In [2]:
```
!pip install seaborn
```

```
Requirement already satisfied: seaborn in f:\python\lib\site-packages (0.11.2)
Requirement already satisfied: scipy>=1.0 in f:\python\lib\site-packages (from
seaborn) (1.7.3)
Requirement already satisfied: pandas>=0.23 in f:\python\lib\site-packages (fro
m seaborn) (1.4.2)
Requirement already satisfied: numpy>=1.15 in f:\python\lib\site-packages (from
seaborn) (1.21.5)
Requirement already satisfied: matplotlib>=2.2 in f:\python\lib\site-packages
(from seaborn) (3.5.1)
Requirement already satisfied: pyparsing>=2.2.1 in f:\python\lib\site-packages
(from matplotlib>=2.2->seaborn) (3.0.4)
Requirement already satisfied: python-dateutil>=2.7 in f:\python\lib\site-packa
ges (from matplotlib>=2.2->seaborn) (2.8.2)
Requirement already satisfied: packaging>=20.0 in f:\python\lib\site-packages
(from matplotlib>=2.2->seaborn) (21.3)
Requirement already satisfied: kiwisolver>=1.0.1 in f:\python\lib\site-packages
(from matplotlib>=2.2->seaborn) (1.3.2)
Requirement already satisfied: cycler>=0.10 in f:\python\lib\site-packages (fro
m matplotlib>=2.2->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in f:\python\lib\site-packages
(from matplotlib>=2.2->seaborn) (4.25.0)
Requirement already satisfied: pillow>=6.2.0 in f:\python\lib\site-packages (fr
om matplotlib>=2.2->seaborn) (9.0.1)
Requirement already satisfied: pytz>=2020.1 in f:\python\lib\site-packages (fro
m pandas>=0.23->seaborn) (2021.3)
Requirement already satisfied: six>=1.5 in f:\python\lib\site-packages (from py
thon-dateutil>=2.7->matplotlib>=2.2->seaborn) (1.16.0)
```

In [9]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams['figure.figsize'] = 20, 10
import pandas as pd
import numpy as np
```

In [32]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams['figure.figsize'] = 20, 10
import pandas as pd
import numpy as np
bikes = pd.read_csv('C:/Users/VivianHuo/Desktop/data/bikeshare.csv')
bikes.head()
bikes['start'] = pd.to_datetime(bikes['Start date'], infer_datetime_format=True)
bikes['end'] = pd.to_datetime(bikes['End date'], infer_datetime_format=True)
bikes["dur"] = (bikes['Duration (ms)']/1000).astype(int)
bikes.head()
```

Out[32]:

	Duration (ms)	Start date	End date	Start station number	Start station	End station number	End station	Bike number	Member Type	s
0	301295	3/31/2016 23:59	4/1/2016 0:04	31280	11th & S St NW	31506	1st & Rhode Island Ave NW	W00022	Registered	20 03 23:59
1	557887	3/31/2016 23:59	4/1/2016 0:08	31275	New Hampshire Ave & 24th St NW	31114	18th St & Wyoming Ave NW	W01294	Registered	20 03 23:59
2	555944	3/31/2016 23:59	4/1/2016 0:08	31101	14th & V St NW	31221	18th & M St NW	W01416	Registered	20 03 23:59
3	766916	3/31/2016 23:57	4/1/2016 0:09	31226	34th St & Wisconsin Ave NW	31214	17th & Corcoran St NW	W01090	Registered	20 03 23:57
4	139656	3/31/2016 23:57	3/31/2016 23:59	31011	23rd & Crystal Dr	31009	27th & Crystal Dr	W21934	Registered	20 03 23:57

