911 Calls Capstone Project

Data and Setup

** Import numpy and pandas **

In [24]:

```
import numpy as np
import pandas as pd
```

** Import visualization libraries and set %matplotlib inline. **

In [25]:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
%matplotlib inline
```

** Read in the csv file as a dataframe called df **

In [26]:

```
df = pd.read_csv('911.csv')
```

** Check the info() of the df **

In [27]:

```
df.info()
```

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

** Check the head of df **

In [28]:

```
df.head(3)
```

Out[28]:

	lat	lng	desc	zip	title	timeStamp	twp	
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	REI &
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:40:00	HATFIELD TOWNSHIP	BRI WH
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	

Creating new features

In [32]:

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

** What is the most common Reason for a 911 call. **

In [33]:

```
df['Reason'].value_counts()
```

Out[33]:

```
EMS        48877
Traffic    35695
Fire       14920
Name: Reason, dtype: int64
```

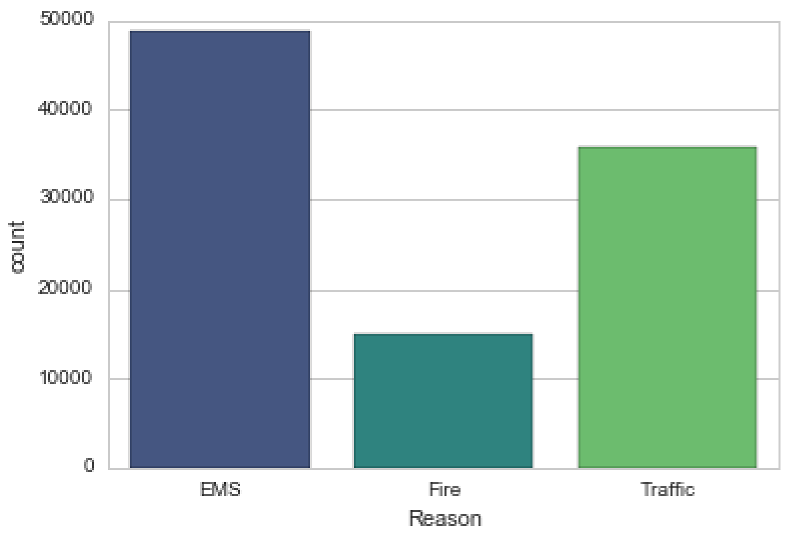
seaborn to create a countplot of 911 calls by Reason.

In [34]:

```
sns.countplot(x='Reason',data=df,palette='viridis')
```

Out[34]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x121757b70>
```



In [37]:

```
df['Hour'] = df['timeStamp'].apply(lambda time: time.hour)
df['Month'] = df['timeStamp'].apply(lambda time: time.month)
df['Day of Week'] = df['timeStamp'].apply(lambda time: time.dayofweek)
```

** Notice how the Day of Week is an integer 0-6. Use the .map() with this dictionary to map the actual string names to the day of the week: **

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

In [38]:

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

In [39]:

```
df['Day of Week'] = df['Day of Week'].map(dmap)
```

seaborn to create a countplot of the Day of Week column with the hue based off of the Reason column.

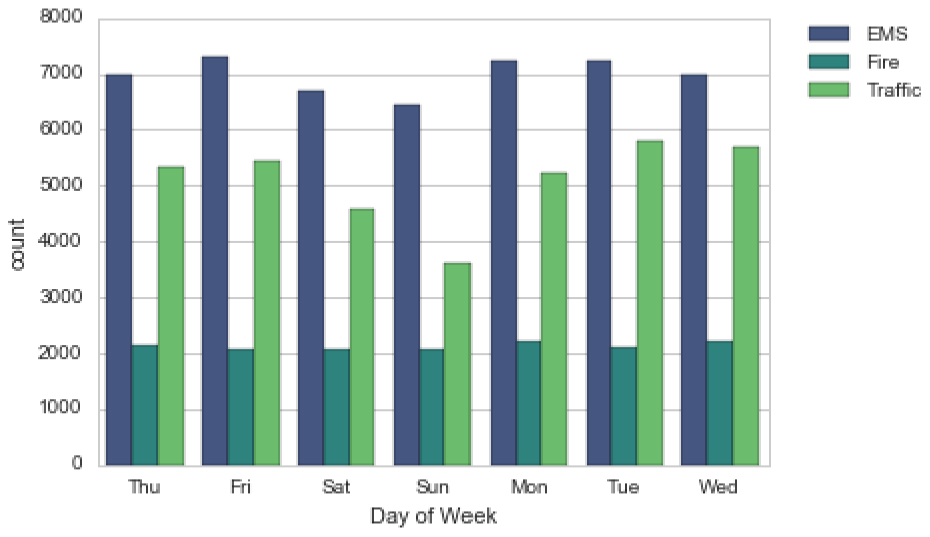
In [40]:

```
sns.countplot(x='Day of Week',data=df,hue='Reason',palette='viridis')

# To relocate the legend
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
```

