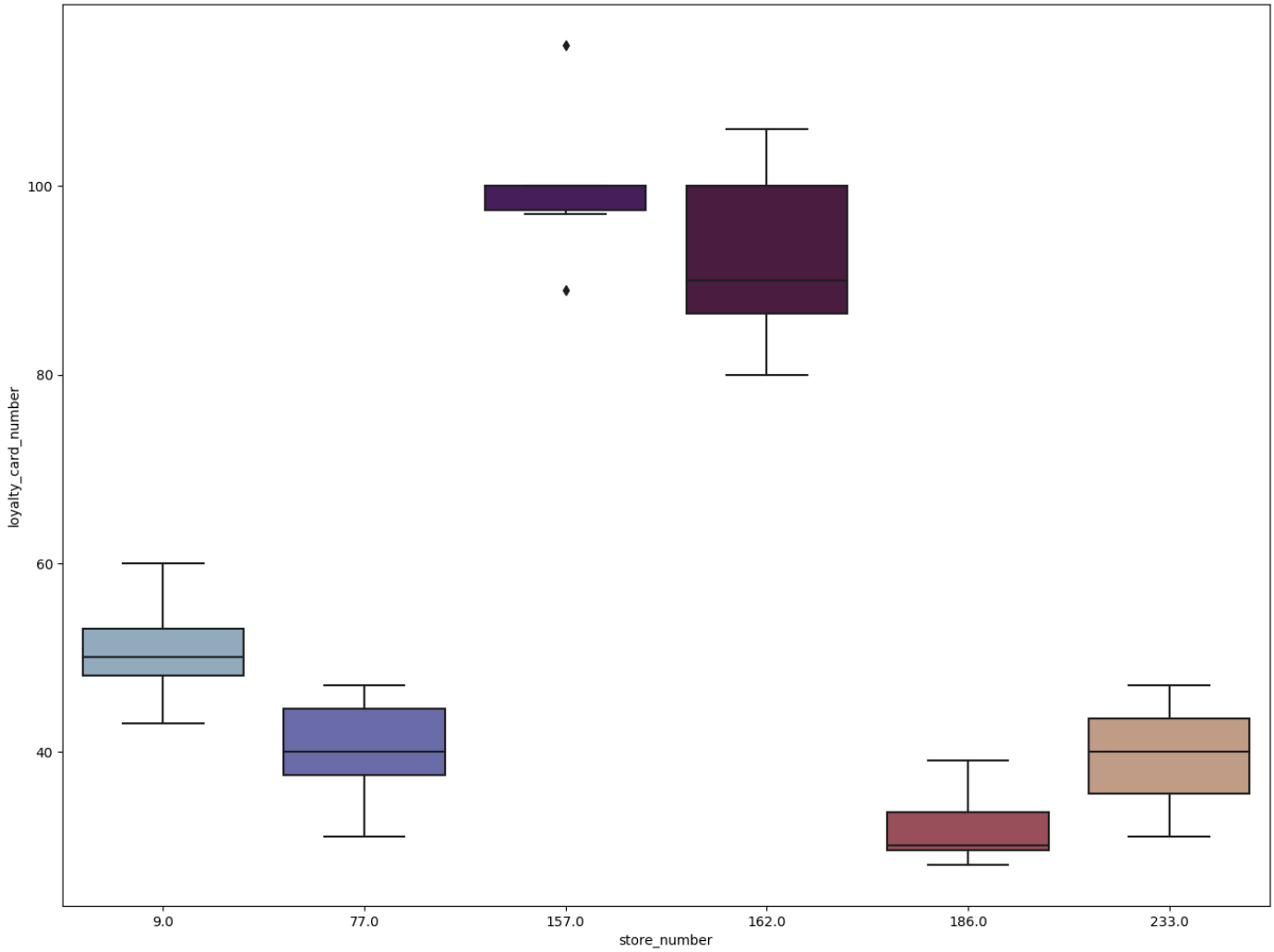
Out[66]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9ffe90b640>



Number of unique customers

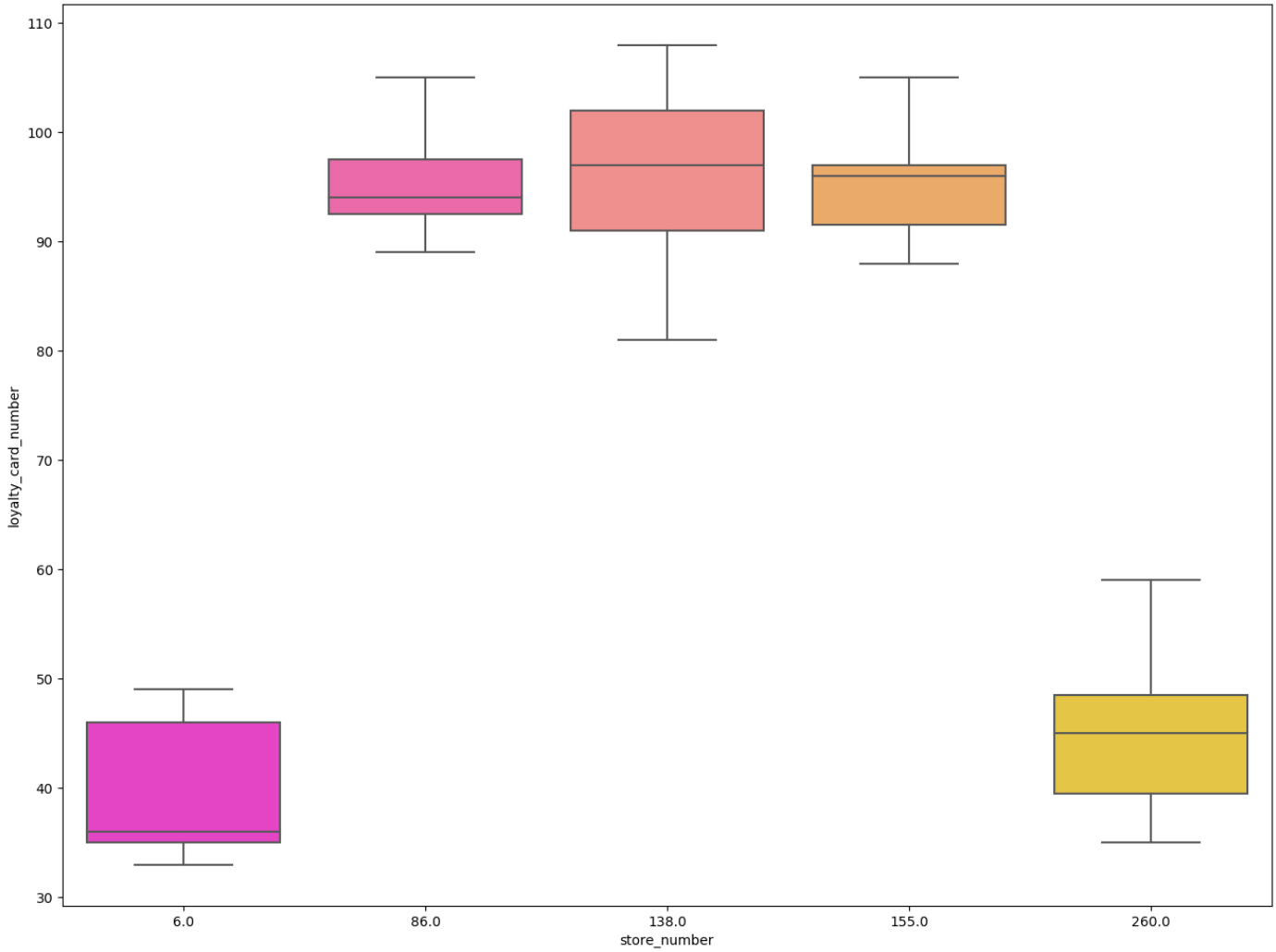
In [67]:
```
 # store 77
sns.boxplot(data = number_of_customers.\
             query('store_number == 77 or store_number == 9 or store_number == 162\
             or store_number == 157 or store_number == 233 or store_number == 186'),\
             x='store_number', y='loyalty_card_number', palette='twilight')
```