In [30]:
```
bikes.dur.mean()
```

Out[30]:
```
992.8716543657755
```

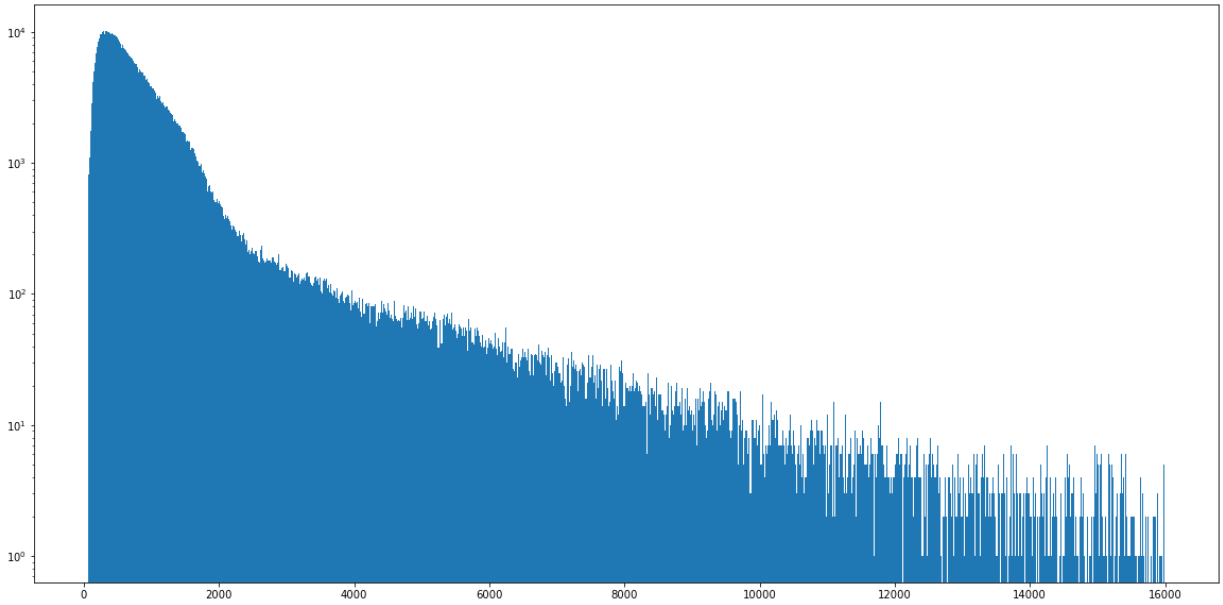
In [33]:
```
bikes.dur.std()
```

Out[33]:
```
2073.9809135296514
```

In [34]:
```
bikes[bikes.dur>16000].shape
```

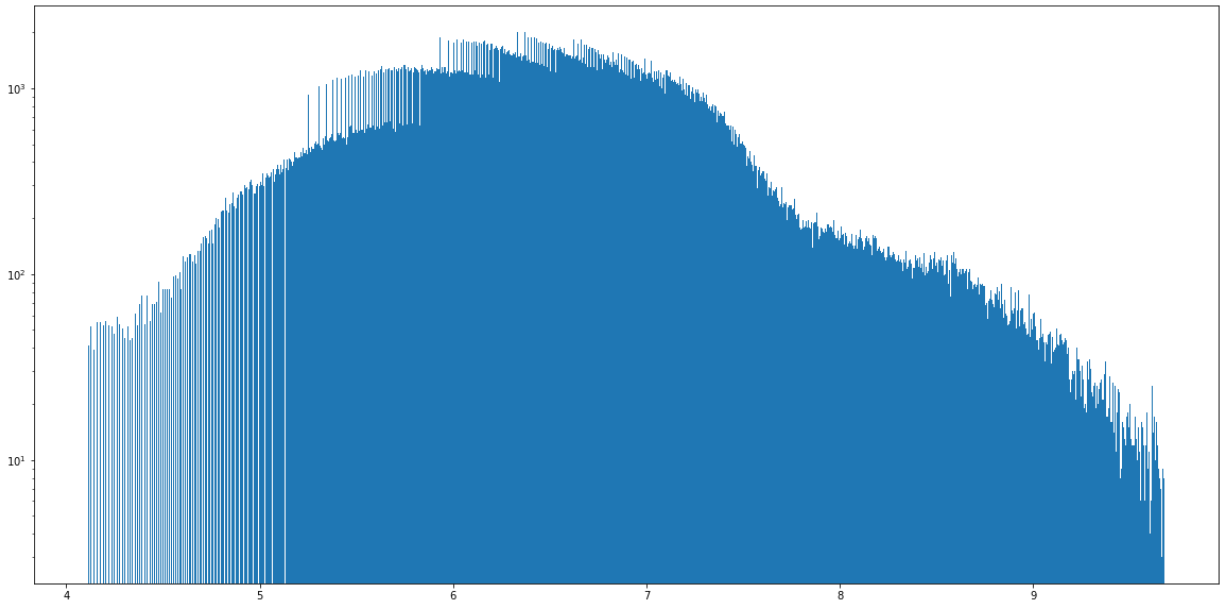
Out[34]:
```
(973, 12)
```

In [37]:
```
plt.rcParams['figure.figsize'] = 20, 10
_=plt.hist(bikes[bikes.dur<16000].dur, log=True, bins=1000)
```



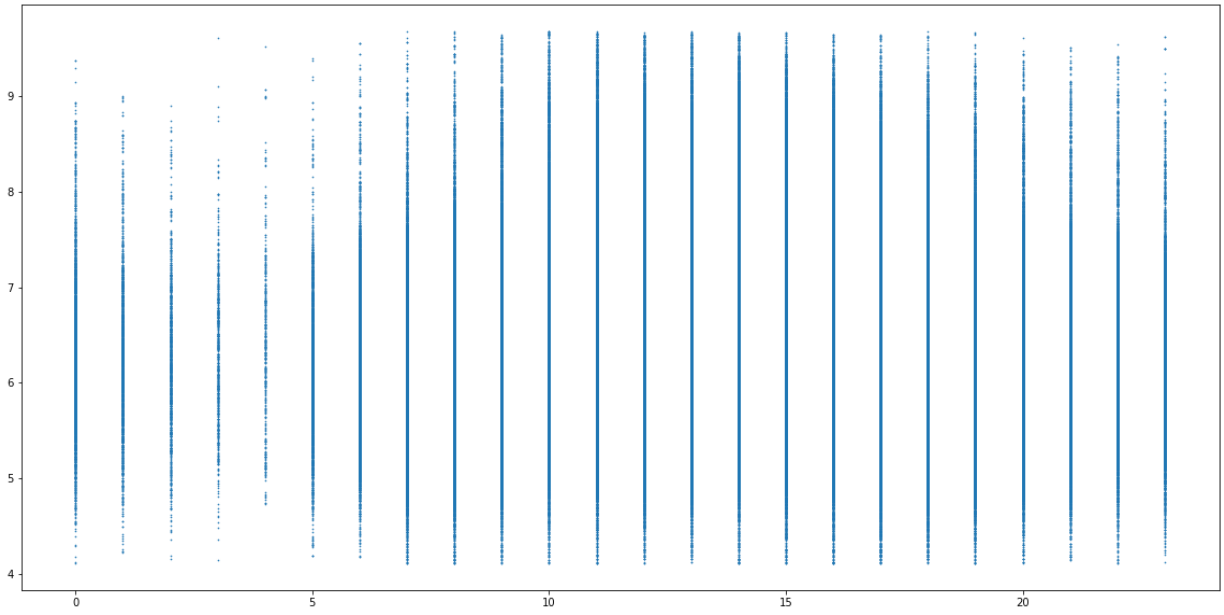
In [38]:
```
short = bikes[bikes.dur<16000]
```

In [39]:
```
_=plt.hist(np.log1p(short.dur), log=True, bins=1000)
```



In [40]:
```
plt.scatter(short.start.dt.hour, np.log1p(short.dur), s=.4)
```

Out[40]:
```
<matplotlib.collections.PathCollection at 0x214099b4fd0>
```



In [41]:
```
np.log1p(0), np.log(0)
```

```
C:\Users\VivianHuo\AppData\Local\Temp\ipykernel_14956\1076539907.py:1: RuntimeW
arning: divide by zero encountered in log
  np.log1p(0), np.log(0)
```

Out[41]:
```
(0.0, -inf)
```

In [42]:
```
bikes['log_dur'] = np.round(np.log1p(bikes.dur), 1)
monday = bikes[bikes.start.dt.dayofweek==1]
dur_hour = monday.groupby(['log_dur', monday.start.dt.hour]).count()
dur_hour
```

Out[42]:

		Duration (ms)	Start date	End date	Start station number	Start station	End station number	End station	Bike number	Member Type	start	e
log_dur	**start**											
	7	1	1	1	1	1	1	1	1	1	1	
	9	2	2	2	2	2	2	2	2	2	2	
4.1	**11**	1	1	1	1	1	1	1	1	1	1	
	14	2	2	2	2	2	2	2	2	2	2	
	16	2	2	2	2	2	2	2	2	2	2	
...	**...**	...	...	...	...	...	...	...	...	...	...	
11.2	**21**	2	2	2	2	2	2	2	2	2	2	
	14	1	1	1	1	1	1	1	1	1	1	
11.3	**17**	1	1	1	1	1	1	1	1	1	1	
	19	1	1	1	1	1	1	1	1	1	1	
11.4	**18**	1	1	1	1	1	1	1	1	1	1	