Out[40]:

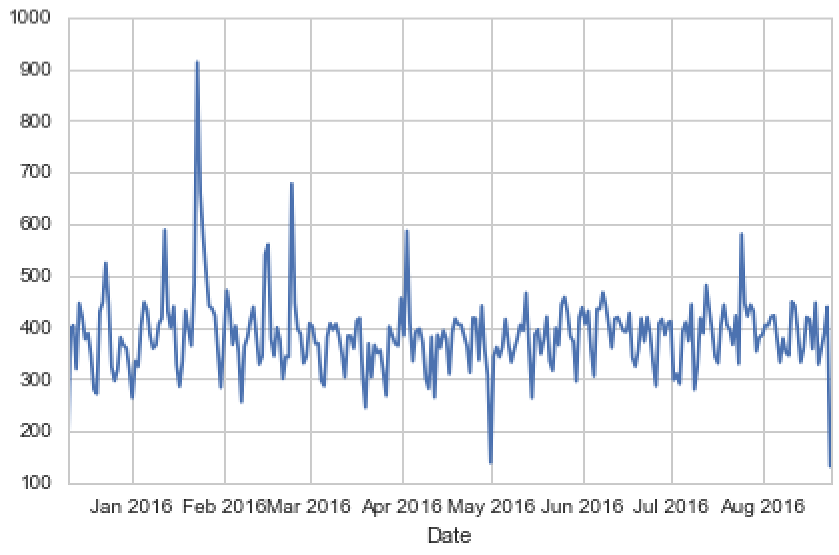
```
<matplotlib.legend.Legend at 0x121762710>
```



groupby this Date column with the count() aggregate and created a plot of counts of 911 calls.

In [47]:

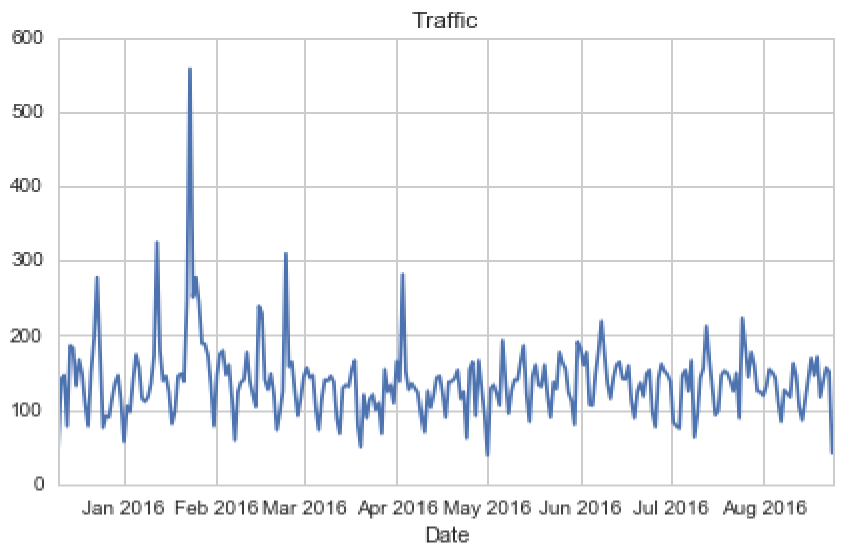
```
df.groupby('Date').count()['twp'].plot()
plt.tight_layout()
```



recreated this plot but create 3 separate plots with each plot representing a Reason for the 911 call

