911 Calls Capstone Project

For this capstone project, have analyzed some 911 call data from Kaggle (https://www.kaggle.com/mchirico/montcoalert).

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [3]:
```
df=pd.read_csv('911.csv')
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99492 entries, 0 to 99491
Data columns (total 9 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   lat        99492 non-null  float64
 1   lng        99492 non-null  float64
 2   desc       99492 non-null  object 
 3   zip        86637 non-null  float64
 4   title      99492 non-null  object 
 5   timeStamp  99492 non-null  object 
 6   twp        99449 non-null  object 
 7   addr       98973 non-null  object 
 8   e          99492 non-null  int64  
dtypes: float64(3), int64(1), object(5)
memory usage: 6.8+ MB
```

In [5]:
```
df.head()
```

Out[5]:

	lat	lng	desc	zip	title	timeStamp	twp	addr	e
0	40.297876	-75.581294	REINDEER CT & DEAD END; NEW HANOVER; Station ...	19525.0	EMS: BACK PAINS/INJURY	2015-12-10 17:40:00	NEW HANOVER	REINDEER CT & DEAD END	1
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:40:00	HATFIELD TOWNSHIP	BRIAR PATH & WHITEMARSH LN	1
2	40.121182	-75.351975	HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-St...	19401.0	Fire: GAS-ODOR/LEAK	2015-12-10 17:40:00	NORRISTOWN	HAWS AVE	1
3	40.116153	-75.343513	AIRY ST & SWEDE ST; NORRISTOWN; Station 308A;...	19401.0	EMS: CARDIAC EMERGENCY	2015-12-10 17:40:01	NORRISTOWN	AIRY ST & SWEDE ST	1
4	40.251492	-75.603350	CHERRYWOOD CT & DEAD END; LOWER POTTSGROVE; S...	NaN	EMS: DIZZINESS	2015-12-10 17:40:01	LOWER POTTSGROVE	CHERRYWOOD CT & DEAD END	1

Top 5 zipcodes for 911 calls

```
In [6]: df['zip'].value_counts().head(5)
```

```
Out[6]: 19401.0    6979
        19464.0    6643
        19403.0    4854
        19446.0    4748
        19406.0    3174
        Name: zip, dtype: int64
```

The top 5 townships (twp) for 911 calls

```
In [7]: df['twp'].value_counts().head(5)
```

```
Out[7]: LOWER MERION    8443
        ABINGTON        5977
        NORRISTOWN      5890
        UPPER MERION    5227
        CHELTENHAM      4575
        Name: twp, dtype: int64
```

```
In [8]: df['title'].nunique()
```

```
Out[8]: 110
```

In the titles column there are "Reasons/Departments" specified before the title code. New column called "Reason" that contains this string value.

```
In [9]: x=df['title'].iloc[0]
```

```
In [10]: x.split(':')[0]
```

```
Out[10]: 'EMS'
```

```
In [11]: df['Reason']=df['title'].apply(lambda title: title.split(':')[0])
```

```
In [12]: df['Reason'].value_counts()
```

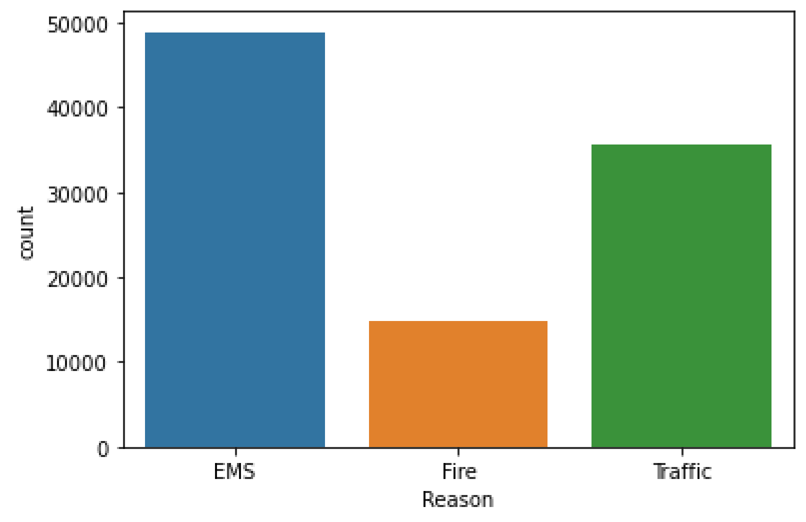
```
Out[12]: EMS        48877
         Traffic    35695
         Fire       14920
         Name: Reason, dtype: int64
```