1184 rows × 12 columns

In [43]:
```
duration_hour = dur_hour.start.unstack().T.fillna(0)
duration_hour
```

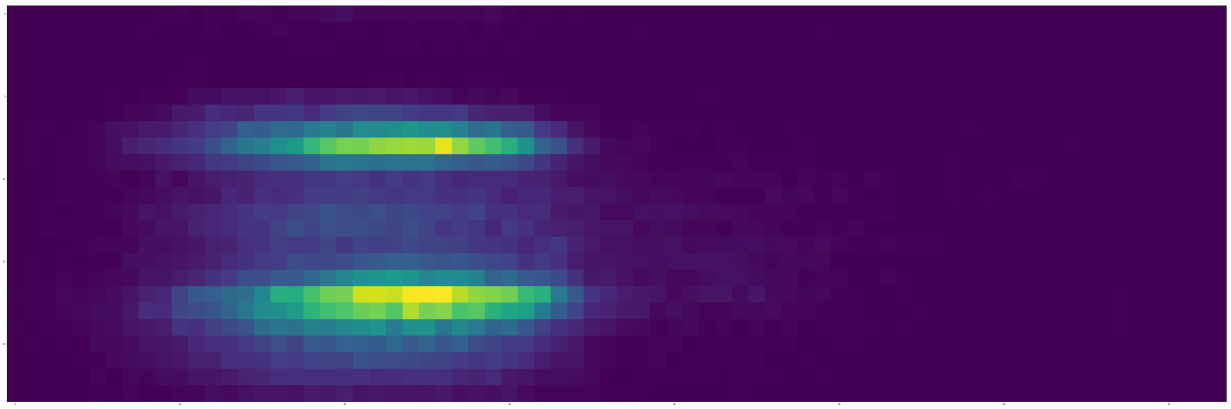
Out[43]:

log_dur	4.1	4.2	4.3	4.4	4.5	4.6	4.7	4.8	4.9	5.0	...	10.5	10.6	10.7	10.8	10.9	11.
start																	
0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	2.0	3.0	...	0.0	0.0	0.0	0.0	0.0	0.
1	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	3.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	1.0	0.0	0.
4	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.
5	0.0	0.0	1.0	0.0	0.0	1.0	4.0	1.0	7.0	6.0	...	0.0	0.0	0.0	0.0	0.0	0.
6	0.0	0.0	0.0	2.0	1.0	2.0	4.0	9.0	11.0	21.0	...	0.0	0.0	0.0	1.0	0.0	0.
7	1.0	5.0	4.0	1.0	5.0	12.0	25.0	31.0	46.0	46.0	...	0.0	1.0	1.0	0.0	0.0	0.
8	0.0	3.0	2.0	6.0	7.0	11.0	22.0	52.0	68.0	79.0	...	4.0	2.0	1.0	0.0	0.0	0.
9	2.0	3.0	2.0	4.0	3.0	11.0	18.0	22.0	28.0	42.0	...	1.0	1.0	0.0	0.0	0.0	0.
10	0.0	0.0	1.0	3.0	5.0	7.0	8.0	5.0	10.0	31.0	...	0.0	0.0	0.0	0.0	0.0	0.
11	1.0	0.0	2.0	5.0	4.0	7.0	7.0	10.0	13.0	22.0	...	1.0	0.0	0.0	0.0	0.0	0.
12	0.0	0.0	4.0	2.0	7.0	6.0	12.0	16.0	36.0	30.0	...	0.0	1.0	0.0	0.0	0.0	0.
13	0.0	2.0	6.0	3.0	5.0	6.0	4.0	15.0	20.0	36.0	...	0.0	0.0	0.0	0.0	0.0	0.
14	2.0	0.0	1.0	1.0	3.0	8.0	9.0	11.0	26.0	24.0	...	0.0	0.0	0.0	0.0	0.0	0.
15	0.0	3.0	0.0	5.0	1.0	7.0	6.0	22.0	26.0	31.0	...	0.0	0.0	0.0	0.0	0.0	0.
16	2.0	6.0	1.0	11.0	6.0	10.0	14.0	17.0	36.0	35.0	...	0.0	0.0	0.0	0.0	2.0	0.
17	3.0	7.0	7.0	13.0	12.0	14.0	20.0	36.0	57.0	71.0	...	0.0	0.0	0.0	3.0	1.0	1.
18	0.0	4.0	7.0	9.0	13.0	20.0	21.0	40.0	79.0	75.0	...	0.0	0.0	2.0	4.0	1.0	0.
19	3.0	0.0	7.0	7.0	9.0	16.0	19.0	34.0	43.0	52.0	...	0.0	1.0	2.0	3.0	0.0	1.
20	0.0	7.0	2.0	4.0	2.0	13.0	14.0	19.0	34.0	38.0	...	0.0	1.0	1.0	1.0	1.0	1.
21	1.0	2.0	1.0	2.0	3.0	6.0	16.0	19.0	26.0	35.0	...	1.0	2.0	0.0	1.0	0.0	0.
22	1.0	0.0	2.0	2.0	1.0	8.0	1.0	13.0	10.0	20.0	...	1.0	0.0	1.0	0.0	0.0	0.
23	0.0	0.0	1.0	0.0	2.0	5.0	4.0	8.0	3.0	5.0	...	0.0	0.0	1.0	1.0	0.0	0.