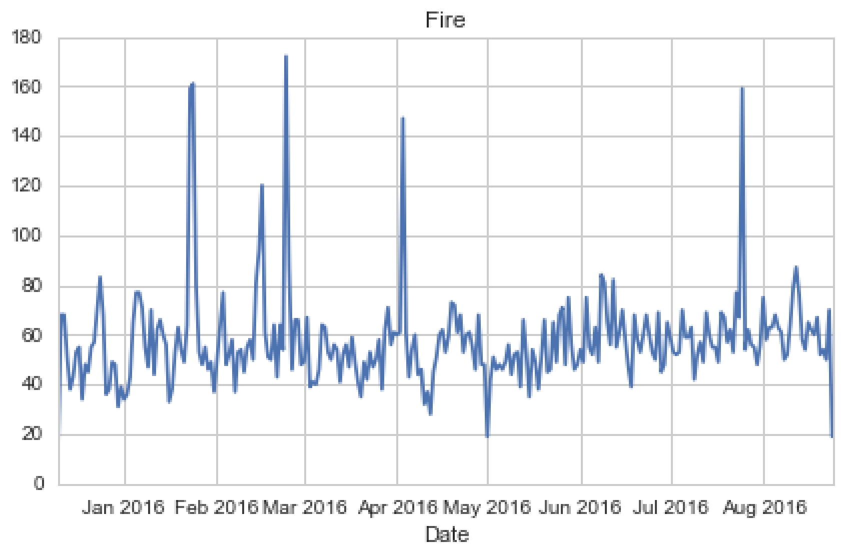
In [48]:

```
df[df['Reason']=='Traffic'].groupby('Date').count()['twp'].plot()
plt.title('Traffic')
plt.tight_layout()
```



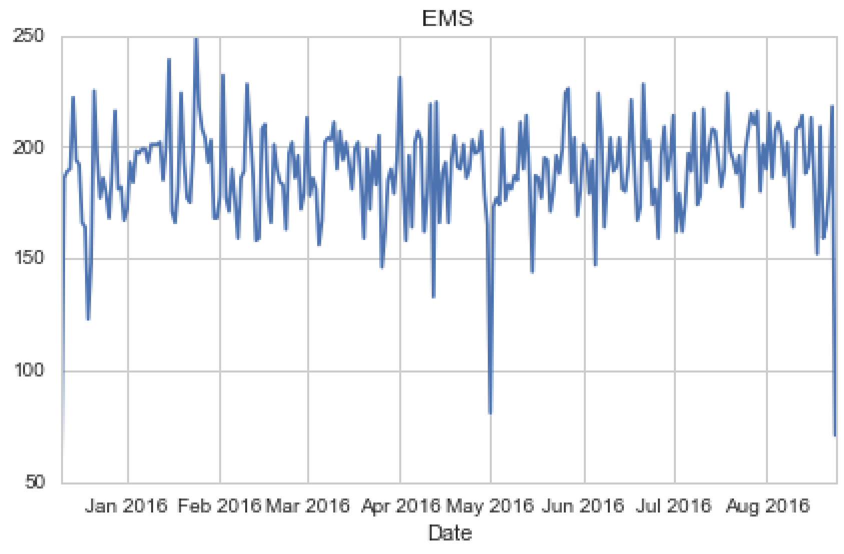
In [49]:

```
df[df['Reason']=='Fire'].groupby('Date').count()['twp'].plot()
plt.title('Fire')
plt.tight_layout()
```



In [50]:

```
df[df['Reason']=='EMS'].groupby('Date').count()['twp'].plot()
plt.title('EMS')
plt.tight_layout()
```