Out[67]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffe8a5bb0>
```



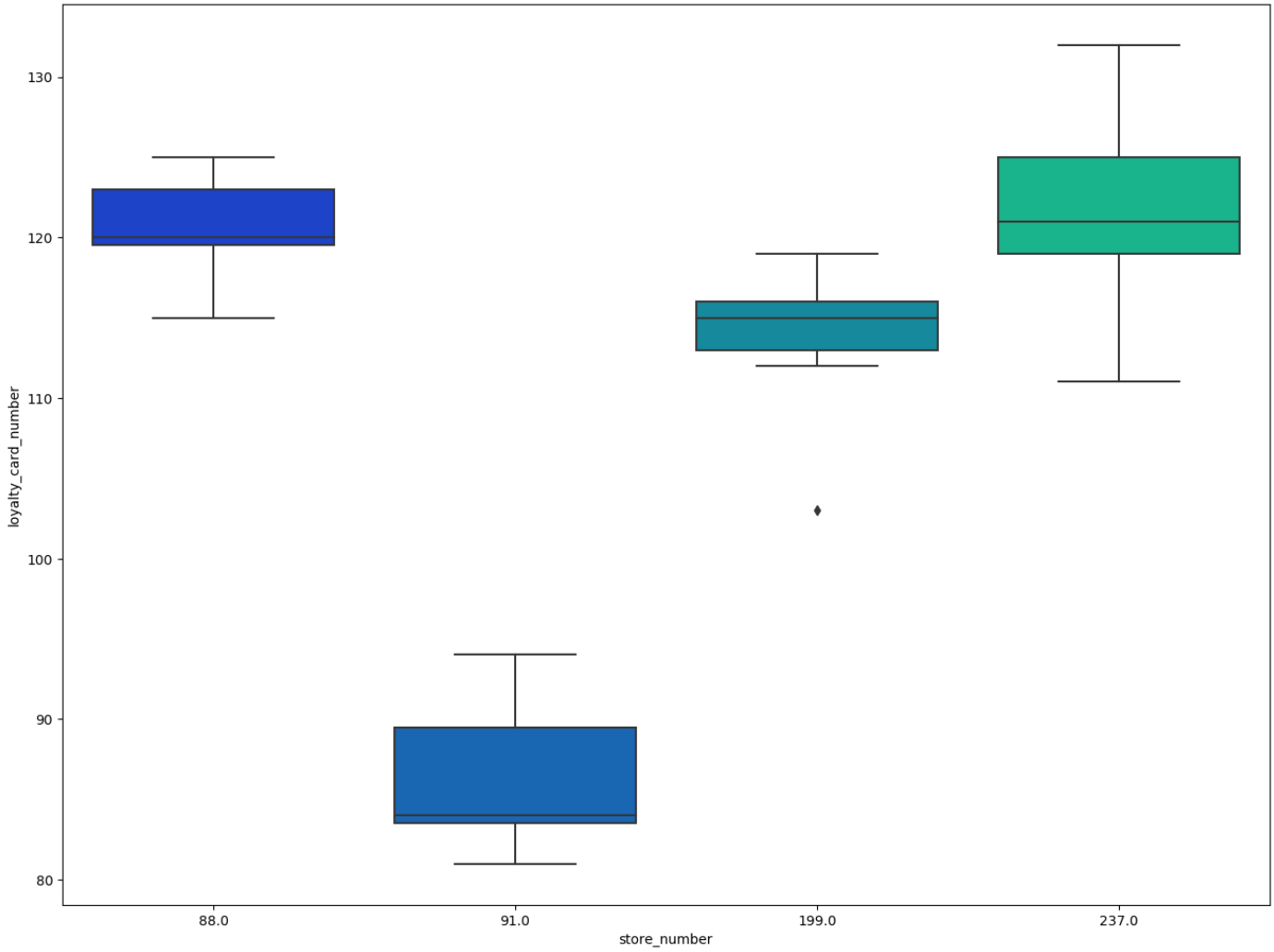
In [68]:
```
# 86 store
sns.boxplot(data = number_of_customers.\
             query('store_number == 86 or store_number == 6 \
        or store_number == 155 or store_number == 138 or store_number == 260'),\
             x='store_number', y='loyalty_card_number', palette='spring')
```

Out[68]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffe7e25b0>
```



In [69]:
```
# 88 store
sns.boxplot(data = number_of_customers.\
             query('store_number == 88 or store_number == 91\
             or store_number == 237\
             or store_number == 199'),\
             x='store_number', y='loyalty_card_number', palette='winter')
```

Out[69]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9ffe6fb610>



We found similar to 77 store - 233.0, similar to 86 store - they are [155, 138], and 1 similar to 88 store - 237.

The trial period goes from the start of February 2019 to April 2019. We now want to see if there has been an uplift in overall chip sales.

Stores 77, 86 and 88 total_sales and number of customers uplift research

77 sales

In [70]:
```
# 77 store pretrial scaling factor
pretrial_sales_77 = df.query('store_number == 77 and all_dates < "2019-02-01"').total_sa
pretrial_sales_233 = df.query('store_number == 233 and all_dates < "2019-02-01"').total_
scaling_factor_pretrial_77_233 = pretrial_sales_77/pretrial_sales_233
scaling_factor_pretrial_77_233
```

Out[70]:
```
1.0397523623330074
```

In [71]:
```
# Apply the scaling factor
sales_233_scaled = df.query('store_number == 233')[['total_sales', 'all_dates', 'month']
groupby(['all_dates','month'], as_index=False).agg({'total_sales': 'sum'})
sales_233_scaled['total_sales'] = sales_233_scaled.total_sales*scaling_factor_pretrial_7
```

In [72]:
```
# Calculate the percentage difference between scaled control sales and trial sales
both_233_77 = sales_233_scaled.groupby('month', as_index=False).agg({'total_sales': 'sum
                              groupby(['month'], as_index=False).agg({'total_sales': 'su
                              on='month')
both_233_77['percent_diff'] = abs(both_233_77.total_sales_x - both_233_77.total_sales_y)
```

As our null hypothesis is that the trial period is the same as the pre-trial period, let's take the standard deviation based on the scaled percentage difference in the pre-trial period

In [73]:
```
from scipy.stats import ttest_ind, tstd, t
```

In [74]:
```
both_233_77.groupby('month', as_index=False).agg({'percent_diff': 'sum'}).\
                 query('month != "2" and month != "3" and month != "4"').\
                 percent_diff.values
```

Out[74]:
```
array([0.20396672, 0.17307859, 0.04746491, 0.07519571, 0.16088022,
       0.20391806, 0.04816228, 0.08692962, 0.05608335])
```

In [75]:
```
stand_dev = tstd(both_233_77.groupby(['month'], as_index=False).\
                 agg({'percent_diff': 'sum'}).\
                 query('month != "2" and month != "3" and month != "4"').\
                 percent_diff.values)
stand_dev
```

Out[75]:
```
0.06719734895050541
```

note that there are 8 months in the pre-trial period hence 8 - 1 = 7 degrees of freedom

In [76]:
```
deg_free = 7
```

In [77]:
```
t_values_both_233_77 = both_233_77.groupby('month', as_index=False).agg({'percent_diff':
t_values_both_233_77['t_values'] = t_values_both_233_77.percent_diff/stand_dev
t_values_both_233_77
```

Out[77]:

	month	percent_diff	t_values
0	1	0.203967	3.035339
1	10	0.173079	2.575676
2	11	0.047465	0.706351
3	12	0.075196	1.119028
4	2	0.077889	1.159102
5	3	0.358510	5.335179
6	4	0.721444	10.736193
7	5	0.160880	2.394145
8	6	0.203918	3.034615
9	7	0.048162	0.716729
10	8	0.086930	1.293647
11	9	0.056083	0.834607

In [78]:
```
deg_free = 7
t.interval(confidence=0.95, df=deg_free, loc=0, scale=1)
```

Out[78]:
```
(-2.3646242510102993, 2.3646242510102993)
```

We can observe that the t-value is much larger than the 95th percentile value of the t-distribution for March and April - i.e. the increase in sales in the trial store in March and April is statistically greater than in the control store

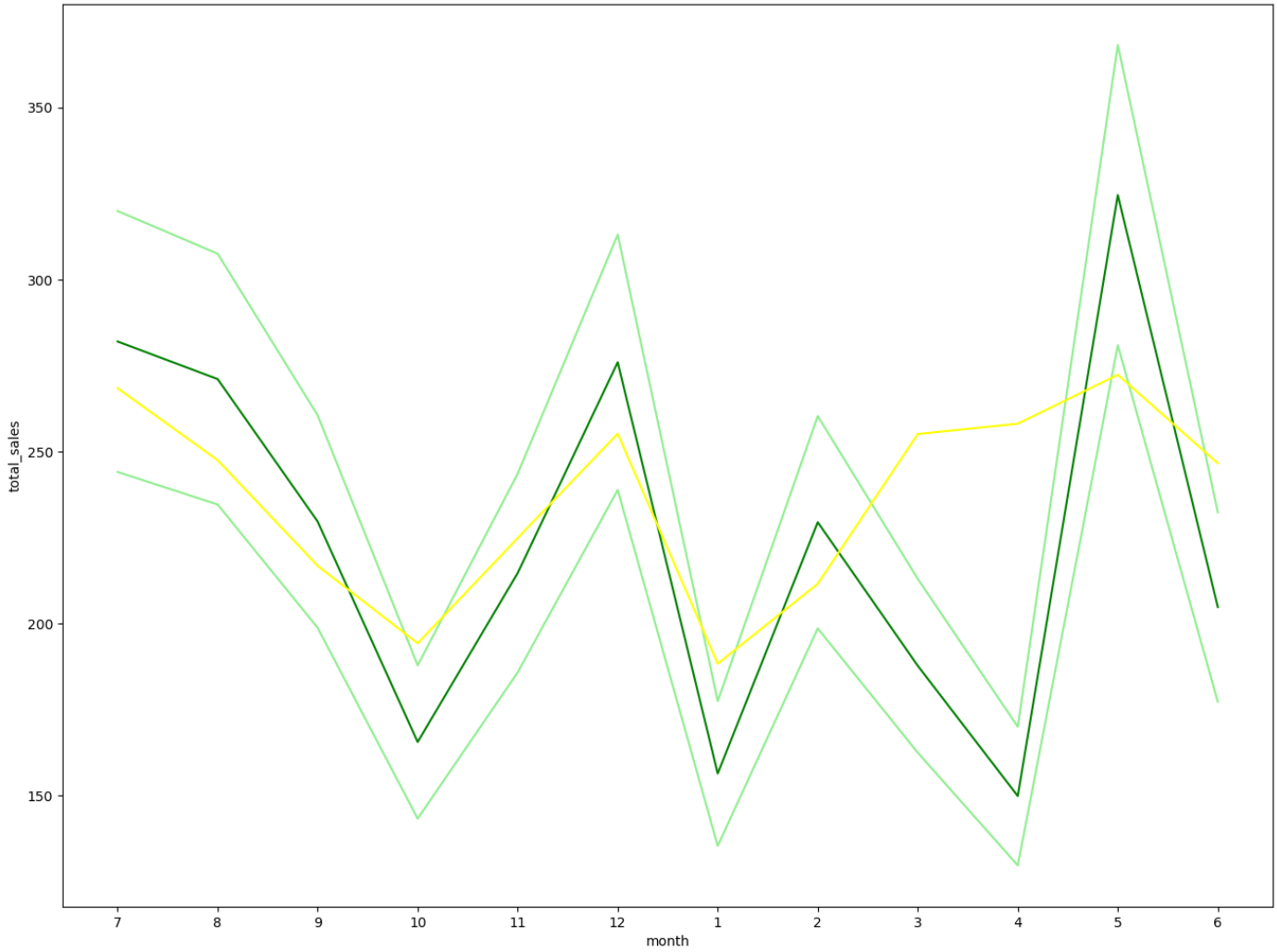
In [79]:
```
for_plot_233_77 = sales_233_scaled.\
                    groupby(['month'], as_index=False).\
                    agg({'total_sales': 'sum'})
for_plot_233_77['95ql'] = for_plot_233_77.total_sales *(1+stand_dev*2)
for_plot_233_77['5ql'] = for_plot_233_77.total_sales*(1-stand_dev*2)

for_plot_233_77['77'] = df.query('store_number == 77').\
groupby(['month'], as_index=False).\
agg({'total_sales': 'sum'}).total_sales
for_plot_233_77 = for_plot_233_77.reindex(index = [9,10,11,1,2,1,3,0,4,5,6,7,8])
```

Lineplot for a whole period to show difference between control - 233 and trial - 77 stores

In [125…
```
sns.lineplot(data=for_plot_233_77, x='month', y='total_sales', color='green')
sns.lineplot(data=for_plot_233_77, x='month', y='95ql', color='lightgreen')
sns.lineplot(data=for_plot_233_77, x='month', y='5ql', color='lightgreen')
sns.lineplot(data=for_plot_233_77, x='month', y='77', color='yellow')
```

```
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```

Out[125]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffcb4f0a0>
```



The results show that the trial figures in store 77 is significantly different to its control store in the trial period as the trial store performance lies outside the 5% to 95% confidence interval of the control store in two of the three trial months

77 customers

In [81]:
```
# 77 store pretrial scaling factor
pretrial_cust_77 = df.query('store_number == 77 and all_dates < "2019-02-01"').loyalty_c
pretrial_cust_233 = df.query('store_number == 233 and all_dates < "2019-02-01"').loyalty
scaling_factor_pretrial_cust_77_233 = pretrial_cust_77/pretrial_cust_233
scaling_factor_pretrial_cust_77_233
```

Out[81]: 1.0

In [82]:
```
# Apply the scaling factor
sales_cust_233_scaled = df.query('store_number == 233')[['loyalty_card_number', 'month']
                        groupby(['month'], as_index=False).\
                        loyalty_card_number.nunique()

sales_cust_233_scaled['scaled_number'] = sales_cust_233_scaled.\
                       loyalty_card_number*scaling_factor_pretrial_cust_77_233
```

In [83]:
```
# Calculate the percentage difference between scaled control sales and trial sales
```

```
both_cust_233_77 = sales_cust_233_scaled.merge(df.query('store_number == 77')[['loyalty_
                        groupby(['month'], as_index=False).\
                        loyalty_card_number.nunique(), \
                        on='month')
both_cust_233_77['percent_diff'] =\
                abs(both_cust_233_77.loyalty_card_number_x - both_cust_233_77.loyalty_c
```

In [84]:
```
both_cust_233_77.groupby('month', as_index=False).agg({'percent_diff': 'sum'}).\
                query('month != "2" and month != "3" and month != "4"').\
                percent_diff.values
```

Out[84]:
```
array([0.        , 0.125     , 0.        , 0.        , 0.01851852,
       0.11764706, 0.        , 0.04545455, 0.        ])
```