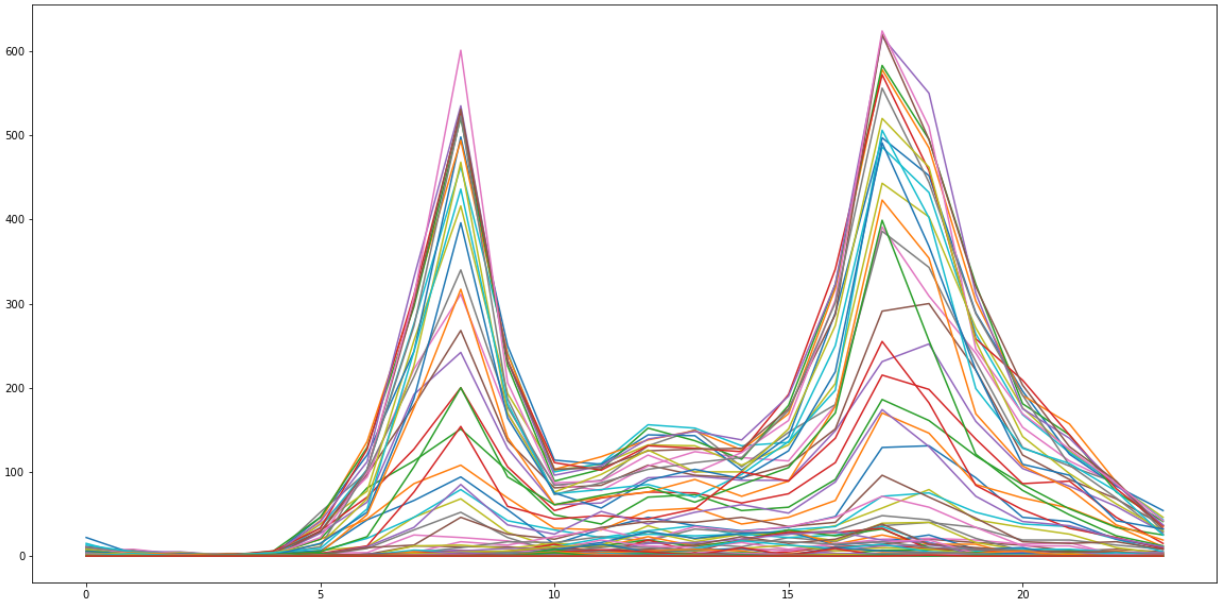
24 rows × 74 columns

In [44]:
```
plt.figure(figsize=(100,100))
plt.imshow(duration_hour)
```

Out[44]:
```
<matplotlib.image.AxesImage at 0x21409a65520>
```



In [45]:
```
_=plt.plot(duration_hour)
```



In [46]:
```
bikes['Member Type'].value_counts()
```

Out[46]:
```
Registered    467432
Casual         84967
Name: Member Type, dtype: int64
```

In [47]:
```
np.round(.65, 1)
```

Out[47]:
```
0.6
```

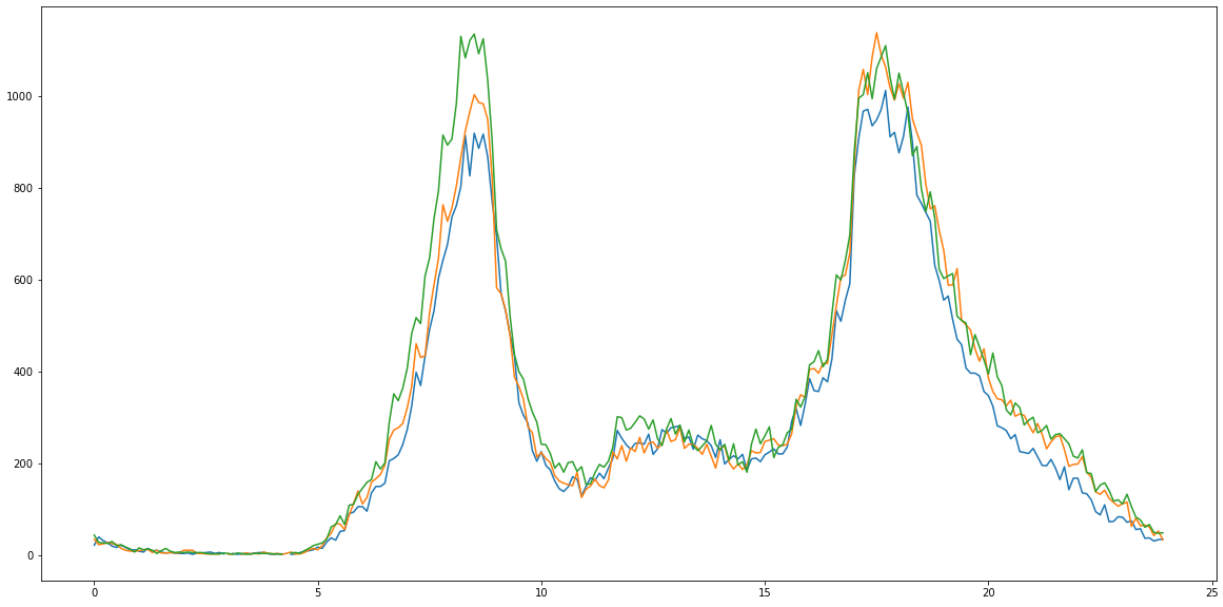
In [48]:
```
37//6, (37//6)/10, 37/60
```

Out[48]:
```
(6, 0.6, 0.6166666666666667)
```

In [51]:
```
bikes['hour_of_day'] = (bikes.start.dt.hour + (bikes.start.dt.minute//6)/10)
bikes['roundhour_of_day'] = (bikes.start.dt.hour ) # keep the hour handy as well
```

In [52]:
```
reg_bikes = bikes[bikes['Member Type']=='Registered']
hours = reg_bikes.groupby([reg_bikes.hour_of_day, reg_bikes.start.dt.dayofweek]).
hours['hour'] = hours.index
day_hour_count = hours.dur.unstack()
plt.figure(figsize=(20,10))
plt.plot(day_hour_count.index, day_hour_count[0])
plt.plot(day_hour_count.index, day_hour_count[1])
plt.plot(day_hour_count.index, day_hour_count[2])
# plt.plot(y.index, day_hour_count[3])
# plt.plot(y.index, day_hour_count[4])
# plt.plot(y.index, day_hour_count[5])
# plt.plot(y.index, day_hour_count[6])
```