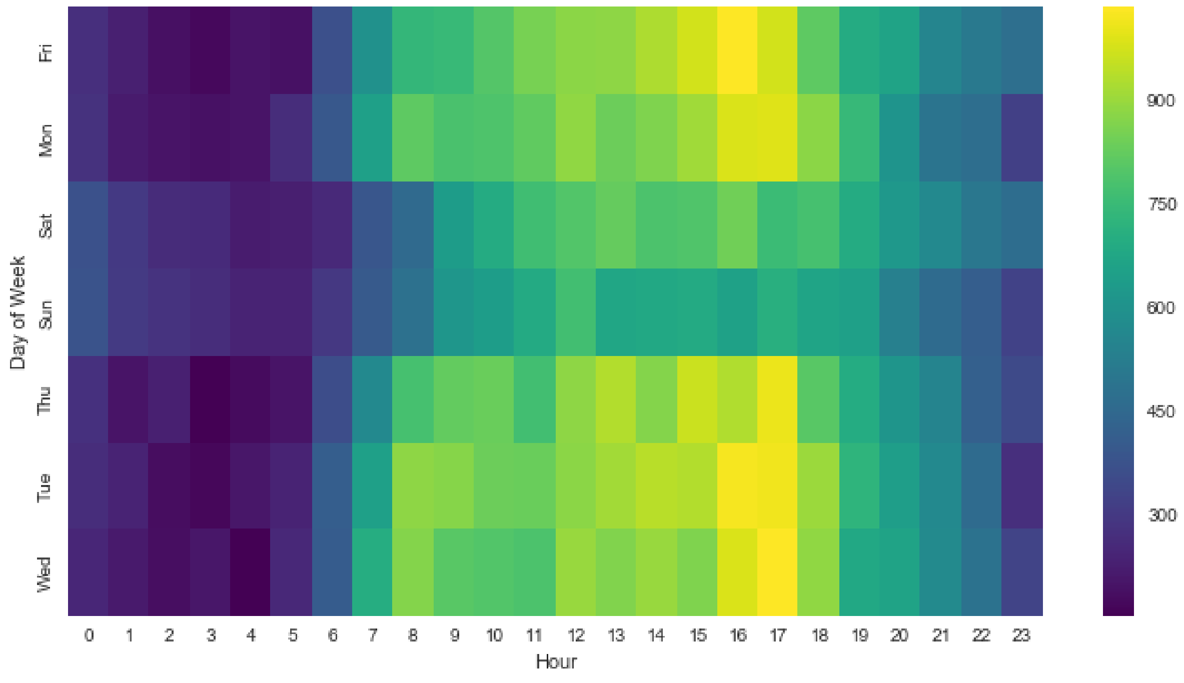
Heatmap

In [52]:

```
plt.figure(figsize=(12,6))
sns.heatmap(dayHour,cmap='viridis')
```

Out[52]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x12305acf8>
```



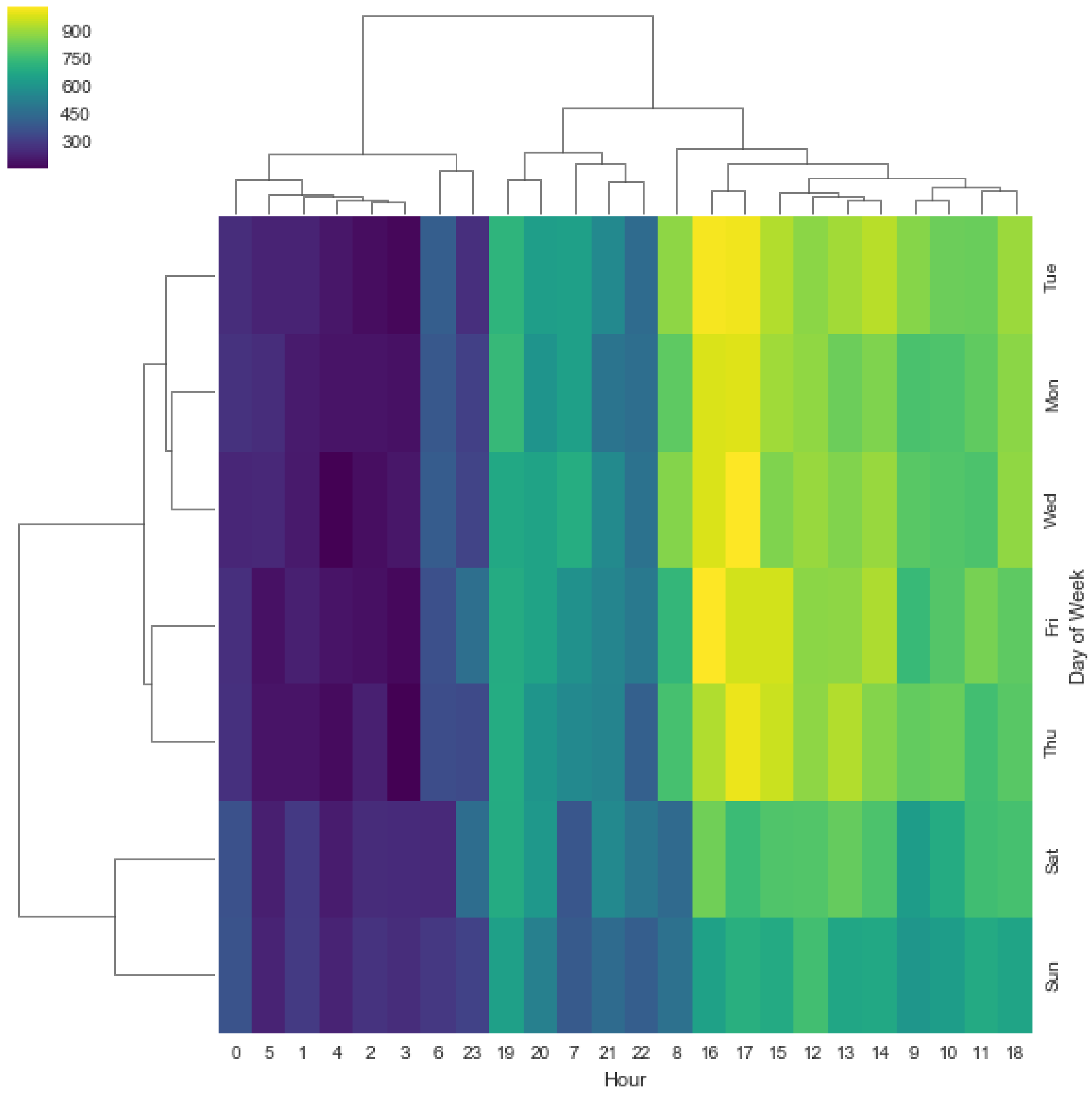
** Now create a clustermap using this DataFrame. **

In [53]:

```
sns.clustermap(dayHour,cmap='viridis')
```

Out[53]:

```
<seaborn.matrix.ClusterGrid at 0x103276748>
```



** Now repeat these same plots and operations, for a DataFrame that shows the Month as the column. **

In [54]:

```
dayMonth = df.groupby(by=['Day of Week','Month']).count()['Reason'].unstack()
dayMonth.head()
```

Out[54]:

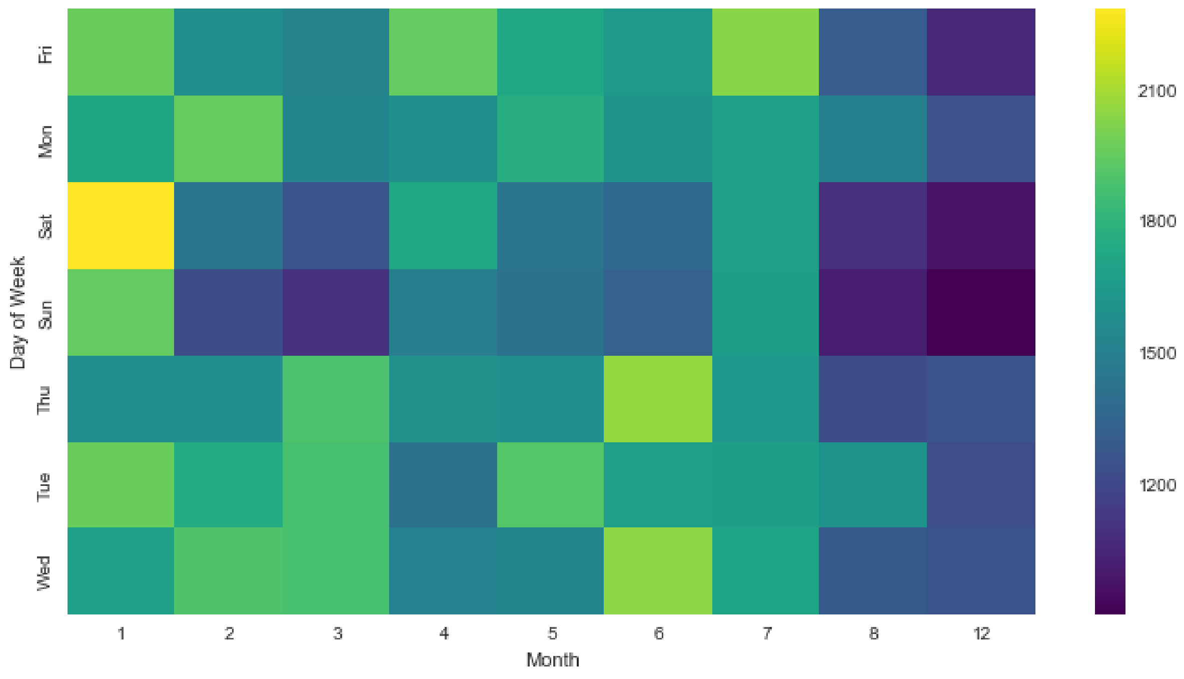
Month	1	2	3	4	5	6	7	8	12
Day of Week									
Fri	1970	1581	1525	1958	1730	1649	2045	1310	1065
Mon	1727	1964	1535	1598	1779	1617	1692	1511	1257
Sat	2291	1441	1266	1734	1444	1388	1695	1099	978
Sun	1960	1229	1102	1488	1424	1333	1672	1021	907
Thu	1584	1596	1900	1601	1590	2065	1646	1230	1266

In [55]:

```
plt.figure(figsize=(12,6))
sns.heatmap(dayMonth,cmap='viridis')
```

Out[55]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11bcabf98>
```

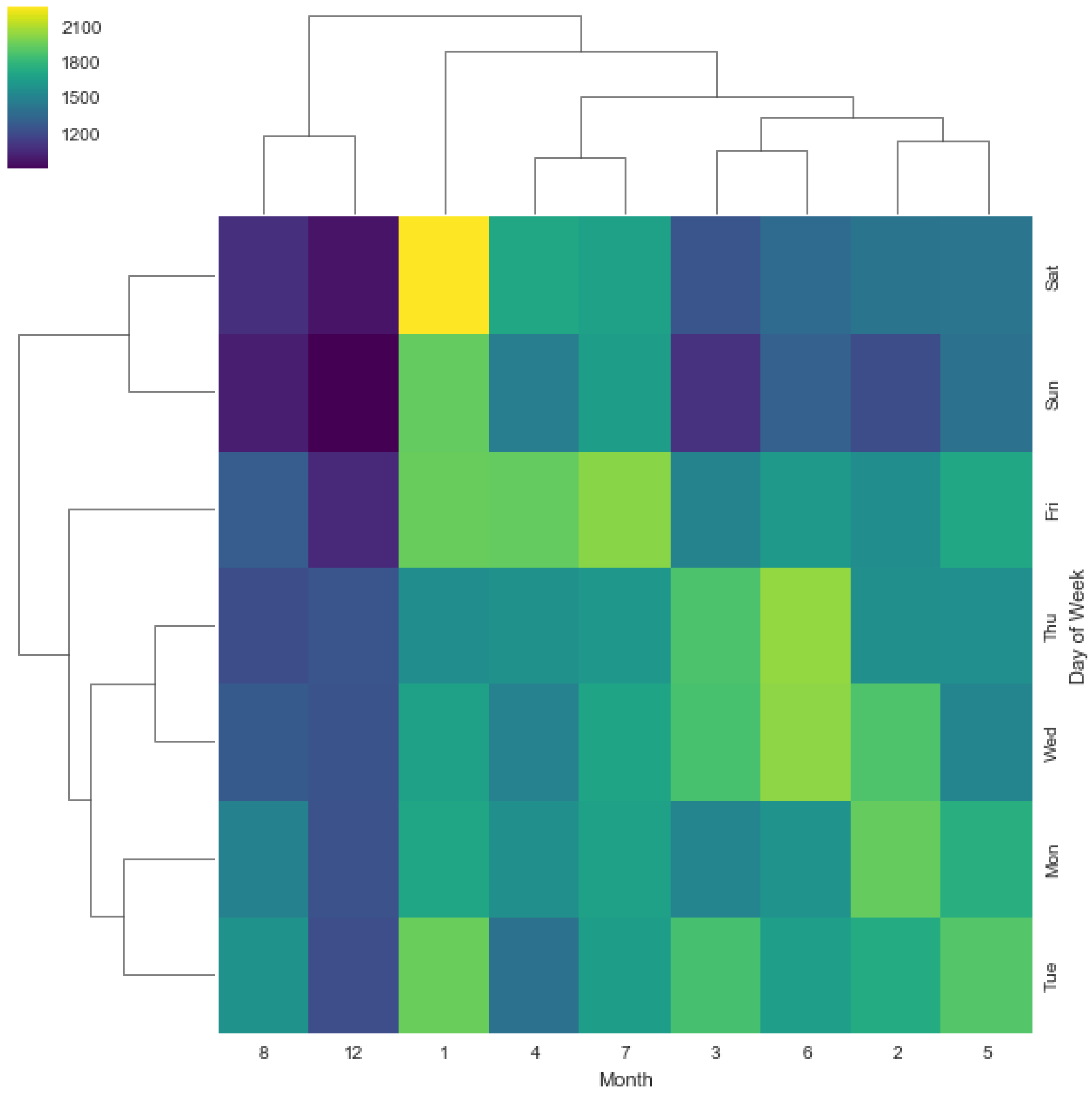


In [56]:

```
sns.clustermap(dayMonth,cmap='viridis')
```

Out[56]:

```
<seaborn.matrix.ClusterGrid at 0x120341e80>
```


Sat
Sun
Fri
Thu
Wed
Mon
Tue
Day of Week
Month