In [85]:
```
stand_dev = tstd(both_cust_233_77.groupby(['month'], as_index=False).\
                agg({'percent_diff': 'sum'}).\
                query('month != "2" and month != "3" and month != "4"').\
                percent_diff.values)
stand_dev
```

Out[85]:
```
0.051755036714105494
```

In [86]:
```
t_values_both_cust_233_77 = both_cust_233_77.groupby('month', as_index=False).agg({'perc
t_values_both_cust_233_77['t_values'] = t_values_both_cust_233_77.percent_diff/stand_dev
t_values_both_cust_233_77
```

Out[86]:

	month	percent_diff	t_values
0	1	0.000000	0.000000
1	10	0.125000	2.415224
2	11	0.000000	0.000000
3	12	0.000000	0.000000
4	2	0.047619	0.920085
5	3	0.314286	6.072563
6	4	0.740741	14.312438
7	5	0.018519	0.357811
8	6	0.117647	2.273152
9	7	0.000000	0.000000
10	8	0.045455	0.878263
11	9	0.000000	0.000000

In [87]:
```
deg_free = 7
t.interval(confidence=0.95, df=deg_free, loc=0, scale=1)
```

Out[87]:
```
(-2.3646242510102993, 2.3646242510102993)
```

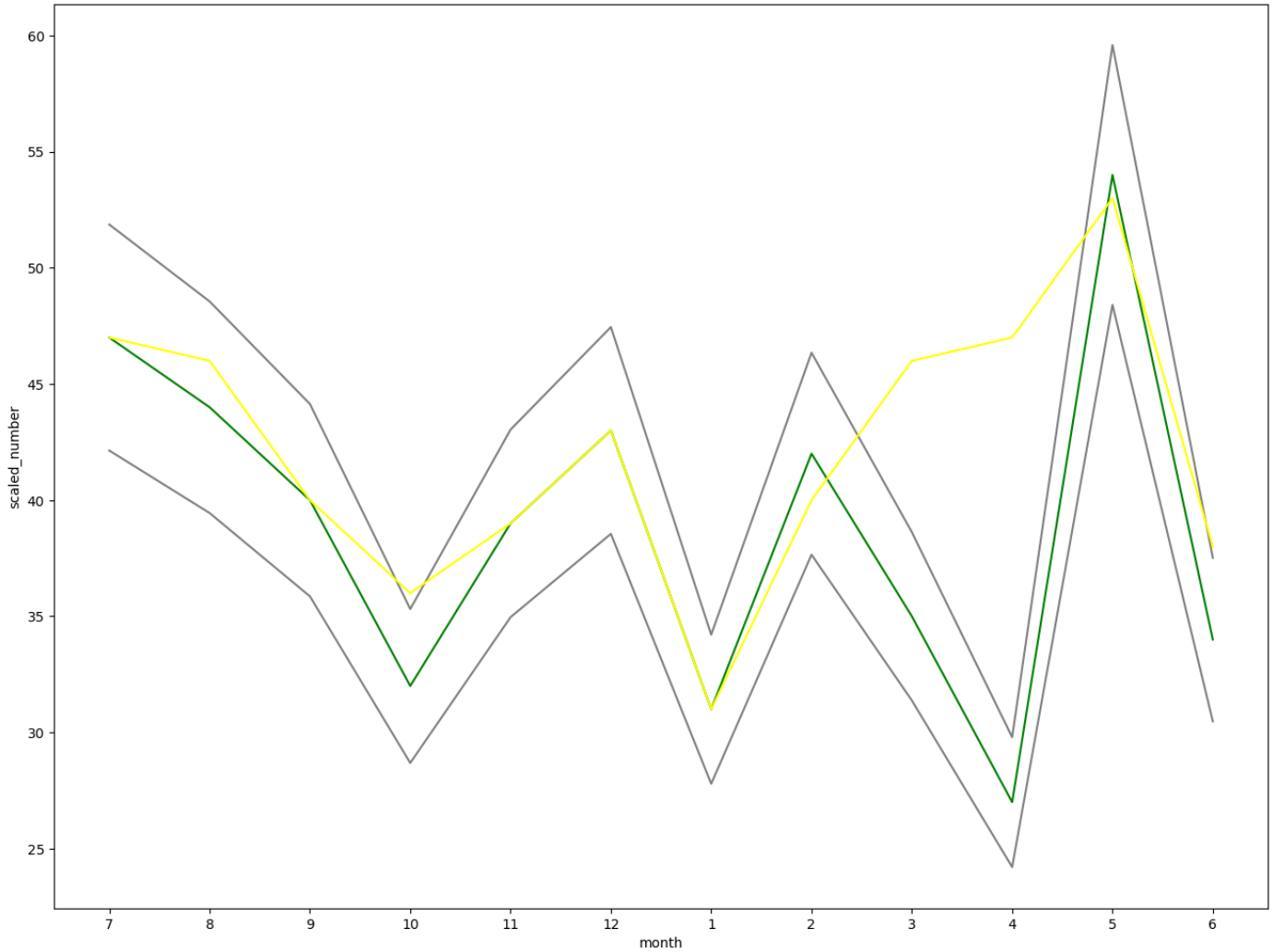
In [88]:
```
for_plot_cust_233_77 = sales_cust_233_scaled
for_plot_cust_233_77['95ql'] = for_plot_cust_233_77.scaled_number *(1+stand_dev*2)
for_plot_cust_233_77['5ql'] = for_plot_cust_233_77.scaled_number*(1-stand_dev*2)

for_plot_cust_233_77['77'] = df.query('store_number == 77').\
                             groupby(['month'], as_index=False).loyalty_card_number.nuni
for_plot_cust_233_77 = for_plot_cust_233_77.reindex(index = [9,10,11,1,2,1,3,0,4,5,6,7,8
```

Lineplot for a whole period to show difference between control - 233 and trial - 77 stores

In [89]:
```
sns.lineplot(data=for_plot_cust_233_77, x='month', y='scaled_number', color='green')
sns.lineplot(data=for_plot_cust_233_77, x='month', y='95ql', color='grey')
sns.lineplot(data=for_plot_cust_233_77, x='month', y='5ql', color='grey')
sns.lineplot(data=for_plot_cust_233_77, x='month', y='77', color='yellow')
```

```
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```

Out[89]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffe54f400>
```



We can see that increase of customers was significant too

86 sales

In [90]:
```
# 86 store pretrial scaling factor
pretrial_sales_86 = df.query('store_number == 86 and all_dates < "2019-02-01"').total_sa
pretrial_sales_155 = df.query('store_number == 155 and all_dates < "2019-02-01"').total_
scaling_factor_pretrial_86_155 = pretrial_sales_86/pretrial_sales_155
scaling_factor_pretrial_86_155
```

Out[90]:
```
0.9720493769183033
```

In [91]:
```
# Apply the scaling factor
sales_155_scaled = df.query('store_number == 155')[['total_sales', 'all_dates', 'month']
groupby(['all_dates','month'], as_index=False).agg({'total_sales': 'sum'})
sales_155_scaled['total_sales'] = sales_155_scaled.total_sales*scaling_factor_pretrial_8
```

In [92]:
```
# Calculate the percentage difference between scaled control sales and trial sales
both_155_86 = sales_155_scaled.groupby('month', as_index=False).\
                               agg({'total_sales': 'sum'}).\
                               merge(df.query('store_number == 86')[['total_sales', 'mo
                              groupby(['month'], as_index=False).agg({'total_sales': 'su
                              on='month')
both_155_86['percent_diff'] = abs(both_155_86.total_sales_x - both_155_86.total_sales_y)
```

In [93]:
```
stand_dev = tstd(both_155_86.groupby(['month'], as_index=False).\
```

```
                    agg({'percent_diff': 'sum'}).\
                    query('month != "2" and month != "3" and month != "4"').\
                    percent_diff.values)
stand_dev
```

Out[93]:
```
0.0191205988044067
```

In [94]:
```
t_values_both_155_86 = both_155_86.groupby('month', as_index=False).agg({'percent_diff':
t_values_both_155_86['t_values'] = t_values_both_155_86.percent_diff/stand_dev
t_values_both_155_86
```

Out[94]:

	month	percent_diff	t_values
0	1	0.013155	0.688009
1	10	0.011646	0.609078
2	11	0.048713	2.547692
3	12	0.044704	2.338002
4	2	0.055356	2.895091
5	3	0.268037	14.018230
6	4	0.033381	1.745835
7	5	0.014565	0.761733
8	6	0.035786	1.871582
9	7	0.027904	1.459353
10	8	0.012251	0.640742
11	9	0.063873	3.340532

In [95]:
```
deg_free = 7
t.interval(confidence=0.95, df=deg_free, loc=0, scale=1)
```

Out[95]:
```
(-2.3646242510102993, 2.3646242510102993)
```

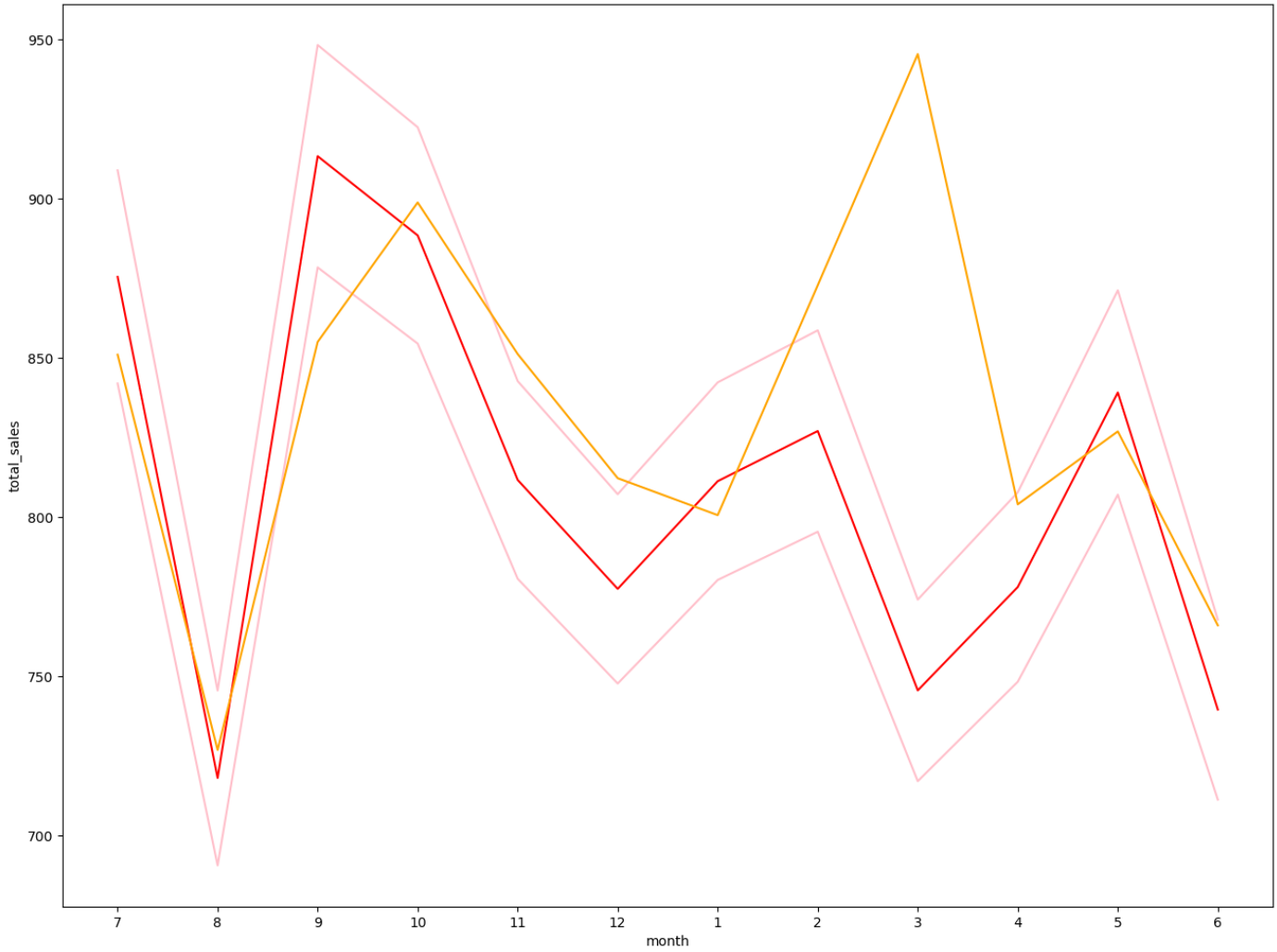
In [96]:
```
for_plot_155_86 = sales_155_scaled.\
                    groupby(['month'], as_index=False).\
                    agg({'total_sales': 'sum'})
for_plot_155_86['95ql'] = for_plot_155_86.total_sales *(1+stand_dev*2)
for_plot_155_86['5ql'] = for_plot_155_86.total_sales*(1-stand_dev*2)

for_plot_155_86['86'] = df.query('store_number == 86').\
groupby(['month'], as_index=False).\
agg({'total_sales': 'sum'}).total_sales
for_plot_155_86 = for_plot_155_86.reindex(index = [9,10,11,1,2,1,3,0,4,5,6,7,8])
```

Lineplot for a whole period to show difference between control - 155 and trial - 86 stores

In [97]:
```
sns.lineplot(data=for_plot_155_86, x='month', y='total_sales', color='red')
sns.lineplot(data=for_plot_155_86, x='month', y='95ql', color='pink')
sns.lineplot(data=for_plot_155_86, x='month', y='5ql', color='pink')
sns.lineplot(data=for_plot_155_86, x='month', y='86', color='orange')
```

```
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```

Out[97]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffe52cc40>
```



The results show that the trial figures in store 86 is not significantly different to its control store in the trial period as the trial store performance lies outside the 5% to 95% confidence interval of the control store in one of the three trial months.

86 customers

In [98]:
```
# 86 store pretrial scaling factor
pretrial_cust_86 = df.query('store_number == 86 and all_dates < "2019-02-01"').loyalty_c
pretrial_cust_155 = df.query('store_number == 155 and all_dates < "2019-02-01"').loyalty
scaling_factor_pretrial_cust_86_155 = pretrial_cust_86/pretrial_cust_155
scaling_factor_pretrial_cust_86_155
```

Out[98]:
```
1.037344398340249
```

In [99]:
```
# Apply the scaling factor
sales_cust_155_scaled = df.query('store_number == 155')[['loyalty_card_number', 'month']
                        groupby(['month'], as_index=False).\
                        loyalty_card_number.nunique()

sales_cust_155_scaled['scaled_number'] = sales_cust_155_scaled.\
                       loyalty_card_number*scaling_factor_pretrial_cust_86_155
```

In [100…
```
# Calculate the percentage difference between scaled control sales and trial sales
```

```
both_cust_155_86 = sales_cust_155_scaled.merge(df.query('store_number == 86')[['loyalty_
                        groupby(['month'], as_index=False).\
                        loyalty_card_number.nunique(), \
                        on='month')
both_cust_155_86['percent_diff'] =\
                abs(both_cust_155_86.loyalty_card_number_x - both_cust_155_86.loyalty_c
```

In [101…
```
both_cust_155_86.groupby('month', as_index=False).agg({'percent_diff': 'sum'}).\
                query('month != "2" and month != "3" and month != "4"').\
                percent_diff.values
```

Out[101]:
```
array([0.0326087 , 0.        , 0.01041667, 0.02197802, 0.01980198,
       0.05747126, 0.04081633, 0.04545455, 0.04166667])
```

In [102…
```
stand_dev = tstd(both_cust_155_86.groupby(['month'], as_index=False).\
                agg({'percent_diff': 'sum'}).\
                query('month != "2" and month != "3" and month != "4"').\
                percent_diff.values)
stand_dev
```

Out[102]:
```
0.018389114484539407
```

In [103…
```
t_values_both_cust_155_86 = both_cust_155_86.groupby('month', as_index=False).agg({'perc
t_values_both_cust_155_86['t_values'] = t_values_both_cust_155_86.percent_diff/stand_dev
t_values_both_cust_155_86
```

Out[103]:

	month	percent_diff	t_values
0	1	0.032609	1.773261
1	10	0.000000	0.000000
2	11	0.010417	0.566458
3	12	0.021978	1.195165
4	2	0.141304	7.684130
5	3	0.186813	10.158901
6	4	0.064516	3.508387
7	5	0.019802	1.076832
8	6	0.057471	3.125287
9	7	0.040816	2.219592
10	8	0.045455	2.471818
11	9	0.041667	2.265833

In [104…
```
deg_free = 7
t.interval(confidence=0.95, df=deg_free, loc=0, scale=1)
```

Out[104]:
```
(-2.3646242510102993, 2.3646242510102993)
```

In [105…
```
for_plot_cust_155_86 = sales_cust_155_scaled
for_plot_cust_155_86['95ql'] = for_plot_cust_155_86.scaled_number *(1+stand_dev*2)
for_plot_cust_155_86['5ql'] = for_plot_cust_155_86.scaled_number*(1-stand_dev*2)

for_plot_cust_155_86['86'] = df.query('store_number == 86').\
                            groupby(['month'], as_index=False).loyalty_card_number.nuni
for_plot_cust_155_86 = for_plot_cust_155_86.reindex(index = [9,10,11,1,2,1,3,0,4,5,6,7,8
```

Lineplot for a whole period to show difference between control - 155 and trial - 86 stores

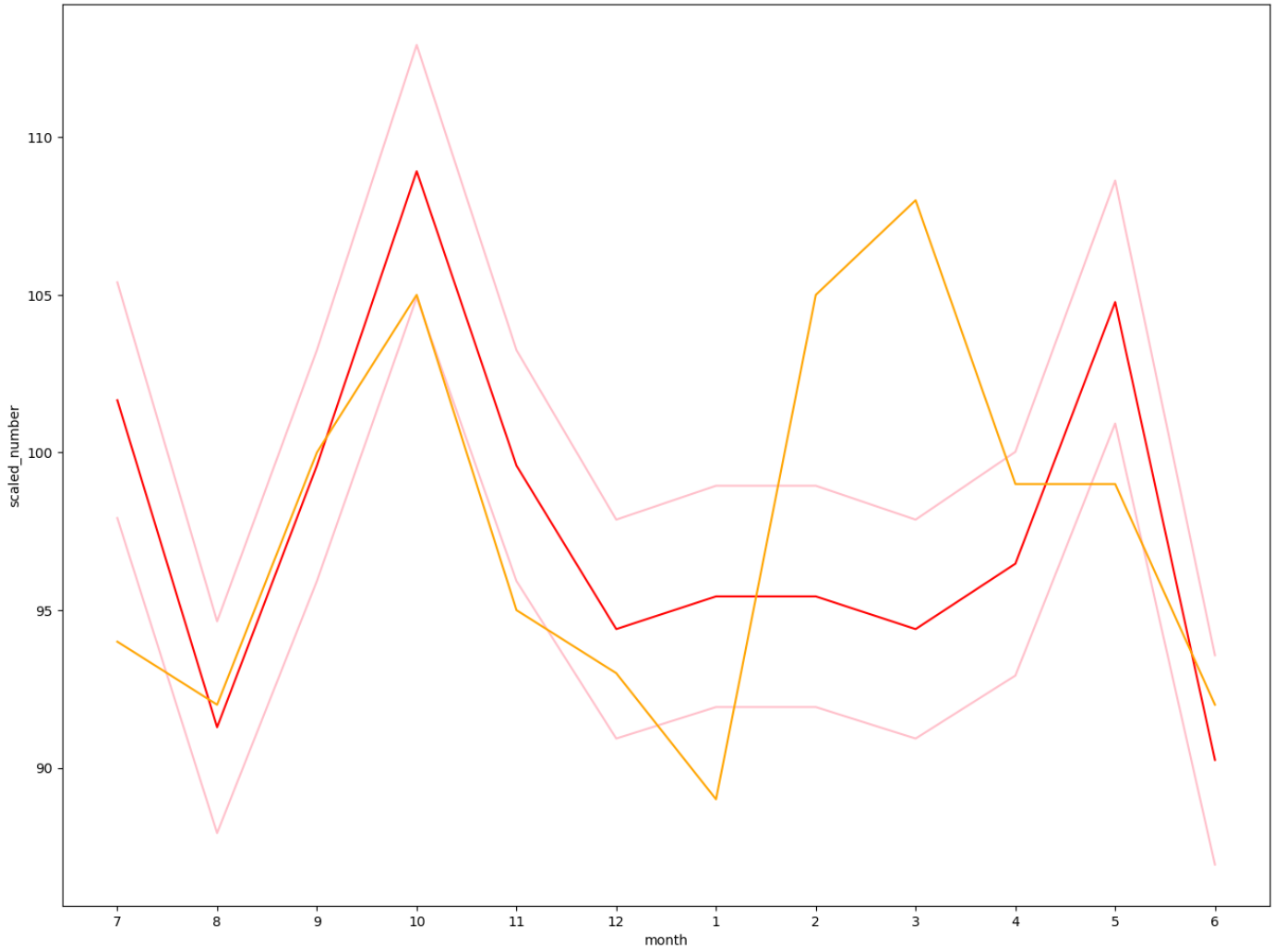
In [106…

```
sns.lineplot(data=for_plot_cust_155_86, x='month', y='scaled_number', color='red')
sns.lineplot(data=for_plot_cust_155_86, x='month', y='95ql', color='pink')
sns.lineplot(data=for_plot_cust_155_86, x='month', y='5ql', color='pink')
sns.lineplot(data=for_plot_cust_155_86, x='month', y='86', color='orange')
```

```
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```

Out[106]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffcc77a00>
```



It looks like the number of customers is significantly higher in all of the three months.

88 sales

In [107…
```
# 88 store pretrial scaling factor
pretrial_sales_88 = df.query('store_number == 88 and all_dates < "2019-02-01"').total_sa
pretrial_sales_237 = df.query('store_number == 237 and all_dates < "2019-02-01"').total_
scaling_factor_pretrial_88_237 = pretrial_sales_88/pretrial_sales_237
scaling_factor_pretrial_88_237
```

Out[107]: 0.9867752464919073

In [108…
```
# Apply the scaling factor
sales_237_scaled = df.query('store_number == 237')[['total_sales', 'all_dates', 'month']
groupby(['all_dates','month'], as_index=False).agg({'total_sales': 'sum'})
sales_237_scaled['total_sales'] = sales_237_scaled.total_sales*scaling_factor_pretrial_8
```

In [109…
```
# Calculate the percentage difference between scaled control sales and trial sales
both_237_88 = sales_237_scaled.groupby('month', as_index=False).\
                               agg({'total_sales': 'sum'}).\
                               merge(df.query('store_number == 88')[['total_sales', 'mo
                              groupby(['month'], as_index=False).agg({'total_sales': 'su
                              on='month')
both_237_88['percent_diff'] = abs(both_237_88.total_sales_x - both_237_88.total_sales_y)
```

In [110…
```
stand_dev = tstd(both_237_88.groupby(['month'], as_index=False).\
                 agg({'percent_diff': 'sum'}).\
                 query('month != "2" and month != "3" and month != "4"').\
                 percent_diff.values)
stand_dev
```

Out[110]: 0.04957564998781151

In [111…
```
t_values_both_237_88 = both_237_88.groupby('month', as_index=False).agg({'percent_diff':
t_values_both_237_88['t_values'] = t_values_both_237_88.percent_diff/stand_dev
t_values_both_237_88
```

Out[111]:

	month	percent_diff	t_values
0	1	0.101985	2.057162
1	10	0.006361	0.128300
2	11	0.009860	0.198882
3	12	0.004167	0.084047
4	2	0.033932	0.684458
5	3	0.307543	6.203515
6	4	0.150267	3.031074
7	5	0.103026	2.078156
8	6	0.110187	2.222595
9	7	0.123209	2.485264
10	8	0.047698	0.962126
11	9	0.096205	1.940564

In [112…
```
deg_free = 7
t.interval(confidence=0.95, df=deg_free, loc=0, scale=1)
```

Out[112]: (-2.3646242510102993, 2.3646242510102993)

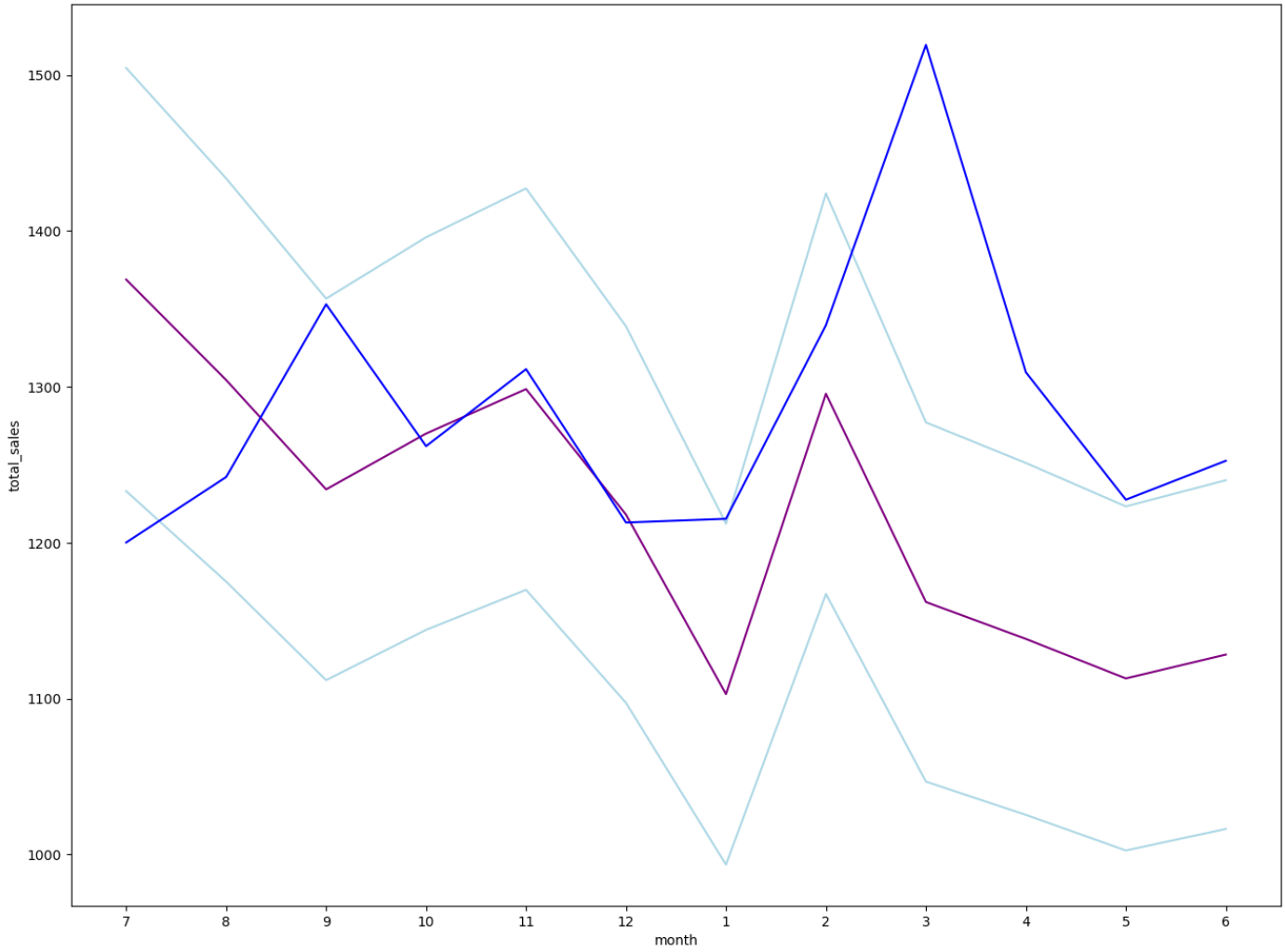
In [113…
```
for_plot_237_88 = sales_237_scaled.\
                    groupby(['month'], as_index=False).\
                    agg({'total_sales': 'sum'})
for_plot_237_88['95ql'] = for_plot_237_88.total_sales *(1+stand_dev*2)
for_plot_237_88['5ql'] = for_plot_237_88.total_sales*(1-stand_dev*2)

for_plot_237_88['88'] = df.query('store_number == 88').\
groupby(['month'], as_index=False).\
agg({'total_sales': 'sum'}).total_sales
for_plot_237_88 = for_plot_237_88.reindex(index = [9,10,11,1,2,1,3,0,4,5,6,7,8])
```

Lineplot for a whole period to show difference between control - 237 and trial - 88 stores

In [114…
```
sns.lineplot(data=for_plot_237_88, x='month', y='total_sales', color='purple')
sns.lineplot(data=for_plot_237_88, x='month', y='95ql', color='lightblue')
sns.lineplot(data=for_plot_237_88, x='month', y='5ql', color='lightblue')
sns.lineplot(data=for_plot_237_88, x='month', y='88', color='blue')
```

```
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```

Out[114]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9ffcb81880>



The results show that the trial in store 88 is significantly different to its control store in the trial period as the trial store performance lies outside of the 5% to 95% confidence interval of the control store in two of the three trial months.

88 customers

In [115…
```
# 88 store pretrial scaling factor
pretrial_cust_88 = df.query('store_number == 88 and all_dates < "2019-02-01"').loyalty_c
pretrial_cust_237 = df.query('store_number == 237 and all_dates < "2019-02-01"').loyalty
scaling_factor_pretrial_cust_88_237 = pretrial_cust_88/pretrial_cust_237
scaling_factor_pretrial_cust_88_237
```

Out[115]: 1.0391061452513966

In [116…
```
# Apply the scaling factor
sales_cust_237_scaled = df.query('store_number == 237')[['loyalty_card_number', 'month']
                        groupby(['month'], as_index=False).\
                        loyalty_card_number.nunique()

sales_cust_237_scaled['scaled_number'] = sales_cust_237_scaled.\
                       loyalty_card_number*scaling_factor_pretrial_cust_88_237
```

In [117…
```
# Calculate the percentage difference between scaled control sales and trial sales
both_cust_237_88 = sales_cust_237_scaled.merge(df.query('store_number == 88')[['loyalty_
```

```
                        groupby(['month'], as_index=False).\
                        loyalty_card_number.nunique(), \
                        on='month')
both_cust_237_88['percent_diff'] =\
                abs(both_cust_237_88.loyalty_card_number_x - both_cust_237_88.loyalty_c
```

In [118…
```
both_cust_237_88.groupby('month', as_index=False).agg({'percent_diff': 'sum'}).\
                query('month != "2" and month != "3" and month != "4"').\
                percent_diff.values
```

Out[118]:
```
array([0.03603604, 0.00847458, 0.016     , 0.00826446, 0.        ,
       0.04237288, 0.016     , 0.0530303 , 0.        ])
```

In [119…
```
stand_dev = tstd(both_cust_237_88.groupby(['month'], as_index=False).\
                agg({'percent_diff': 'sum'}).\
                query('month != "2" and month != "3" and month != "4"').\
                percent_diff.values)
stand_dev
```

Out[119]:
```
0.019206951491390737
```

In [120…
```
t_values_both_cust_237_88 = both_cust_237_88.groupby('month', as_index=False).agg({'perc
t_values_both_cust_237_88['t_values'] = t_values_both_cust_237_88.percent_diff/stand_dev
t_values_both_cust_237_88
```

Out[120]:

	month	percent_diff	t_values
0	1	0.036036	1.876198
1	10	0.008475	0.441224
2	11	0.016000	0.833032
3	12	0.008264	0.430285
4	2	0.025210	1.312550
5	3	0.155172	8.078972
6	4	0.017241	0.897664
7	5	0.000000	0.000000
8	6	0.042373	2.206122
9	7	0.016000	0.833032
10	8	0.053030	2.760995
11	9	0.000000	0.000000

In [121…
```
deg_free = 7
t.interval(confidence=0.95, df=deg_free, loc=0, scale=1)
```

Out[121]:
```
(-2.3646242510102993, 2.3646242510102993)
```

In [122…
```
for_plot_cust_237_88 = sales_cust_237_scaled
for_plot_cust_237_88['95ql'] = for_plot_cust_237_88.scaled_number *(1+stand_dev*2)
for_plot_cust_237_88['5ql'] = for_plot_cust_237_88.scaled_number*(1-stand_dev*2)

for_plot_cust_237_88['88'] = df.query('store_number == 88').\
                              groupby(['month'], as_index=False).loyalty_card_number.nuni
for_plot_cust_237_88 = for_plot_cust_237_88.reindex(index = [9,10,11,1,2,1,3,0,4,5,6,7,8
```

Lineplot for a whole period to show difference between control - 237 and

trial - 88 stores

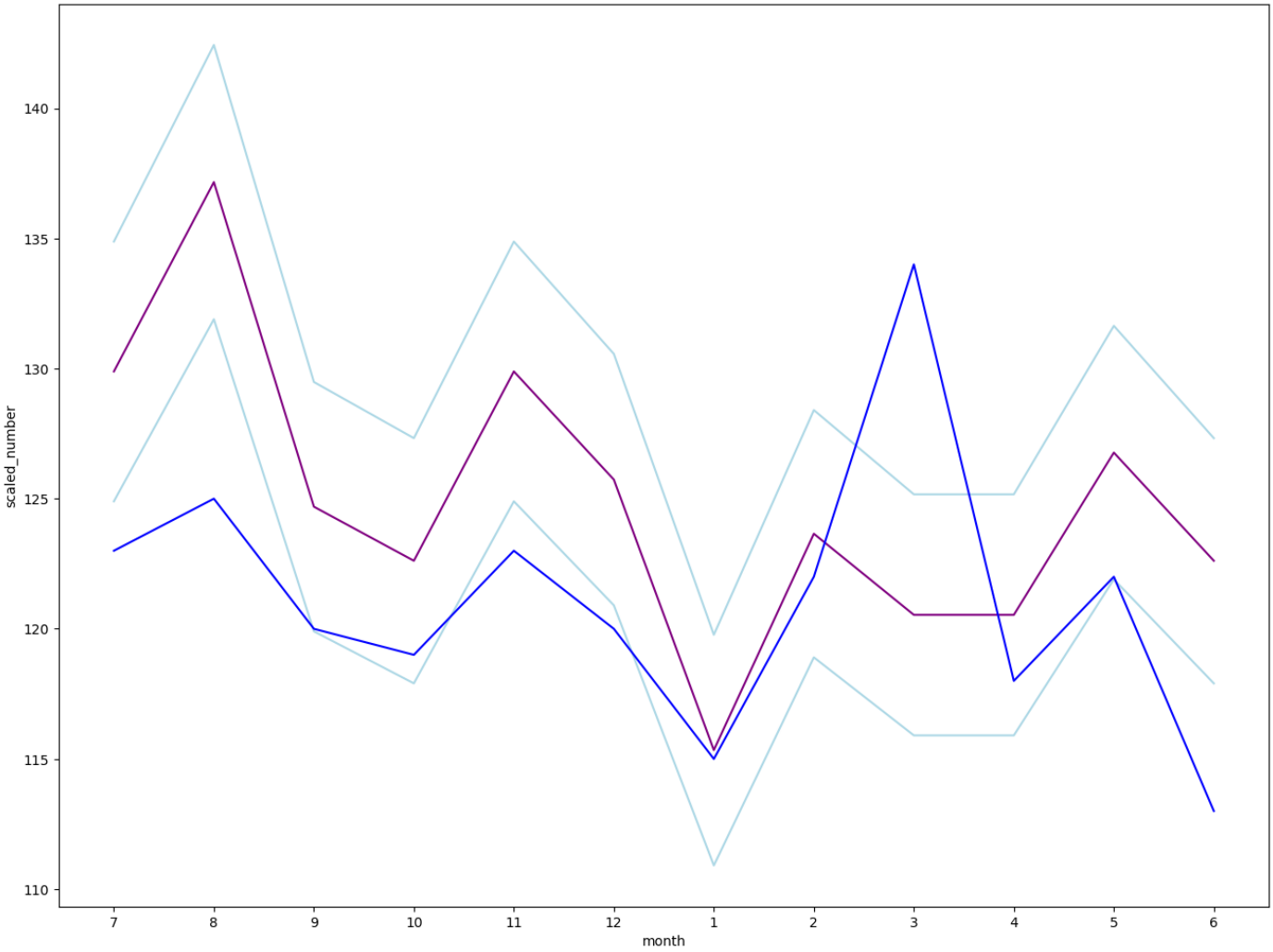
In [123…

```
sns.lineplot(data=for_plot_cust_237_88, x='month', y='scaled_number', color='purple')
sns.lineplot(data=for_plot_cust_237_88, x='month', y='95ql', color='lightblue')
sns.lineplot(data=for_plot_cust_237_88, x='month', y='5ql', color='lightblue')
sns.lineplot(data=for_plot_cust_237_88, x='month', y='88', color='blue')
```

```
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support
for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed i
n a future version.  Convert to a numpy array before indexing instead.
  ndim = x[:, None].ndim
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
/usr/lib/python3/dist-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for
multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will be removed in a
future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```

Out[123]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ffcaef910>
```



Conclusion

We've found control stores 233, 155, 237 for trial stores 77, 86 and 88 respectively. The results for trial stores 77 and 86 during the trial period show a significant difference in at least two of the three trial months but this is not the case for trial store 88. We can check with the client if the implementation of the trial was different in trial store 88 but overall, the trial shows a significant increase in sales. Now that we have finished our analysis, we can prepare our presentation to the Category Manager.

In [124…

```
#!jupyter nbconvert --to webpdf --allow-chromium-download quantium_task2.ipynb
```