Out[52]:
```
[<matplotlib.lines.Line2D at 0x2140762e520>]
```



In [53]:
```
day_hour_count
```

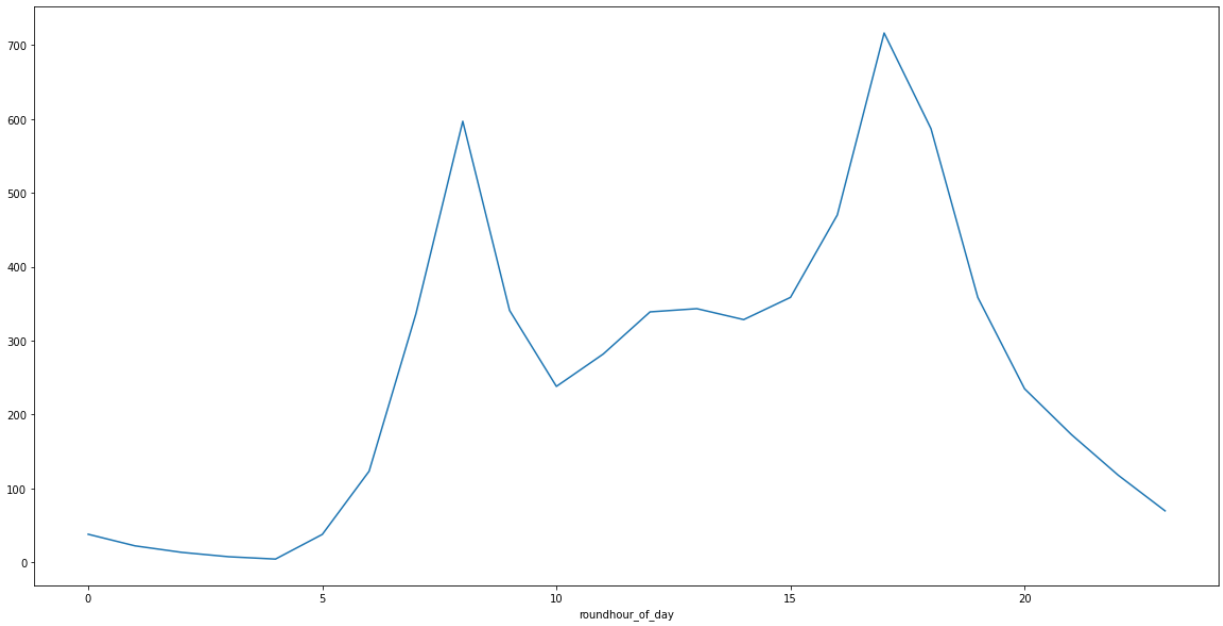
Out[53]:

start	0	1	2	3	4	5	6
hour_of_day							
0.0	21.0	34.0	43.0	47.0	51.0	89.0	106.0
0.1	39.0	22.0	27.0	37.0	56.0	87.0	100.0
0.2	31.0	24.0	26.0	42.0	50.0	98.0	77.0
0.3	26.0	27.0	25.0	29.0	52.0	99.0	87.0
0.4	19.0	24.0	29.0	29.0	50.0	98.0	69.0
...	...	...	...	...	...	...	...
23.5	36.0	65.0	60.0	94.0	80.0	93.0	28.0
23.6	37.0	61.0	66.0	100.0	81.0	95.0	28.0
23.7	30.0	42.0	49.0	80.0	101.0	105.0	27.0
23.8	33.0	52.0	47.0	79.0	91.0	93.0	24.0
23.9	34.0	33.0	48.0	65.0	105.0	111.0	23.0

240 rows × 7 columns

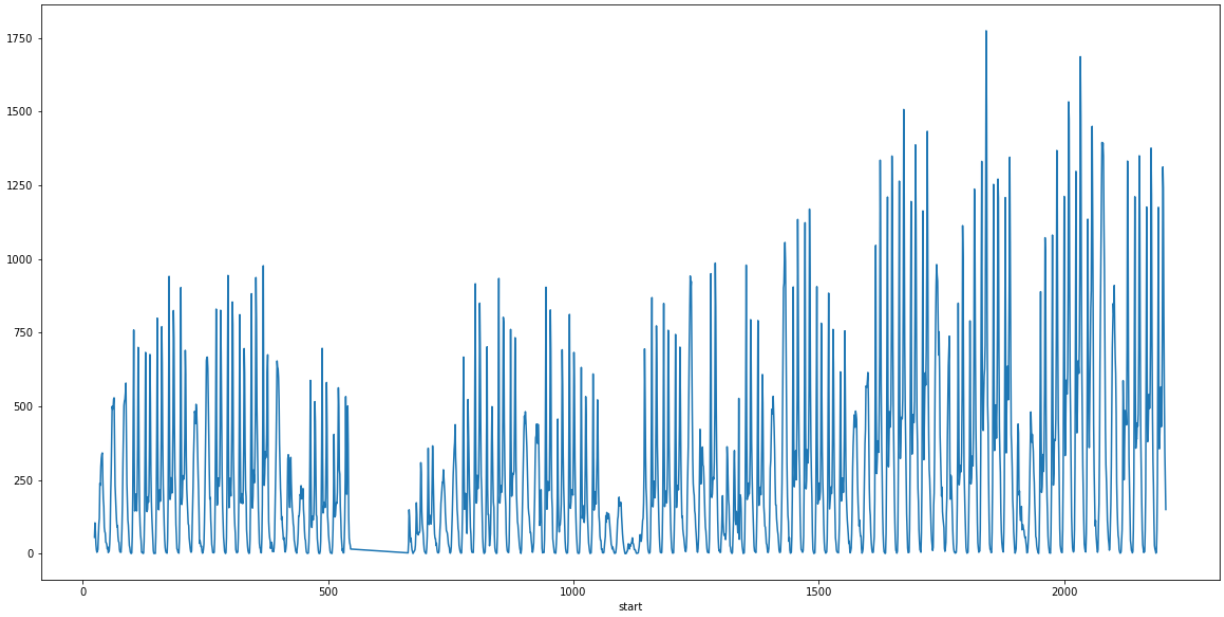
In [54]:
```
hoursn = bikes.groupby('roundhour_of_day').agg('count')
hoursn['hour'] = hoursn.index
(hoursn.start/90).plot() # 90 days in a quarter
```

Out[54]:
```
<AxesSubplot:xlabel='roundhour_of_day'>
```



In [55]:
```
hour_count = bikes.groupby(bikes.start.dt.dayofyear*24 + bikes.start.dt.hour).cou
plt.figure(figsize=(20,10))
hour_count.start.plot()
```

Out[55]: <AxesSubplot:xlabel='start'>



In [56]:
```
day_count = bikes.groupby(bikes.start.dt.dayofyear).count()
```

In [57]:
```
day_hour = bikes.groupby([bikes.start.dt.dayofyear, bikes.start.dt.hour]).count()
```