Most common Reason for a 911 call based off of this new column

countplot of 911 calls by Reason.

```
In [13]: sns.countplot(x='Reason',data=df)

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x2203f79b9b0>
```



data type of the objects in the timeStamp column

```
In [14]: type(df['timeStamp'].iloc[0])

Out[14]: str

In [16]: df['timeStamp']=pd.to_datetime(df['timeStamp'])

In [17]: time = df['timeStamp'].iloc[0]

In [18]: time.hour

Out[18]: 17

In [19]: time

Out[19]: Timestamp('2015-12-10 17:40:00')
```

explore the various attributes you can call. 3 new columns called Hour, Month, and Day of Week

```
In [20]: df['Hour']=df['timeStamp'].apply(lambda time: time.hour)

In [21]: df['Month']=df['timeStamp'].apply(lambda time: time.month)
         df['Day of week']=df['timeStamp'].apply(lambda time: time.dayofweek)

In [22]: dmap = {0:'Mon',1:'Tue',2:'Wed',3:'Thu',4:'Fri',5:'Sat',6:'Sun'}
```

In [23]:
```
df['Day of week']=df['Day of week'].map(dmap)
```

In [24]:
```
df.head()
```

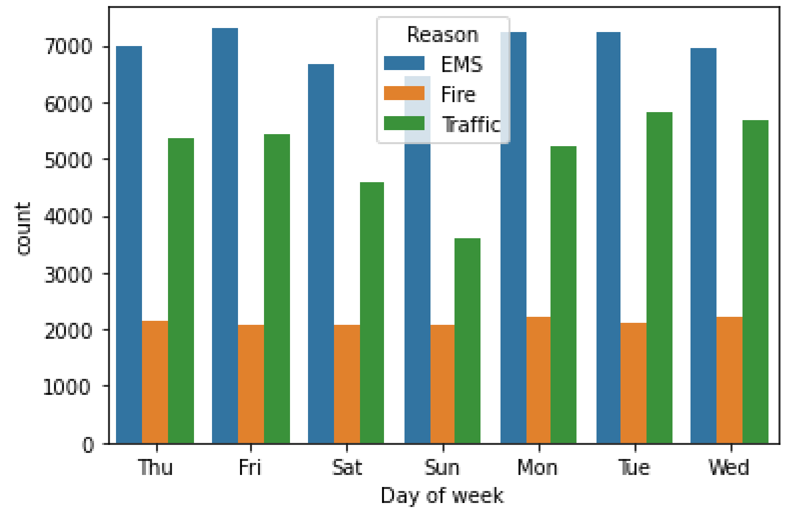
Out[24]:

	lat	lng	desc	zip	title	timeStamp	twp	addr	e	Reason	Hour	Month	Day of week
0	40.297876	-75.581294	REINDEER CT & DEAD END; NEW HANOVER; Station ...	19525.0	EMS: BACK PAINS/INJURY	2015-12-10 17:40:00	NEW HANOVER	REINDEER CT & DEAD END	1	EMS	17	12	Thu
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:40:00	HATFIELD TOWNSHIP	BRIAR PATH & WHITEMARSH LN	1	EMS	17	12	Thu
2	40.121182	-75.351975	HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-St...	19401.0	Fire: GAS-ODOR/LEAK	2015-12-10 17:40:00	NORRISTOWN	HAWS AVE	1	Fire	17	12	Thu
3	40.116153	-75.343513	AIRY ST & SWEDE ST; NORRISTOWN; Station 308A;...	19401.0	EMS: CARDIAC EMERGENCY	2015-12-10 17:40:01	NORRISTOWN	AIRY ST & SWEDE ST	1	EMS	17	12	Thu
4	40.251492	-75.603350	CHERRYWOOD CT & DEAD END; LOWER POTTSGROVE; S...	NaN	EMS: DIZZINESS	2015-12-10 17:40:01	LOWER POTTSGROVE	CHERRYWOOD CT & DEAD END	1	EMS	17	12	Thu

countplot of the Day of Week column with the hue based off of the Reason column.

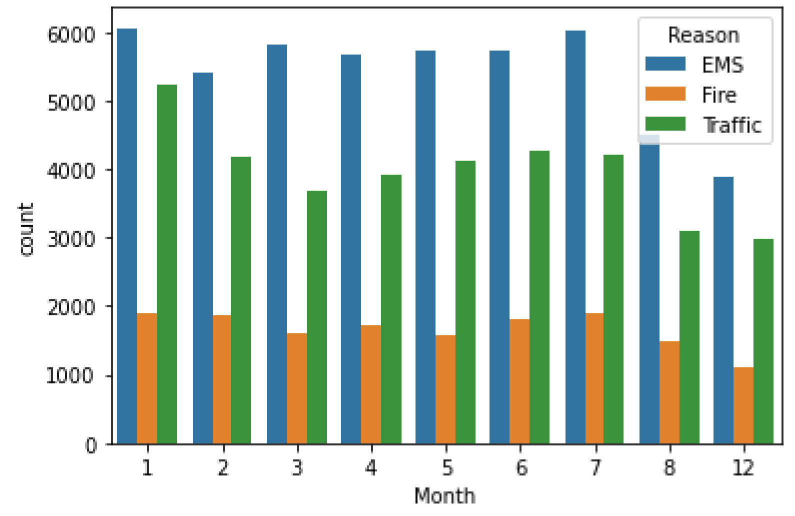
In [25]:
```
sns.countplot(x='Day of week',data=df,hue='Reason')
```

Out[25]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2203fa764a8>
```



In [26]:
```
sns.countplot(x='Month',data=df,hue='Reason')
```

Out[26]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2203fe645f8>
```



In [27]:
```
byMonth=df.groupby('Month').count()
```

In [28]:
```
byMonth
```

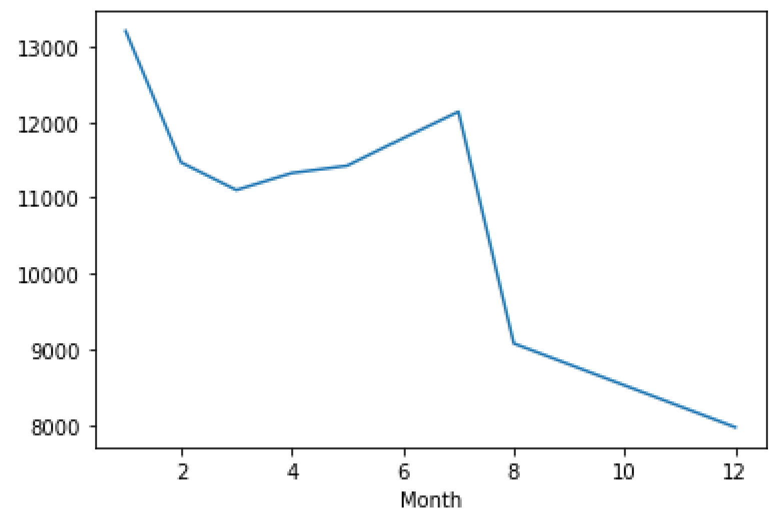
Out[28]:

	lat	lng	desc	zip	title	timeStamp	twp	addr	e	Reason	Hour	Day of week
Month												
1	13205	13205	13205	11527	13205	13205	13203	13096	13205	13205	13205	13205
2	11467	11467	11467	9930	11467	11467	11465	11396	11467	11467	11467	11467
3	11101	11101	11101	9755	11101	11101	11092	11059	11101	11101	11101	11101
4	11326	11326	11326	9895	11326	11326	11323	11283	11326	11326	11326	11326
5	11423	11423	11423	9946	11423	11423	11420	11378	11423	11423	11423	11423
6	11786	11786	11786	10212	11786	11786	11777	11732	11786	11786	11786	11786
7	12137	12137	12137	10633	12137	12137	12133	12088	12137	12137	12137	12137
8	9078	9078	9078	7832	9078	9078	9073	9025	9078	9078	9078	9078
12	7969	7969	7969	6907	7969	7969	7963	7916	7969	7969	7969	7969

simple plot off of the dataframe indicating the count of calls per month.

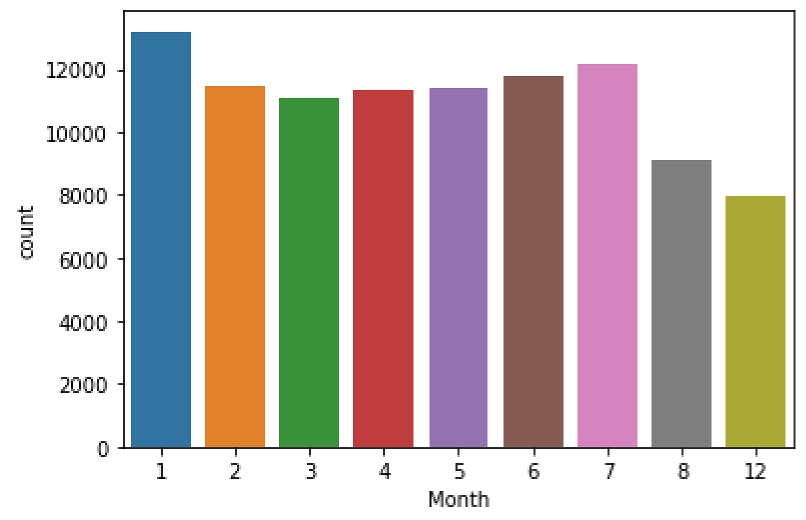
```
In [29]: byMonth['lat'].plot()
```

```
Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x2203fac54a8>
```



```
In [30]: sns.countplot(x='Month',data=df)
```

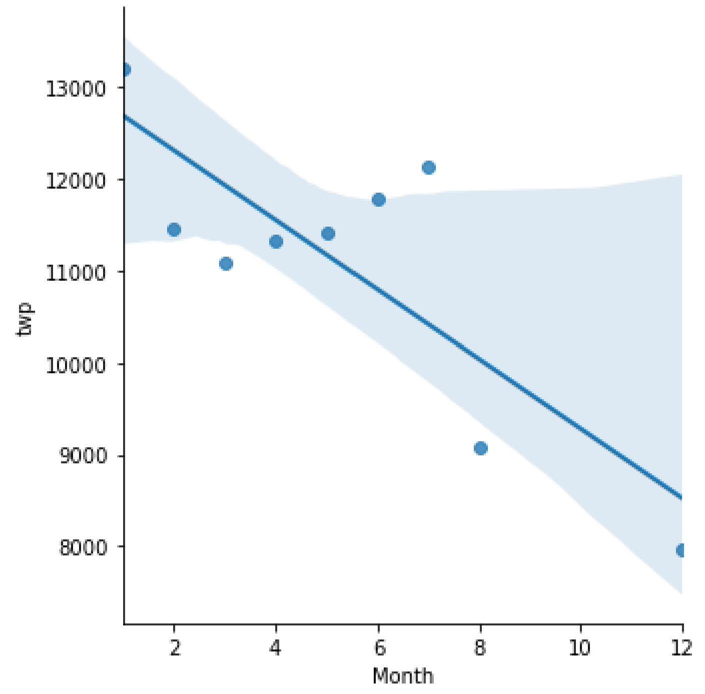
```
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x2203fb39be0>
```



creating a linear fit on the number of calls per month.

```
In [31]: sns.lmplot(x='Month',y='twp',data=byMonth.reset_index())
```

```
Out[31]: <seaborn.axisgrid.FacetGrid at 0x2203fb6acf8>
```



```
In [32]: t=df['timeStamp'].iloc[0]
```

```
In [33]: t.date()
```

```
Out[33]: datetime.date(2015, 12, 10)
```

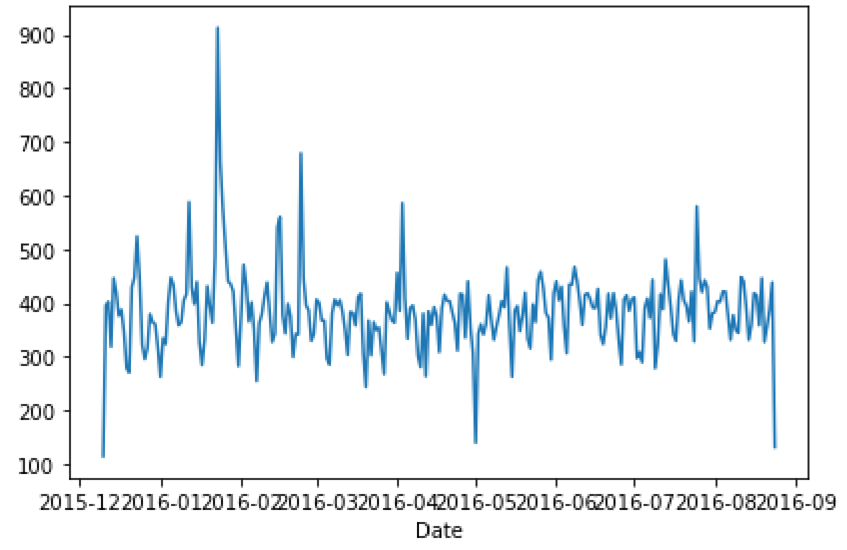
```
In [34]: df['Date']=df['timeStamp'].apply(lambda t: t.date())
```

```
In [35]: df.head()
```

Out[35]:

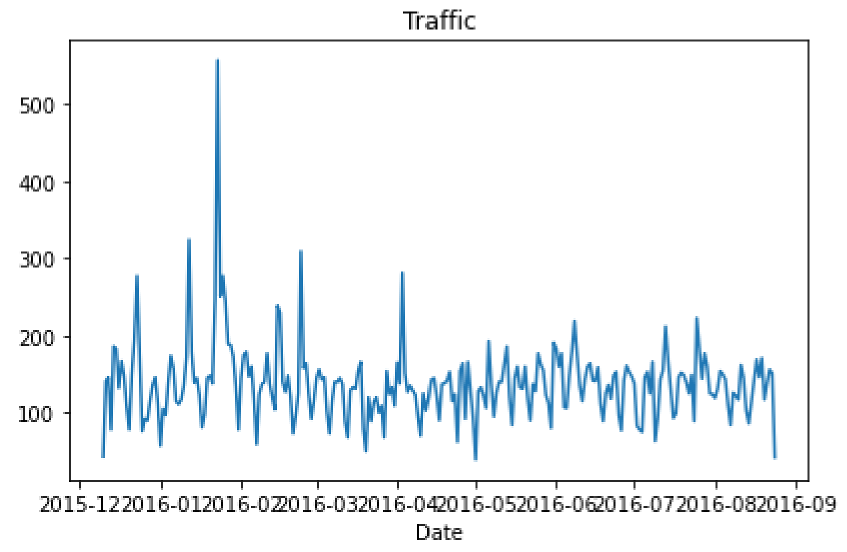
	lat	lng	desc	zip	title	timeStamp	twp	addr	e	Reason	Hour	Month	Day of week	Date
0	40.297876	-75.581294	REINDEER CT & DEAD END; NEW HANOVER; Station ...	19525.0	EMS: BACK PAINS/INJURY	2015-12-10 17:40:00	NEW HANOVER	REINDEER CT & DEAD END	1	EMS	17	12	Thu	2015-12-10
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:40:00	HATFIELD TOWNSHIP	BRIAR PATH & WHITEMARSH LN	1	EMS	17	12	Thu	2015-12-10
2	40.121182	-75.351975	HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-St...	19401.0	Fire: GAS-ODOR/LEAK	2015-12-10 17:40:00	NORRISTOWN	HAWS AVE	1	Fire	17	12	Thu	2015-12-10
3	40.116153	-75.343513	AIRY ST & SWEDE ST; NORRISTOWN; Station 308A;...	19401.0	EMS: CARDIAC EMERGENCY	2015-12-10 17:40:01	NORRISTOWN	AIRY ST & SWEDE ST	1	EMS	17	12	Thu	2015-12-10
4	40.251492	-75.603350	CHERRYWOOD CT & DEAD END; LOWER POTTSGROVE; S...	NaN	EMS: DIZZINESS	2015-12-10 17:40:01	LOWER POTTSGROVE	CHERRYWOOD CT & DEAD END	1	EMS	17	12	Thu	2015-12-10

In [36]:
```
df.groupby('Date').count()['lat'].plot()
plt.tight_layout()
```

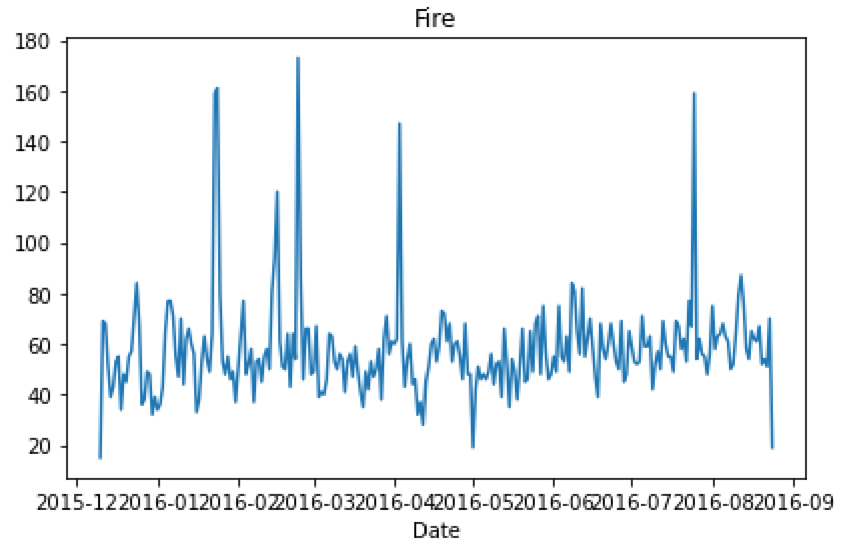


plot of counts of 911 calls.

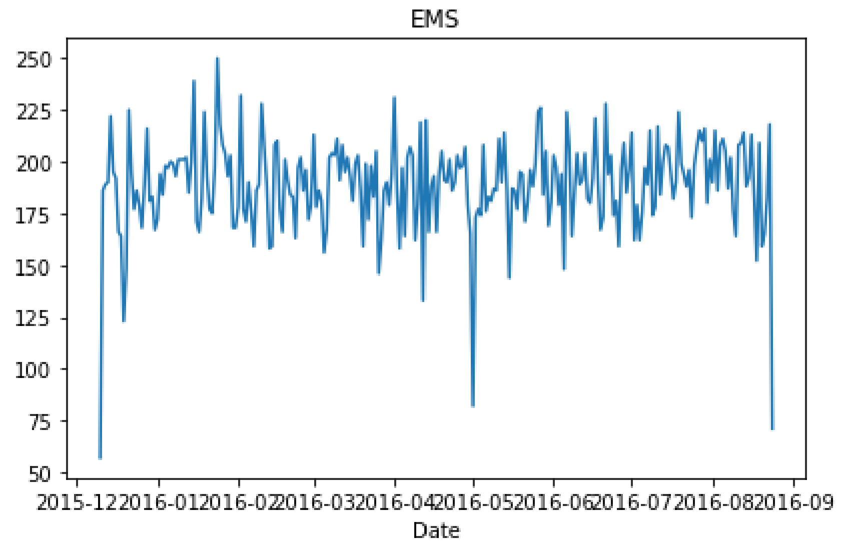
In [38]:
```
df[df['Reason']=='Traffic'].groupby('Date').count()['lat'].plot()
plt.title('Traffic')
plt.tight_layout()
```



In [39]:
```
df[df['Reason']=='Fire'].groupby('Date').count()['lat'].plot()
plt.title('Fire')
plt.tight_layout()
```



In [40]:
```
df[df['Reason']=='EMS'].groupby('Date').count()['lat'].plot()
plt.title('EMS')
plt.tight_layout()
```



restructuring the dataframe so that the columns become the Hours and the Index becomes the Day of the Week.

In [41]:
```
dayhour=df.groupby(by=['Day of week','Hour']).count()['Reason'].unstack()
```

```
In [42]: dayhour
```

Out[42]:

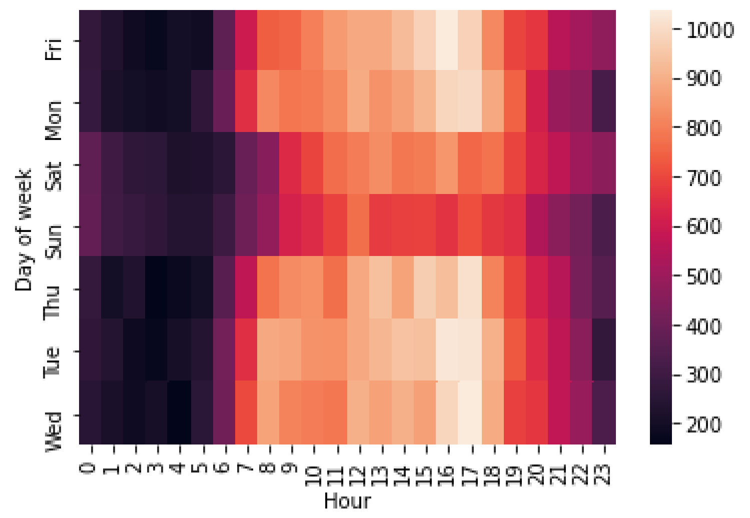
Hour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
Day of week																					
Fri	275	235	191	175	201	194	372	598	742	752	...	932	980	1039	980	820	696	667	559	514	474
Mon	282	221	201	194	204	267	397	653	819	786	...	869	913	989	997	885	746	613	497	472	325
Sat	375	301	263	260	224	231	257	391	459	640	...	789	796	848	757	778	696	628	572	506	467
Sun	383	306	286	268	242	240	300	402	483	620	...	684	691	663	714	670	655	537	461	415	330
Thu	278	202	233	159	182	203	362	570	777	828	...	876	969	935	1013	810	698	617	553	424	354
Tue	269	240	186	170	209	239	415	655	889	880	...	943	938	1026	1019	905	731	647	571	462	274
Wed	250	216	189	209	156	255	410	701	875	808	...	904	867	990	1037	894	686	668	575	490	335

7 rows × 24 columns

HeatMap using this new DataFrame.

```
In [43]: sns.heatmap(dayhour)
```

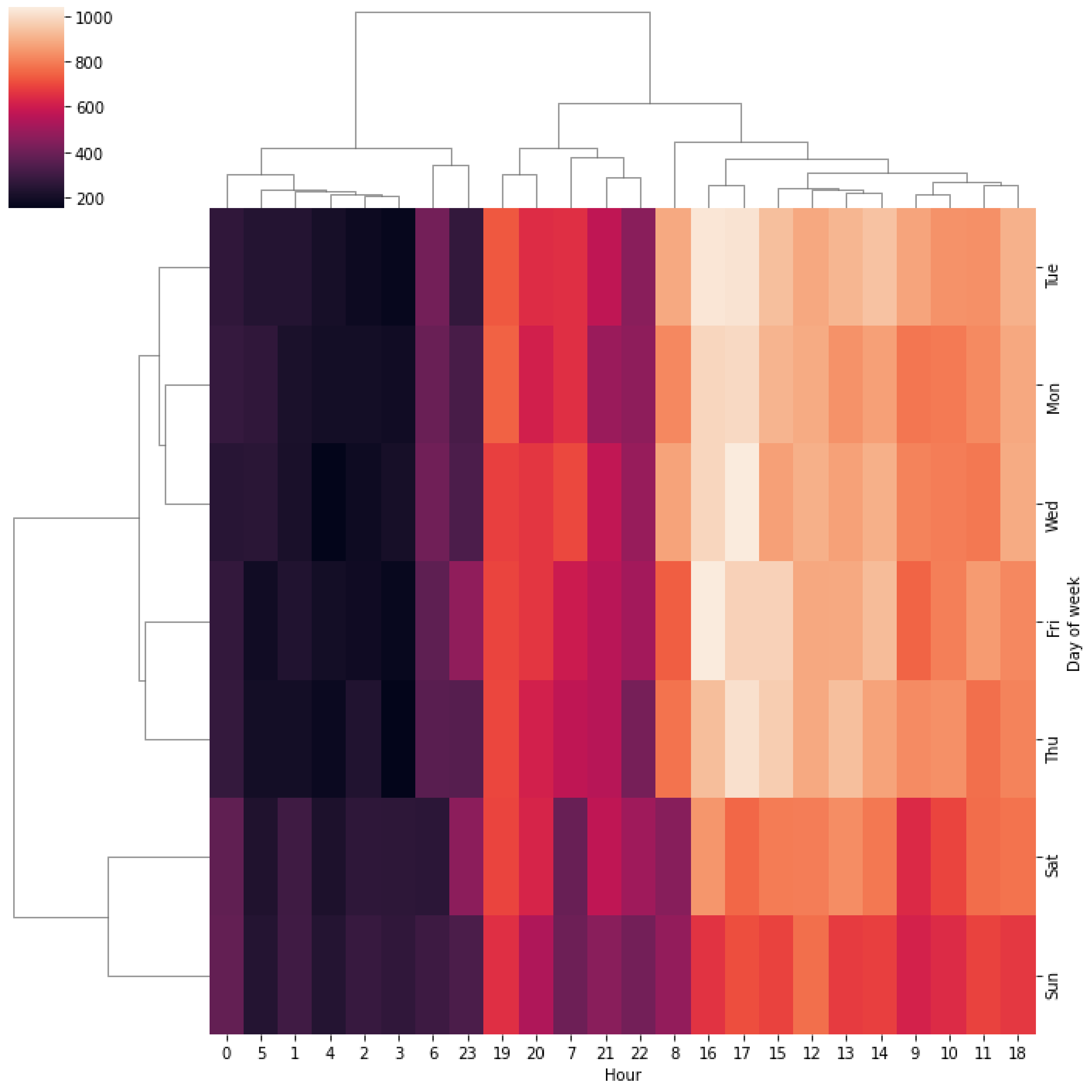
```
Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x2203ff90cf8>
```



clustermap using this DataFrame.

```
In [44]: sns.clustermap(dayhour)
```

```
Out[44]: <seaborn.matrix.ClusterGrid at 0x2203fd7ef60>
```



Month as the column.

```
In [45]: daymonth=df.groupby(by=['Day of week','Month']).count()['Reason'].unstack()
```

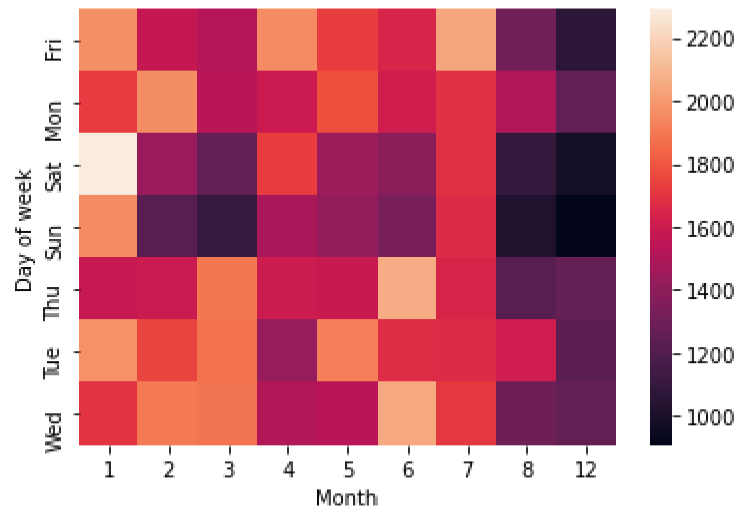
In [46]:
```
daymonth
```

Out[46]:

Month	1	2	3	4	5	6	7	8	12
Day of week									
Fri	1970	1581	1525	1958	1730	1649	2045	1310	1065
Mon	1727	1964	1535	1598	1779	1617	1692	1511	1257
Sat	2291	1441	1266	1734	1444	1388	1695	1099	978
Sun	1960	1229	1102	1488	1424	1333	1672	1021	907
Thu	1584	1596	1900	1601	1590	2065	1646	1230	1266
Tue	1973	1753	1884	1430	1918	1676	1670	1612	1234
Wed	1700	1903	1889	1517	1538	2058	1717	1295	1262

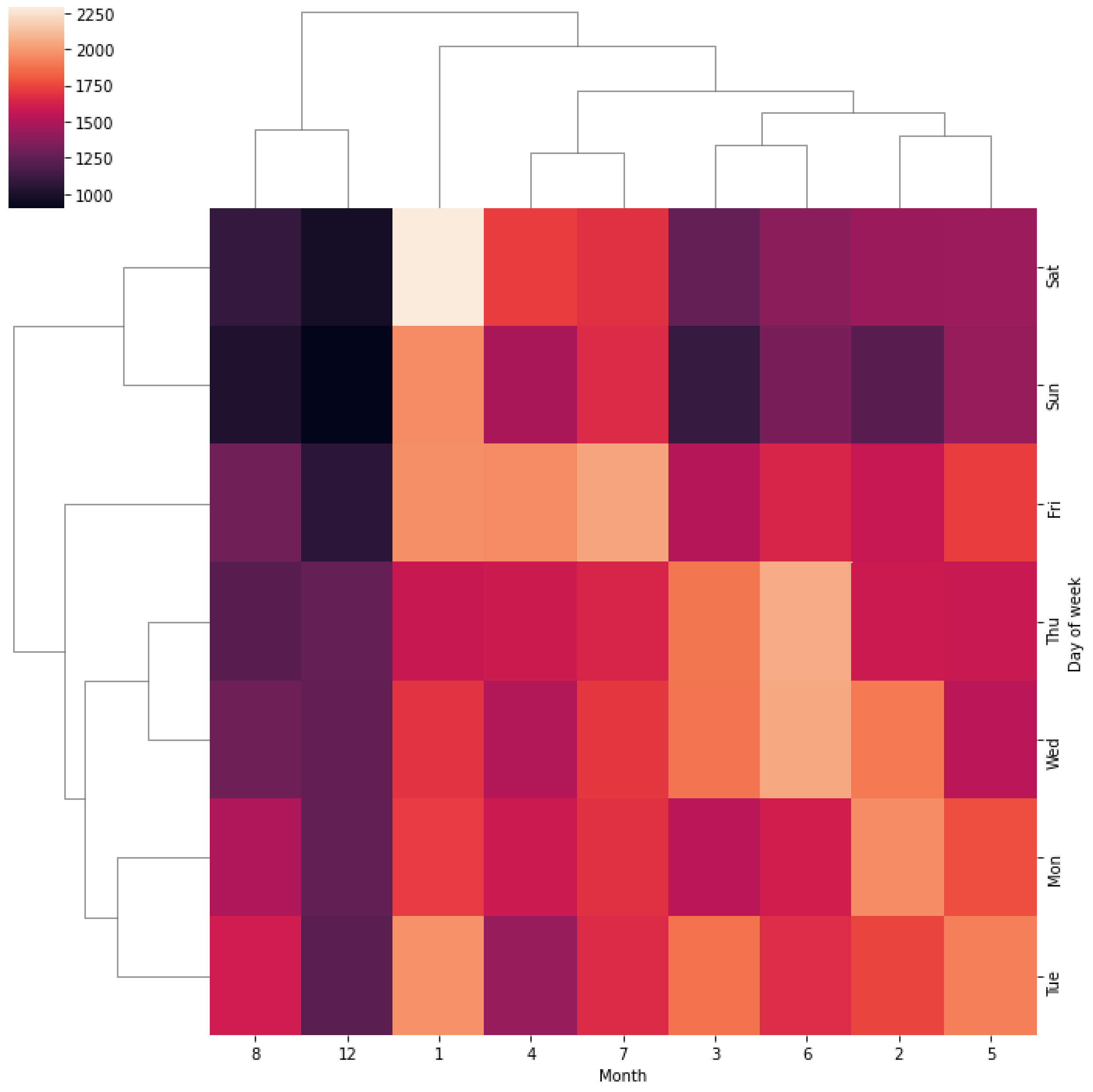
In [47]:
```
sns.heatmap(daymonth)
```

Out[47]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2203fcb28d0>
```



```
In [48]: sns.clustermap(daymonth)
```

```
Out[48]: <seaborn.matrix.ClusterGrid at 0x2203fac5a90>
```



```
In [ ]:
```