In [58]:
```
day_hour.start.unstack()
```

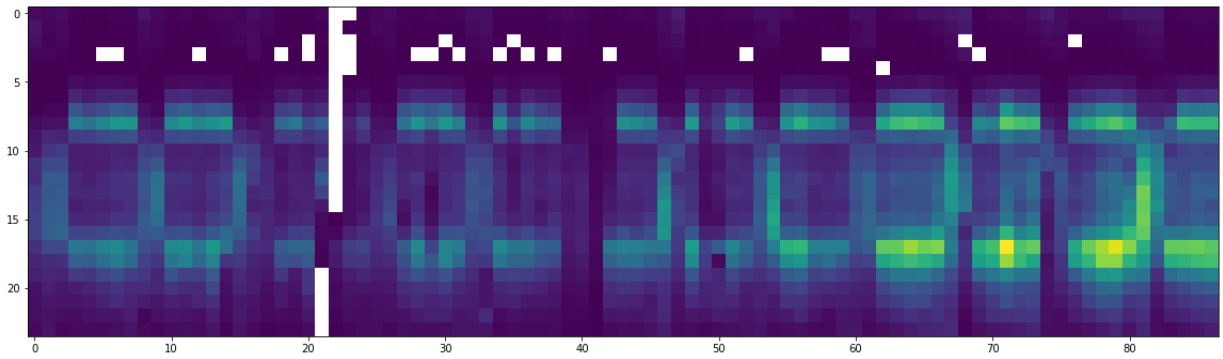
Out[58]:

start	0	1	2	3	4	5	6	7	8	9	...	14	15	16	
start															
1	56.0	105.0	74.0	32.0	13.0	5.0	10.0	14.0	54.0	101.0	...	324.0	338.0	342.0	24
2	37.0	31.0	17.0	23.0	4.0	7.0	10.0	34.0	80.0	203.0	...	495.0	525.0	529.0	39
3	59.0	42.0	39.0	15.0	6.0	9.0	5.0	33.0	87.0	168.0	...	524.0	546.0	579.0	39
4	20.0	6.0	2.0	1.0	3.0	58.0	192.0	468.0	759.0	321.0	...	145.0	206.0	365.0	70
5	5.0	5.0	3.0	1.0	2.0	42.0	131.0	363.0	683.0	329.0	...	175.0	208.0	365.0	67
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
87	113.0	82.0	50.0	34.0	12.0	24.0	94.0	166.0	297.0	509.0	...	910.0	761.0	667.0	6
88	15.0	7.0	2.0	3.0	8.0	42.0	81.0	197.0	587.0	464.0	...	481.0	437.0	696.0	133
89	31.0	11.0	9.0	3.0	8.0	79.0	240.0	727.0	1211.0	564.0	...	433.0	473.0	700.0	135
90	31.0	18.0	4.0	6.0	7.0	79.0	215.0	703.0	1176.0	593.0	...	493.0	545.0	749.0	137
91	28.0	16.0	10.0	2.0	8.0	80.0	240.0	750.0	1175.0	589.0	...	431.0	504.0	746.0	13

87 rows × 24 columns

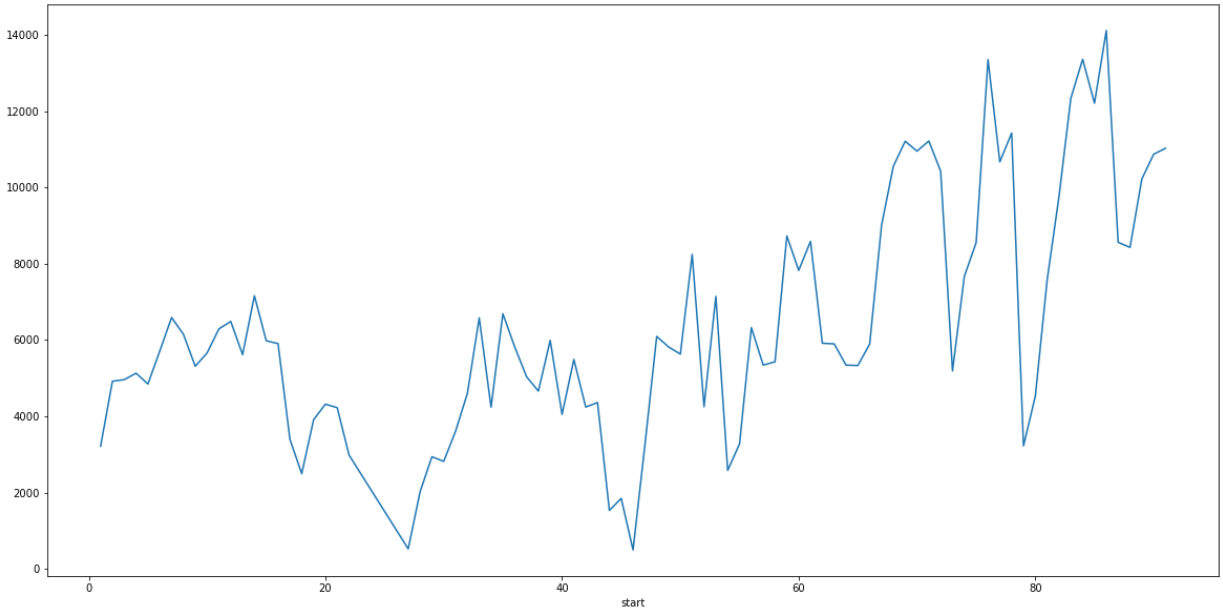
In [59]:
```
plt.figure(figsize=(20,10))
plt.imshow(day_hour.start.unstack().T)
```

Out[59]:
```
<matplotlib.image.AxesImage at 0x2140a271220>
```



In [60]:
```
day_count.start.plot()
```

Out[60]: <AxesSubplot:xlabel='start'>



In [61]:
```
bikes.start.dt.dayofyear
```

Out[61]:
```
0         91
1         91
2         91
3         91
4         91
          ..
552394     1
552395     1
552396     1
552397     1
552398     1
Name: start, Length: 552399, dtype: int64
```

In [62]:
```
bikes[bikes.start=="2016-01-10"].shape
```

Out[62]: (1, 15)

In [63]:
```
#HW1
monday = day_hour_count[[0]].copy()
monday["hour"] = monday.index
monday
```

Out[63]:

hour_of_day	start	0	hour
0.0		21.0	0.0
0.1		39.0	0.1
0.2		31.0	0.2
0.3		26.0	0.3
0.4		19.0	0.4
...		...	...
23.5		36.0	23.5
23.6		37.0	23.6
23.7		30.0	23.7
23.8		33.0	23.8
23.9		34.0	23.9

240 rows × 2 columns

In [64]:
```
saturday = day_hour_count[[5]].copy()
saturday["hour"] = saturday.index
saturday
```

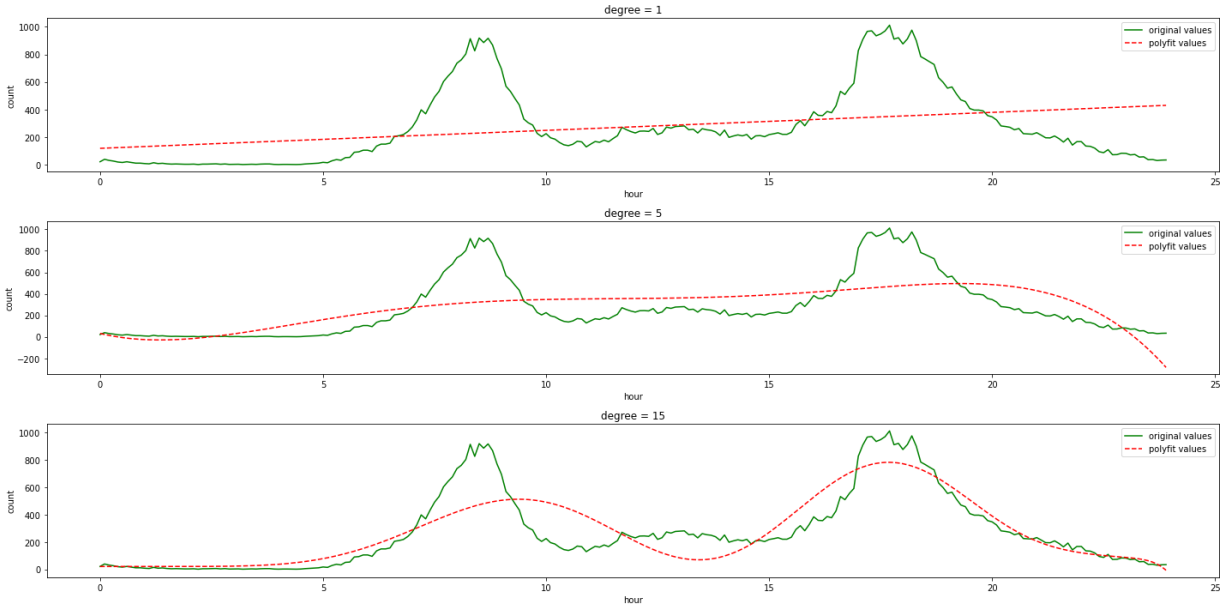
Out[64]:

hour_of_day	start	5	hour
0.0		89.0	0.0
0.1		87.0	0.1
0.2		98.0	0.2
0.3		99.0	0.3
0.4		98.0	0.4
...		...	...
23.5		93.0	23.5
23.6		95.0	23.6
23.7		105.0	23.7
23.8		93.0	23.8
23.9		111.0	23.9

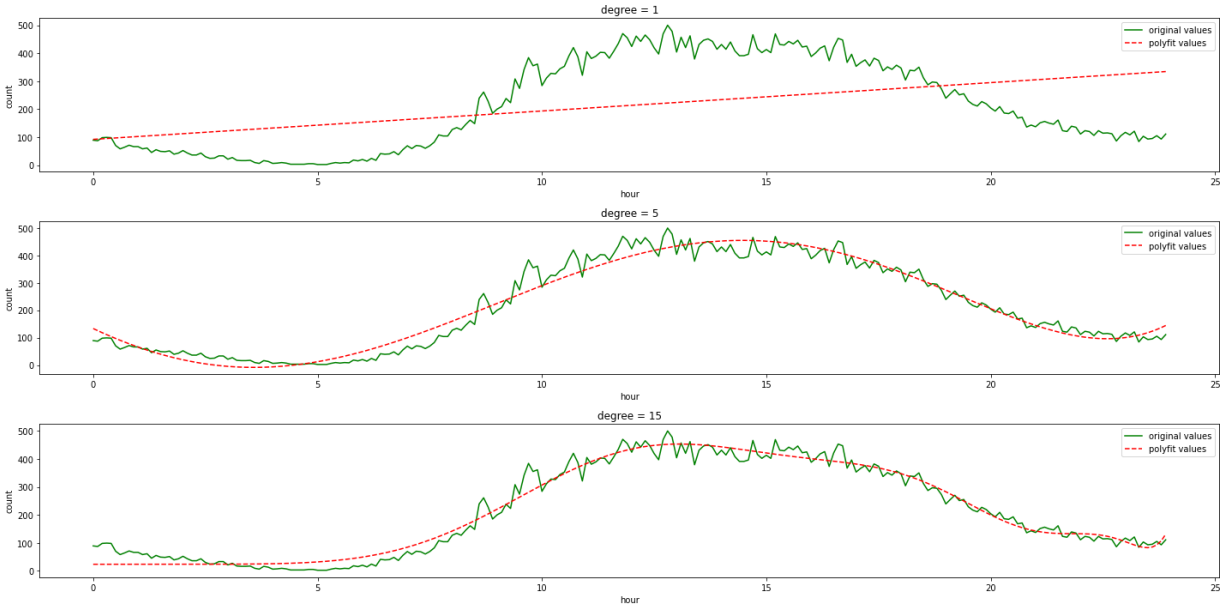
240 rows × 2 columns

In [69]:
```
#2a. Create 3 models fit to monday.hour_of_day with varying polynomial degree
#Degree = 1
# monday['hour_of_day'] = (monday.start.dt.hour + (monday.start.dt.minute/60).rou
#
# hours = monday.groupby('hour_of_day').agg('count')
# hours['hour'] = hours.index
#
# hours.start.plot()
# # import seaborn as sns
#
# sns.lmplot(x='hour', y='start', data=hours, aspect=1.5, scatter_kws={'alpha':0.
from sklearn.preprocessing import PolynomialFeatures
from sklearn import linear_model

count = 1
plt.figure()
x = monday.dropna().iloc[:, 1].values
true_y = monday.dropna().iloc[:, 0].values
for n in [1, 5, 15]:
    poly = PolynomialFeatures(degree=n)
    x_n = poly.fit_transform(x.reshape(-1, 1))
    linear = linear_model.LinearRegression()
    linear.fit(x_n, true_y)
    plt.subplot(3, 1, count)
    plt.plot(x, true_y, 'g-', label='original values')
    plt.plot(x, np.dot(x_n, linear.coef_) + linear.intercept_, 'r--', label='poly
    plt.xlabel("hour")
    plt.ylabel("count")
    plt.title(f'degree = {n}')
    plt.legend()
    count += 1
plt.tight_layout()
plt.show()
```

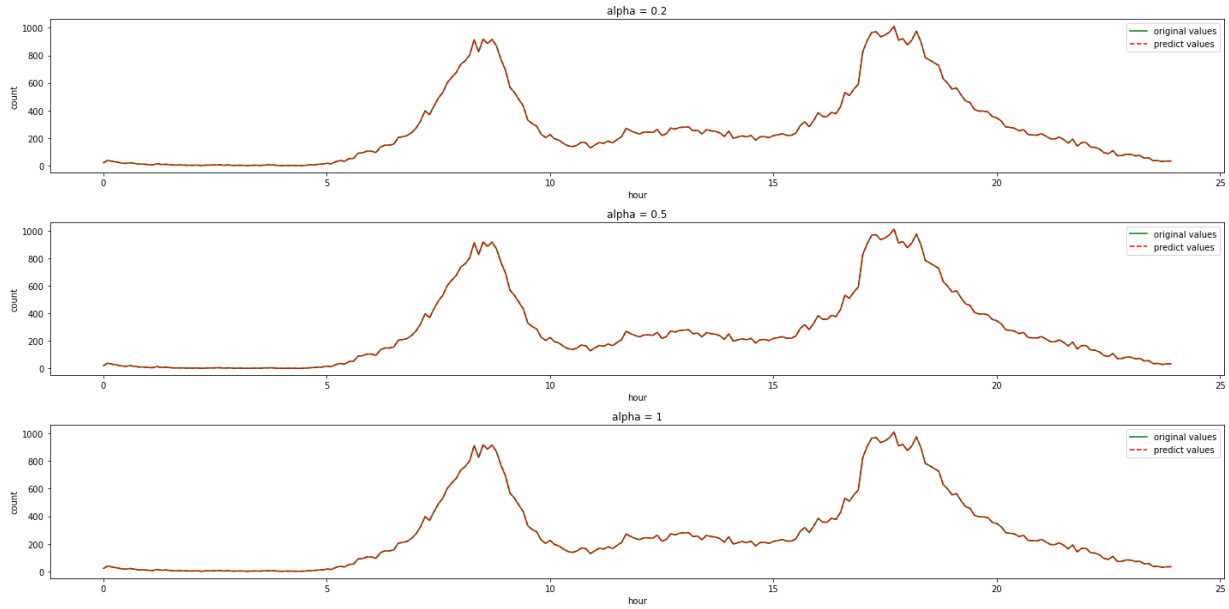


In [70]:
```
count = 1
plt.figure()
x = saturday.dropna().iloc[:, 1].values
true_y = saturday.dropna().iloc[:, 0].values
for n in [1, 5, 15]:
    poly = PolynomialFeatures(degree=n)
    x_n = poly.fit_transform(x.reshape(-1, 1))
    linear = linear_model.LinearRegression()
    linear.fit(x_n, true_y)
    plt.subplot(3, 1, count)
    plt.plot(x, true_y, 'g-', label='original values')
    plt.plot(x, np.dot(x_n, linear.coef_) + linear.intercept_, 'r--', label='poly
    plt.xlabel("hour")
    plt.ylabel("count")
    plt.title(f'degree = {n}')
    plt.legend()
    count += 1
plt.tight_layout()
plt.show()
```

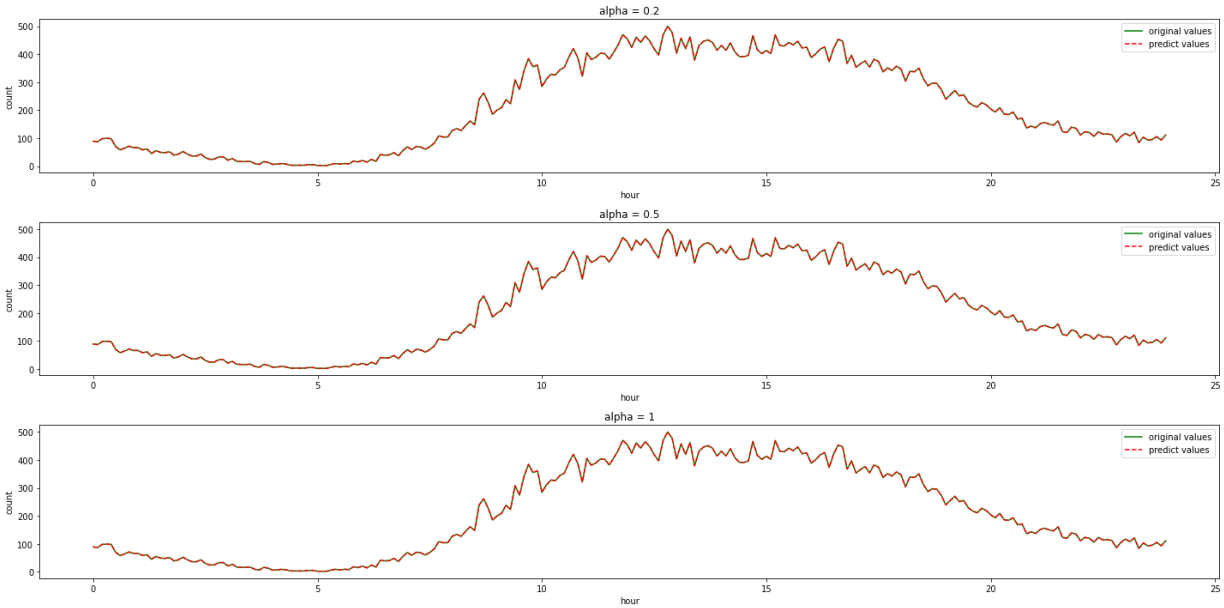


In [71]:
```
from sklearn.linear_model import Ridge

true_x = monday.dropna().iloc[:, 1].values
true_y = monday.dropna().iloc[:, 0].values
count = 1
plt.figure()
for n in [0.2, 0.5, 1]:
    ri = Ridge(alpha=n)
    ri.fit(true_x.reshape(1, -1), true_y.reshape(1, -1))
    plt.subplot(3, 1, count)
    plt.plot(true_x, true_y, 'g-', label='original values')
    plt.plot(true_x, np.dot(true_x, ri.coef_) + ri.intercept_, 'r--', label='pred
    plt.xlabel("hour")
    plt.ylabel("count")
    plt.title(f'alpha = {n}')
    plt.legend()
    count += 1
plt.tight_layout()
plt.show()
```



In [73]:
```
true_x = saturday.dropna().iloc[:, 1].values
true_y = saturday.dropna().iloc[:, 0].values
count = 1
plt.figure()
for n in [0.2, 0.5, 1]:
    ri = Ridge(alpha=n)
    ri.fit(true_x.reshape(1, -1), true_y.reshape(1, -1))
    plt.subplot(3, 1, count)
    plt.plot(true_x, true_y, 'g-', label='original values')
    plt.plot(true_x, np.dot(true_x, ri.coef_) + ri.intercept_, 'r--', label='pred
    plt.xlabel("hour")
    plt.ylabel("count")
    plt.title(f'alpha = {n}')
    plt.legend()
    count += 1
plt.tight_layout()
plt.show()
```



In [ ]: