Uber New York Trip Data Analysis

In [1]:

```
import pandas as pd
import numpy as np                          # import library to analysis and preprosing the Data
import matplotlib.pyplot as plt            # import the library to batter visulization the data
import seaborn as sns
```

In [2]:

```
import os   # to check path and intract with system we import os module
```

In [3]:

```
os.listdir(path = 'C:\\Users\\RAJAT\\Desktop\\my notes\\Data Analysis Projects\\uber-pickups-in-new-york-city')
# to show the uber new york list of data csv file avilable in my pc
```

Out[3]:

```
['other-American_B01362.csv',
 'other-Carmel_B00256.csv',
 'other-Dial7_B00887.csv',
 'other-Diplo_B01196.csv',
 'other-Federal_02216.csv',
 'other-FHV-services_jan-aug-2015.csv',
 'other-Firstclass_B01536.csv',
 'other-Highclass_B01717.csv',
 'other-Lyft_B02510.csv',
 'other-Prestige_B01338.csv',
 'other-Skyline_B00111.csv',
 'Uber-Jan-Feb-FOIL.csv',
 'uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-janjune-15.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [4]:

```
File = os.listdir(path = 'C:\\Users\\RAJAT\\Desktop\\my notes\\Data Analysis Projects\\uber-pickups-in-new-york-city')[-7:]
File
```

Out[4]:

```
['uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-janjune-15.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [5]:

```
File.remove('uber-raw-data-janjune-15.csv') # to remove the file
```

collect and importing the Data

In [6]:

```
Path = 'C:\\Users\\RAJAT\\Desktop\\my notes\\Data Analysis Projects\\uber-pickups-in-new-york-city
' # path of our csv files
```

In [7]:

```
final = pd.DataFrame()                                                    # create blank c
a frame

for f in File:
    Data = pd.read_csv(Path+"/"+f,encoding="utf-8")                       # read all data
iles
    final = pd.concat([Data,final])                                  # cancat all files with the bi
nk final dataframe
```

In [8]:

```
final.shape                    # check the dimansion of data ("its very Big Amount of Data")
```

Out[8]:

```
(4534327, 4)
```

Preparing Data For Analysis (Data Preprocessing)

In [9]:

```
df = final.copy() # create a copy of orignal data
```

In [10]:

```
df.head()
```

Out[10]:

	Date/Time	Lat	Lon	Base
0	9/1/2014 0:01:00	40.2201	-74.0021	B02512
1	9/1/2014 0:01:00	40.7500	-74.0027	B02512
2	9/1/2014 0:03:00	40.7559	-73.9864	B02512
3	9/1/2014 0:06:00	40.7450	-73.9889	B02512
4	9/1/2014 0:11:00	40.8145	-73.9444	B02512

In [11]:

```
df.dtypes               # checking the Data types of our DataFrame
```

Out[11]:

```
Date/Time     object
Lat          float64
Lon          float64
Base          object
dtype: object
```

In [12]:

```
# Data time having object data so we need to convert itno pandas date time module
```

In [13]:

```
df['Date/Time'] = pd.to_datetime(df['Date/Time'],format="%m/%d/%Y %H:%M:%S") # covert and formate
Date Time
```

In [14]:

```
df.dtypes
```

Out[14]:

```
Date/Time     datetime64[ns]
Lat                  float64
Lon                  float64
Base                  object
dtype: object
```

In [15]:

```
df["WeekDays"] = df['Date/Time'].dt.day_name()
```

In [16]:

```
df["Day"] = df['Date/Time'].dt.day
df["Month"] = df['Date/Time'].dt.month
df["Year"] = df['Date/Time'].dt.year
df["Hours"] = df['Date/Time'].dt.hour
df["Minutes"] = df['Date/Time'].dt.minute
```

In [17]:

```
df.head()
```

Out[17]:

	Date/Time	Lat	Lon	Base	WeekDays	Day	Month	Year	Hours	Minutes
0	2014-09-01 00:01:00	40.2201	-74.0021	B02512	Monday	1	9	2014	0	1
1	2014-09-01 00:01:00	40.7500	-74.0027	B02512	Monday	1	9	2014	0	1
2	2014-09-01 00:03:00	40.7559	-73.9864	B02512	Monday	1	9	2014	0	3
3	2014-09-01 00:06:00	40.7450	-73.9889	B02512	Monday	1	9	2014	0	6
4	2014-09-01 00:11:00	40.8145	-73.9444	B02512	Monday	1	9	2014	0	11

In [18]:

```
df.tail()
```

Out[18]:

	Date/Time	Lat	Lon	Base	WeekDays	Day	Month	Year	Hours	Minutes
564511	2014-04-30 23:22:00	40.7640	-73.9744	B02764	Wednesday	30	4	2014	23	22
564512	2014-04-30 23:26:00	40.7629	-73.9672	B02764	Wednesday	30	4	2014	23	26
564513	2014-04-30 23:31:00	40.7443	-73.9889	B02764	Wednesday	30	4	2014	23	31
564514	2014-04-30 23:32:00	40.6756	-73.9405	B02764	Wednesday	30	4	2014	23	32
564515	2014-04-30 23:48:00	40.6880	-73.9608	B02764	Wednesday	30	4	2014	23	48

In [19]:

```
df.dtypes
```

Out[19]:

```
Date/Time     datetime64[ns]
Lat                  float64
Lon                  float64
Base                  object
WeekDays              object
Day                    int64
Month                  int64
Year                   int64
```

```
Hours                int64
Minutes              int64
dtype: object
```

1. Analysis data --->> journey according Weekdays

In [20]:

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

Out[20]:

```
Thursday     755145
Friday       741139
Wednesday    696488
Tuesday      663789
Saturday     646114
Monday       541472
Sunday       490180
Name: WeekDays, dtype: int64
```

In [21]:

```
df['WeekDays'].value_counts().index
```

Out[21]:

```
Index(['Thursday', 'Friday', 'Wednesday', 'Tuesday', 'Saturday', 'Monday',
       'Sunday'],
      dtype='object')
```

In [22]:

```
import plotly.express as px
```

In [23]:

```
px.bar(x = df['WeekDays'].value_counts().index,
      y = df['WeekDays'].value_counts())
```

so in weekday thursday Having high Journeys and sunday have low journeys

so on Thursday we required more uber cabs becouse we have very high rush on thursday

2. Analysis data --->> Journey according Hours

In [25]:

```
(df["Month"].unique())
```

Out[25]:

```
array([9, 5, 6, 7, 8, 4], dtype=int64)
```

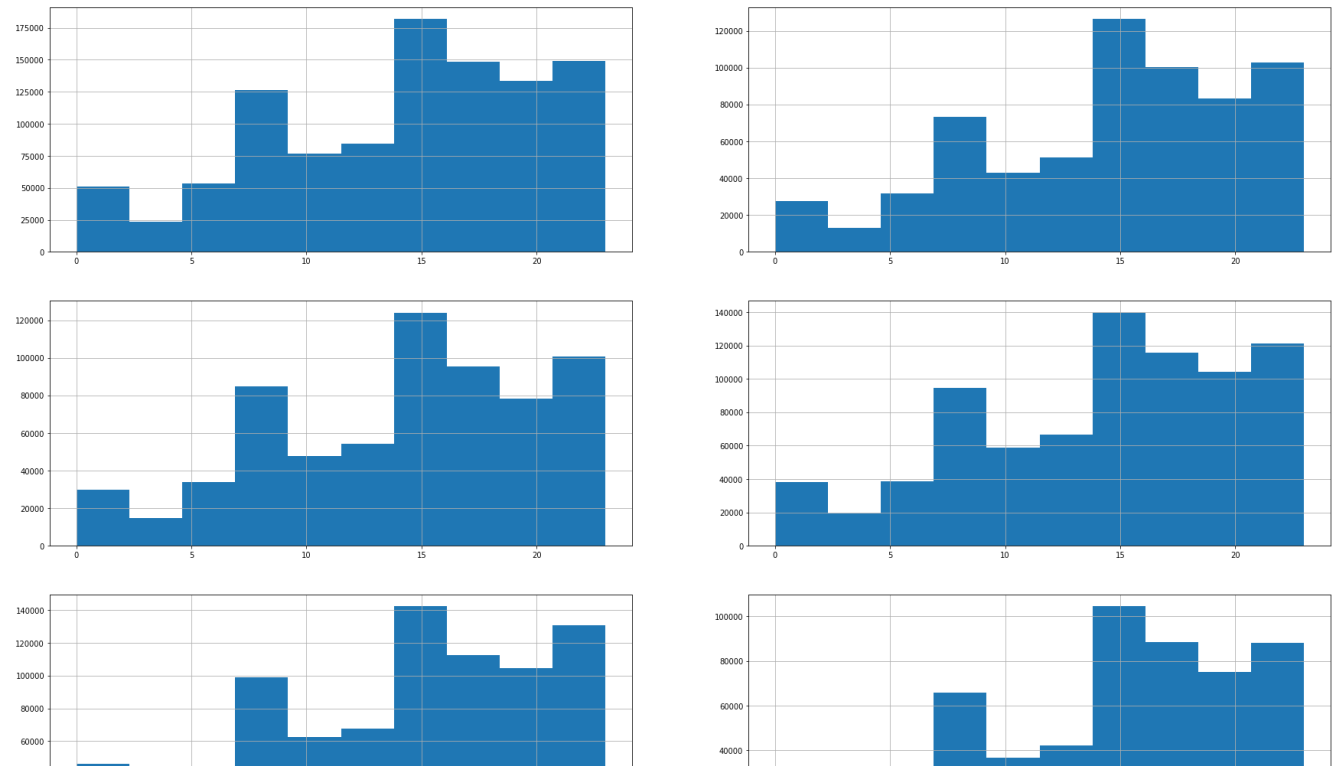
In [26]:

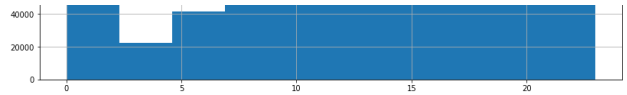
```
for i,month in enumerate(df["Month"].unique()):
    print(i)
    print(month)
```

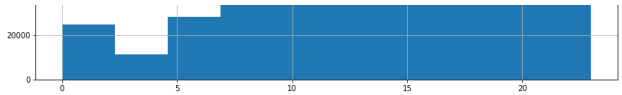
```
0
9
1
5
2
6
3
7
4
8
5
4
```

In [27]:

```
plt.figure(figsize=(30,20))
for i,month in enumerate(df["Month"].unique()):
    plt.subplot(3,2,i+1)
    df[df["Month"]==month]['Hours'].hist()
```







so in above graph we see that in evening Time 3:00 pm to 9:00 pm having very high rush.

3.which month having maximum rides ?

In [29]:

```
df.head()
```

Out[29]:

	Date/Time	Lat	Lon	Base	WeekDays	Day	Month	Year	Hours	Minutes
0	2014-09-01 00:01:00	40.2201	-74.0021	B02512	Monday	1	9	2014	0	1
1	2014-09-01 00:01:00	40.7500	-74.0027	B02512	Monday	1	9	2014	0	1
2	2014-09-01 00:03:00	40.7559	-73.9864	B02512	Monday	1	9	2014	0	3
3	2014-09-01 00:06:00	40.7450	-73.9889	B02512	Monday	1	9	2014	0	6
4	2014-09-01 00:11:00	40.8145	-73.9444	B02512	Monday	1	9	2014	0	11

In [30]:

```
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
```

In [31]:

```
df.groupby("Month")['Hours'].count()
```

Out[31]:

```
Month
4      564516
5      652435
6      663844
7      796121
8      829275
9     1028136
Name: Hours, dtype: int64
```

In [32]:

```
df.groupby("Month")['Hours'].count().index
```

Out[32]:

```
Int64Index([4, 5, 6, 7, 8, 9], dtype='int64', name='Month')
```

In [33]:

```
G1=go.Bar(x=df.groupby("Month")['Hours'].count().index,y=df.groupby("Month")['Hours'].count(),)
G1
```

Out[33]:

```
Bar({
    'x': array([4, 5, 6, 7, 8, 9], dtype=int64),
    'y': array([ 564516,  652435,  663844,  796121,  829275, 1028136], dtype=int64)
})
```

In [34]:

```
iplot([G1])
```

so 9 month sept having maximum rides and in month 4 having low rides

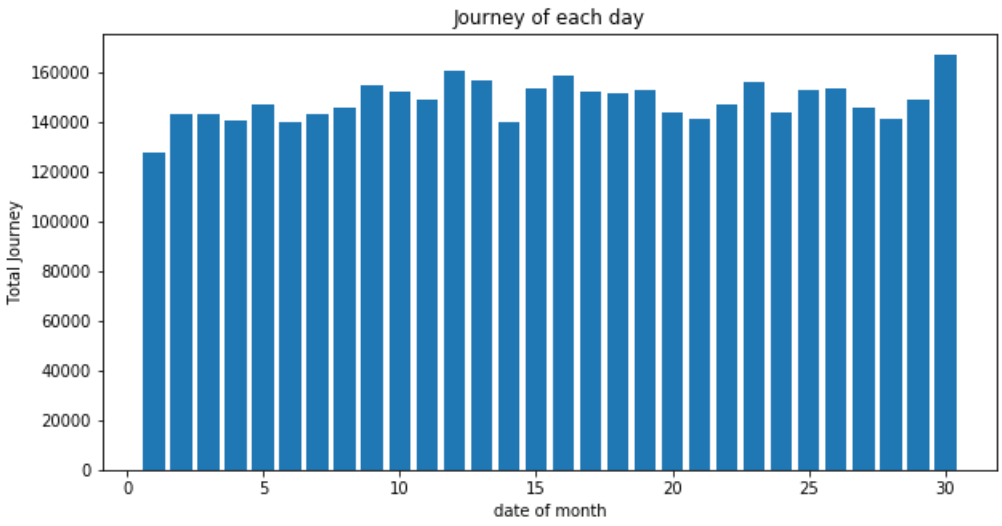
4.jurneny of each day

In [36]:

```
plt.figure(figsize=(10,5))
plt.hist(df['Day'],bins = 30,rwidth=0.8,range=(0.5,30.5))
plt.xlabel("date of month")
plt.ylabel("Total Journey")
plt.title("Journey of each day")
```

Out[36]:

```
Text(0.5, 1.0, 'Journey of each day')
```

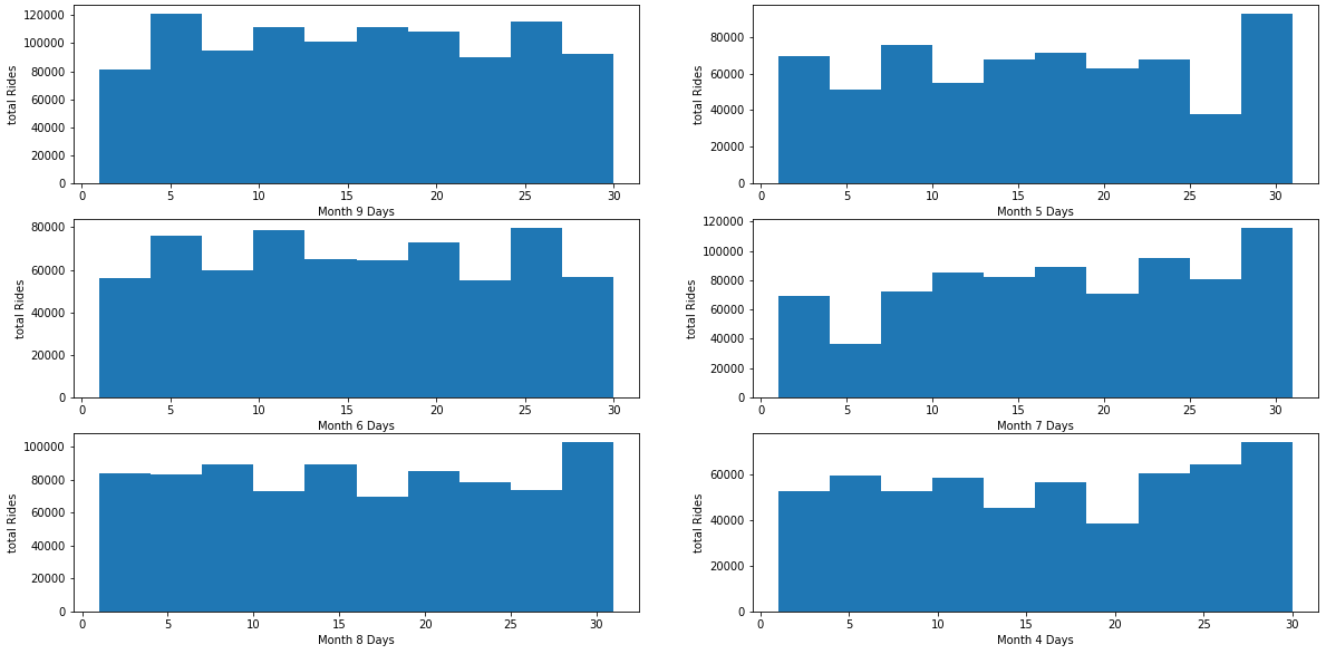


so that day 30 having Higest Rides

5.Analysis of total rides Month Wise

In [38]:

```
plt.figure(figsize=(20,10))
for i,month in enumerate(df["Month"].unique(),1):
    plt.subplot(3,2,i)
    df_out = df[df['Month']==month]
    plt.hist(df_out['Day'])
    plt.xlabel(f"Month {month} Days")
    plt.ylabel("total Rides")
```



last week of each month having higest Rush

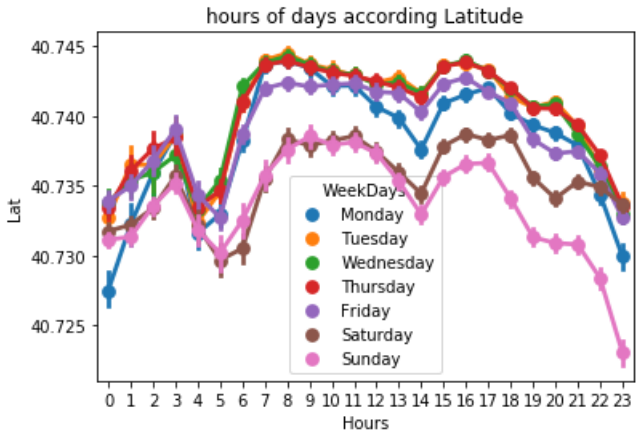
6.analysis hours of days according Latitude

In [40]:

```
ax = sns.pointplot(x = "Hours", y = "Lat",data = df , hue= "WeekDays")
ax.set_title("hours of days according Latitude")
```

Out[40]:

```
Text(0.5, 1.0, 'hours of days according Latitude')
```



so Thursday (Red graph) having very Rush

In [41]:

```
df.head()
```

Out[41]:

	Date/Time	Lat	Lon	Base	WeekDays	Day	Month	Year	Hours	Minutes
0	2014-09-01 00:01:00	40.2201	-74.0021	B02512	Monday	1	9	2014	0	1
1	2014-09-01 00:01:00	40.7500	-74.0027	B02512	Monday	1	9	2014	0	1
2	2014-09-01 00:03:00	40.7559	-73.9864	B02512	Monday	1	9	2014	0	3
3	2014-09-01 00:06:00	40.7450	-73.9889	B02512	Monday	1	9	2014	0	6
4	2014-09-01 00:11:00	40.8145	-73.9444	B02512	Monday	1	9	2014	0	11

7.Analysis which base number get popular by Month

In [42]:

```
Base = df.groupby(["Base","Month"])["Date/Time"].count().reset_index()
Base
```

Out[42]:

	Base	Month	Date/Time
0	B02512	4	35536
1	B02512	5	36765
2	B02512	6	32509
3	B02512	7	35021
4	B02512	8	31472
5	B02512	9	34370
6	B02598	4	183263
7	B02598	5	260549
8	B02598	6	242975
9	B02598	7	245597
10	B02598	8	220129
11	B02598	9	240600
12	B02617	4	108001
13	B02617	5	122734
14	B02617	6	184460
15	B02617	7	310160
16	B02617	8	355803
17	B02617	9	377695
18	B02682	4	227808
19	B02682	5	222883
20	B02682	6	194926
21	B02682	7	196754
22	B02682	8	173280
23	B02682	9	197138
24	B02764	4	9908
25	B02764	5	9504
26	B02764	6	8974

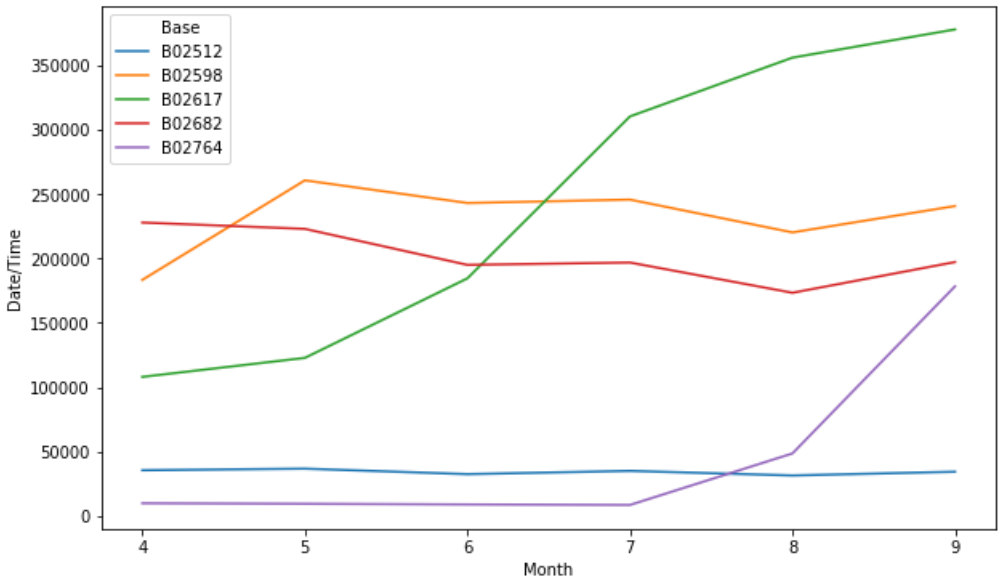
	Base	Month	Date/Time
27	B02764	7	8589
28	B02764	8	48591
29	B02764	9	178333

In [43]:

```
plt.figure(figsize=(10,6))
sns.lineplot(x='Month',y='Date/Time',hue="Base",data=Base)
```

Out[43]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a015ba448>
```



so by this graph base no. B02617 green line is most popular in month 7,8,9

8.Cross Analysis of Data

1.heatmap acc. to Hours and weekdays

2.heatmap acc. to Hours and Days

3.heatmap acc. to Month and Days

4.heatmap acc. to Month and weekdays

In [45]:

```
def count_rows(rows):
    return len(rows)
```

In [46]:

```
cross = df.groupby(['WeekDays',"Hours"]).apply(count_rows)
```

In [47]:

```
cross
```

Out[47]:

```
WeekDays   Hours
Friday     0        13716
           1         8163
```

```
1          8163
2          5350
3          6930
4          8806
           ...
Wednesday  19        47017
           20        47772
           21        44553
           22        32868
           23        18146
Length: 168, dtype: int64
```

In [48]:

```
pivot  = cross.unstack()
pivot
```

Out[48]:

Hours	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	
WeekDays																		
Friday	13716	8163	5350	6930	8806	13450	23412	32061	31509	25230	...	36206	43673	48169	51961	54762	49595	43
Monday	6436	3737	2938	6232	9640	15032	23746	31159	29265	22197	...	28157	32744	38770	42023	37000	34159	32
Saturday	27633	19189	12710	9542	6846	7084	8579	11014	14411	17669	...	31418	38769	43512	42844	45883	41098	38
Sunday	32877	23015	15436	10597	6374	6169	6596	8728	12128	16401	...	28151	31112	33038	31521	28291	25948	25
Thursday	9293	5290	3719	5637	8505	14169	27065	37038	35431	27812	...	36699	44442	50560	56704	55825	51907	51
Tuesday	6237	3509	2571	4494	7548	14241	26872	36599	33934	25023	...	34846	41338	48667	55500	50186	44789	44
Wednesday	7644	4324	3141	4855	7511	13794	26943	36495	33826	25635	...	35148	43388	50684	55637	52732	47017	47

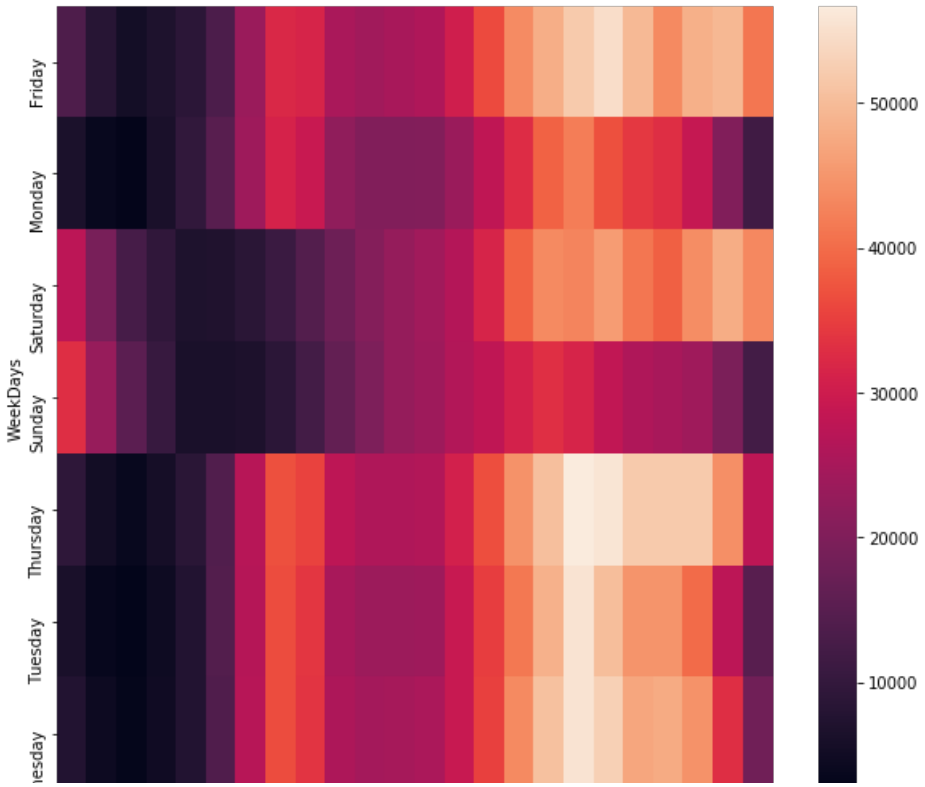
7 rows × 24 columns

In [49]:

```
plt.figure(figsize=(10,9))
sns.heatmap(pivot)
```

Out[49]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a0149ed08>
```



so acc to heatmap evening(4:00pm to 11:00pm) time having most Rush

In [51]:

```
# i will create a function for further all 3 analysis
```

In [52]:

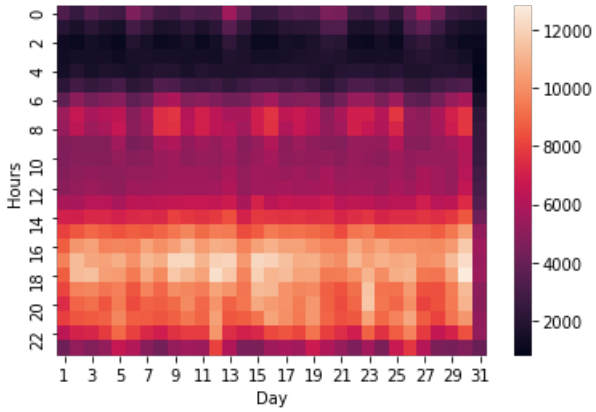
```
def heatmap(col1,col2):
    cross = df.groupby([col1,col2]).apply(count_rows)
    pivot  = cross.unstack()
    return sns.heatmap(pivot)
```

In [53]:

```
heatmap("Hours","Day")  # 2nd
```

Out[53]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a011e0b08>
```

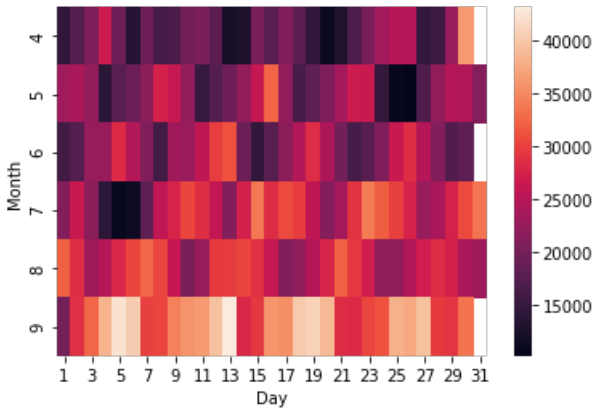


In [54]:

```
heatmap("Month","Day") # 3rd
```

Out[54]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a014d75c8>
```

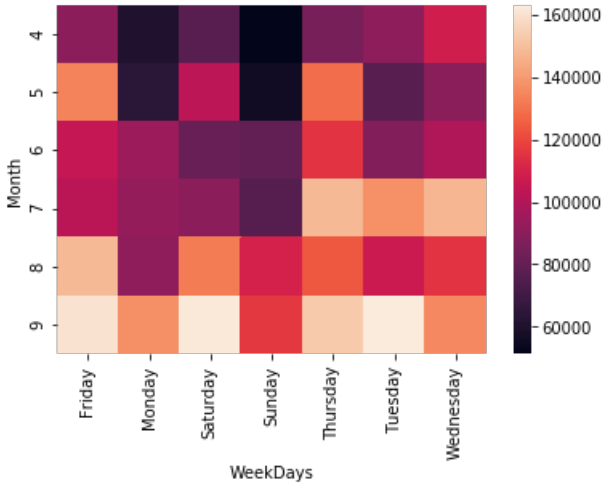


In [55]:

```
heatmap("Month","WeekDays") # 4th
```

Out[55]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a0150b408>
```



Analysing the results

We observe that the number of trips increases each month, we can say that from April to September 2014, Uber was in a continuous improvement process.

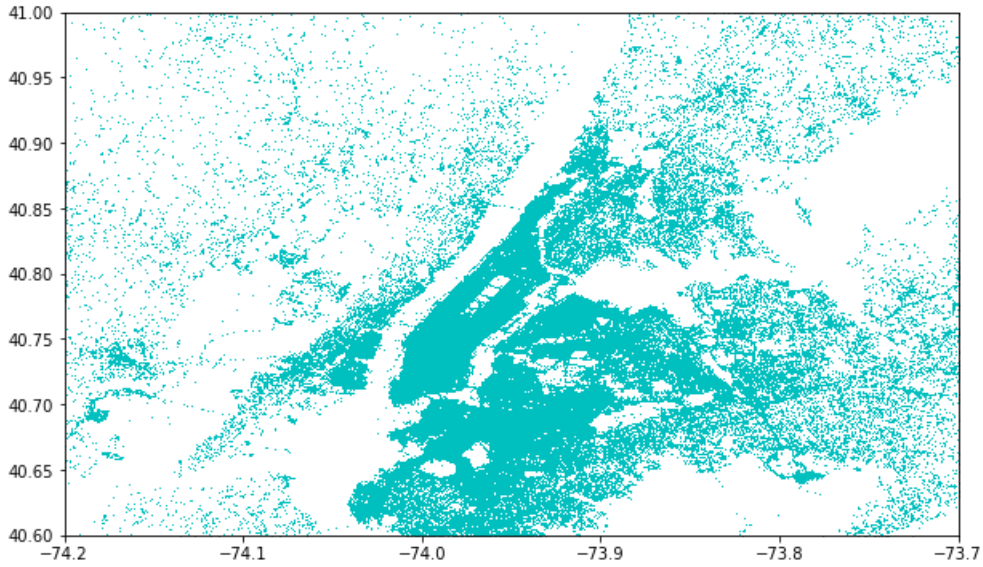
9.Analysis Location of Data Points

In [56]:

```
plt.figure(figsize=(10,6))
plt.plot(df['Lon'],df['Lat'],"c+",ms=0.5)
plt.xlim(-74.2, -73.7)
plt.ylim(40.6,41)
```

Out[56]:

```
(40.6, 41)
```



We can see a number of hot spots here. Midtown Manhattan is clearly a huge bright spot.

& these are made from Midtown to Lower Manhattan.

Followed by Upper Manhattan and the Heights of Brooklyn.

In [57]:

```
df_out=df[df['WeekDays']=='Sunday']
df_out.head()
```

Out[57]:

	Date/Time	Lat	Lon	Base	WeekDays	Day	Month	Year	Hours	Minutes
8011	2014-09-07 00:00:00	40.7341	-74.0005	B02512	Sunday	7	9	2014	0	0
8012	2014-09-07 00:00:00	40.7344	-73.9900	B02512	Sunday	7	9	2014	0	0
8013	2014-09-07 00:00:00	40.7806	-73.9582	B02512	Sunday	7	9	2014	0	0
8014	2014-09-07 00:01:00	40.7293	-73.9859	B02512	Sunday	7	9	2014	0	1
8015	2014-09-07 00:01:00	40.7713	-74.0133	B02512	Sunday	7	9	2014	0	1

In [58]:

```
df_out.groupby(['Lat','Lon'])['WeekDays'].count().reset_index()
```

Out[58]:

	Lat	Lon	WeekDays
0	39.9374	-74.0722	1
1	39.9378	-74.0721	1
2	39.9384	-74.0742	1
3	39.9385	-74.0734	1
4	39.9415	-74.0736	1
...	...	...	...
209225	41.3141	-74.1249	1
209226	41.3180	-74.1298	1
209227	41.3195	-73.6905	1
209228	41.3197	-73.6903	1
209229	42.1166	-72.0666	1

209230 rows × 3 columns

In [59]:

```
import folium
from folium.plugins import HeatMap
basemap = folium.Map()
```

In [90]:

```
HeatMap(df_out.groupby(['Lat','Lon'])['WeekDays'].count().reset_index(),zoom=20,radius=15).add_to(b
asemap)
basemap
```

Out[90]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [61]:

```
#Lets create a function for a specific day
```

In [62]:

```
def Heatmap(df,Day):
    df_out=df[df['WeekDays']==Day]
    df_out.groupby(['Lat','Lon'])['WeekDays'].count().reset_index()
    HeatMap(df_out.groupby(['Lat','Lon'])['WeekDays'].count().reset_index(),zoom=20,radius=15).add_
to(basemap)
    return basemap
```

In [63]:

```
Heatmap(df,'Monday')
```

Out[63]:

we see on the map Midtown Manhattan is clearly a huge bright spot

10.Analysis the Data of Uber Jan to June

In [64]:

```
Uber = pd.read_csv("C://Users//RAJAT//Desktop//my notes//Data Analysis Projects//uber-pickups-in-new-york-city//uber-raw-data-janjune-15.csv")
```

In [65]:

```
Uber.head()
```

Out[65]:

	Dispatching_base_num	Pickup_date	Affiliated_base_num	locationID
0	B02617	2015-05-17 09:47:00	B02617	141
1	B02617	2015-05-17 09:47:00	B02617	65
2	B02617	2015-05-17 09:47:00	B02617	100
3	B02617	2015-05-17 09:47:00	B02774	80
4	B02617	2015-05-17 09:47:00	B02617	90

In [66]:

```
Uber.dtypes
```

Out[66]:

```
Dispatching_base_num    object
Pickup_date             object
Affiliated_base_num     object
locationID               int64
dtype: object
```

In [67]:

```
Uber['Pickup_date'] = pd.to_datetime(Uber['Pickup_date'],format='%Y-%m-%d %H:%M:%S')
```

In [68]:

```
Uber.dtypes
```

Out[68]:

```
Dispatching_base_num            object
Pickup_date             datetime64[ns]
Affiliated_base_num             object
locationID                       int64
dtype: object
```

In [69]:

```
Uber['weekday']=Uber['Pickup_date'].dt.day_name()
Uber['Day']=Uber['Pickup_date'].dt.day
Uber['Minute']=Uber['Pickup_date'].dt.minute
Uber['Month']=Uber['Pickup_date'].dt.month
Uber['Hour']=Uber['Pickup_date'].dt.hour
```

In [70]:

```
Uber.head()
```

Out[70]:

	Dispatching_base_num	Pickup_date	Affiliated_base_num	locationID	weekday	Day	Minute	Month	Hour

	Dispatching_base_num	Pickup_date	Affiliated_base_num	locationID	weekday	Day	Minute	Month	Hour
0	B02617	2015-05-17 09:47:00	B02617	141	Sunday	17	47	5	9
1	B02617	2015-05-17 09:47:00	B02617	65	Sunday	17	47	5	9
2	B02617	2015-05-17 09:47:00	B02617	100	Sunday	17	47	5	9
3	B02617	2015-05-17 09:47:00	B02774	80	Sunday	17	47	5	9
4	B02617	2015-05-17 09:47:00	B02617	90	Sunday	17	47	5	9

11.Uber pickups by the month in NYC

In [71]:

```
Uber['Month'].value_counts().values
```

Out[71]:

```
array([2816895, 2695553, 2280837, 2263620, 2259773, 1953801], dtype=int64)
```

In [72]:

```
Uber['Month'].value_counts().index
```

Out[72]:

```
Int64Index([6, 5, 4, 2, 3, 1], dtype='int64')
```

In [73]:

```
px.bar(x=Uber['Month'].value_counts().index,
       y=Uber['Month'].value_counts().values)
```

so in month 6 haveing highest pickups

jan having lowest pickups

12.Analysing Peak in Newyork City

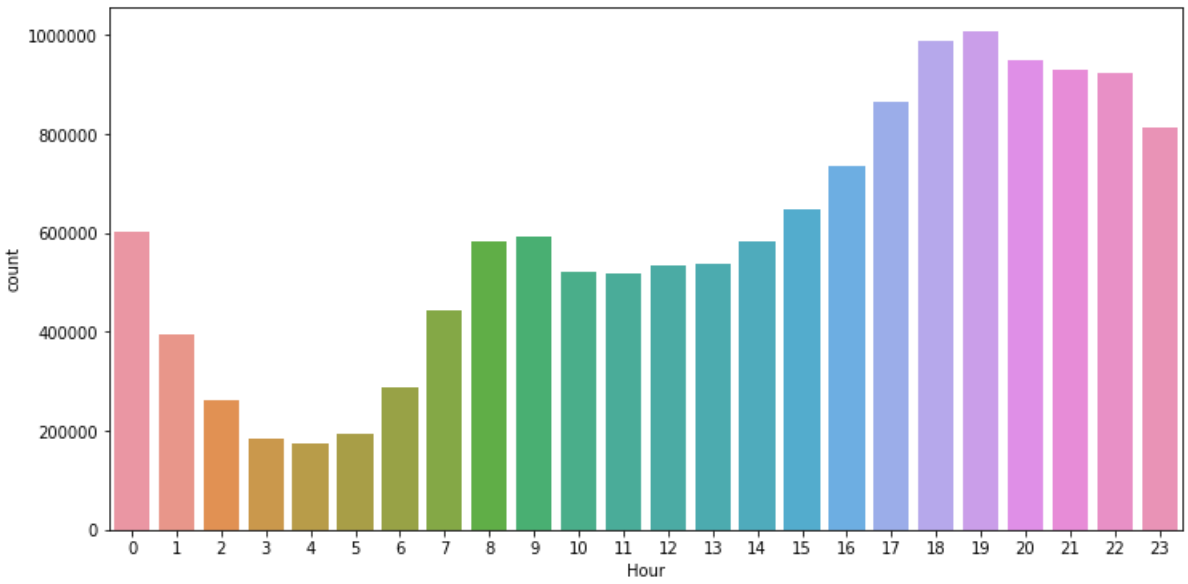
12.Analysing Rush in New york City

In [75]:

```
plt.figure(figsize=(12,6))
sns.countplot(Uber["Hour"])
```

Out[75]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a7ce76e08>
```



In evening time having high Rush (4:00 to 11:00pm)

In afternoon Rush are equal not much more But in morning time Rush is very low

13.Analysing In-Depth Analysis of Rush in New york City Day & hour wise

In [76]:

```
Uber.groupby(["weekday","Hour"])['Pickup_date'].count()
```

Out[76]:

```
weekday    Hour
Friday     0         85939
           1         46616
           2         28102
           3         19518
           4         23575
                     ...
Wednesday  19       143751
           20       136003
           21       133993
           22       127026
           23        99490
Name: Pickup_date, Length: 168, dtype: int64
```

In [77]:

```
summary = Uber.groupby(["weekday","Hour"])['Pickup_date'].count().reset_index()
```

In [78]:

```
summary.head()
```

Out[78]:

	weekday	Hour	Pickup_date
0	Friday	0	85939
1	Friday	1	46616
2	Friday	2	28102
3	Friday	3	19518
4	Friday	4	23575

In [79]:

```
summary=summary.rename(columns = {'Pickup_date':'Counts'})
summary
```

Out[79]:

	weekday	Hour	Counts
0	Friday	0	85939
1	Friday	1	46616
2	Friday	2	28102
3	Friday	3	19518
4	Friday	4	23575
...	...	...	...
163	Wednesday	19	143751
164	Wednesday	20	136003
165	Wednesday	21	133993
166	Wednesday	22	127026
167	Wednesday	23	99490

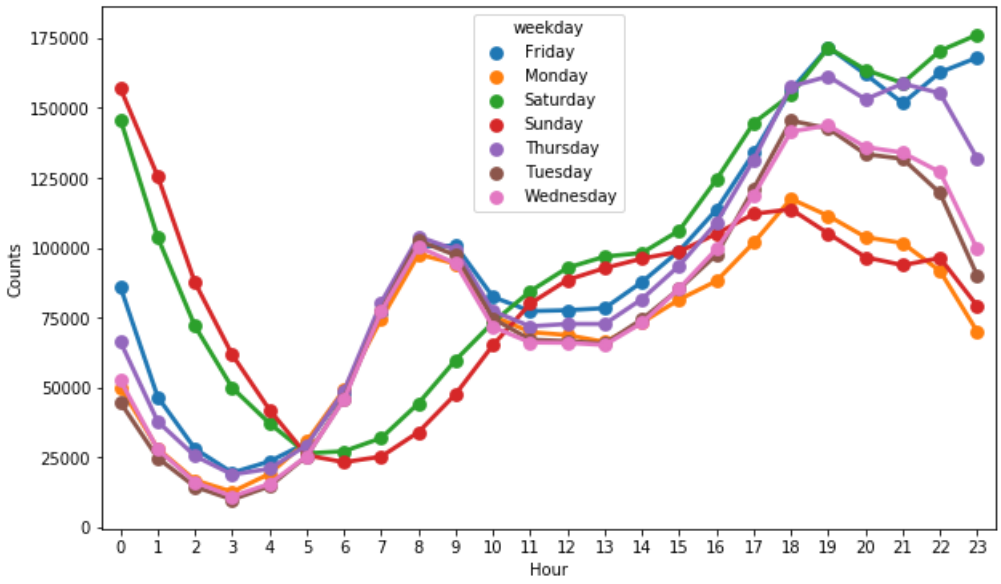
168 rows × 3 columns

In [80]:

```
plt.figure(figsize=(10,6))
sns.pointplot(x="Hour", y="Counts", hue="weekday", data=summary)
```

Out[80]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22af1c8af88>
```



saturday in evening time Rush is having much more

14.Which dispatching_base_number is highly active

In [81]:

```
uber_foil = pd.read_csv("C://Users//RAJAT//Desktop//my notes//Data Analysis Projects//uber-pickups-in-new-york-city//Uber-Jan-Feb-FOIL.csv")
```

In [82]:

```
uber_foil.head()
```

Out[82]:

	dispatching_base_number	date	active_vehicles	trips
0	B02512	1/1/2015	190	1132
1	B02765	1/1/2015	225	1765
2	B02764	1/1/2015	3427	29421
3	B02682	1/1/2015	945	7679
4	B02617	1/1/2015	1228	9537

In [83]:

```
uber_foil["dispatching_base_number"].unique()
```

Out[83]:

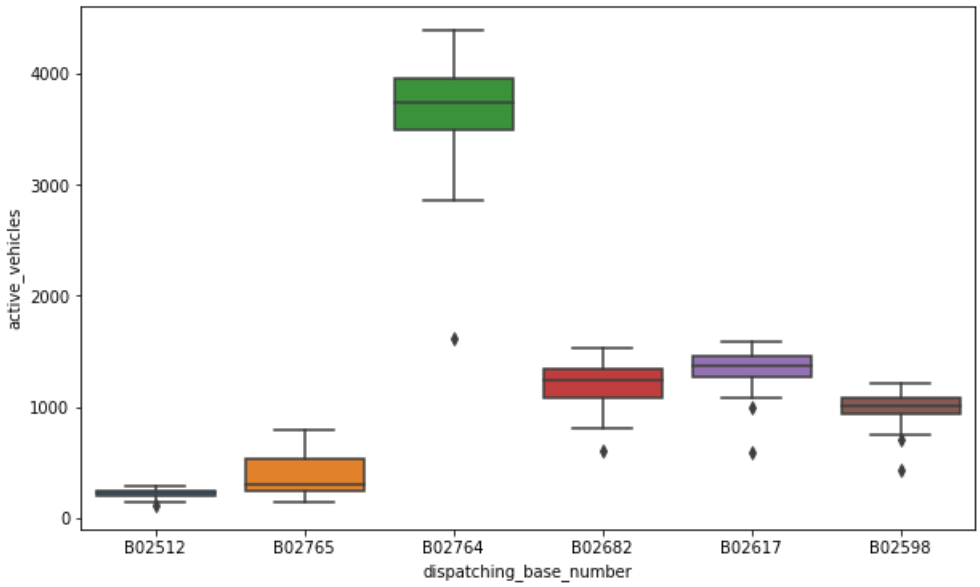
```
array(['B02512', 'B02765', 'B02764', 'B02682', 'B02617', 'B02598'],
      dtype=object)
```

In [84]:

```
plt.figure(figsize=(10,6))
sns.boxplot(x="dispatching_base_number",y="active_vehicles",data=uber_foil)
```

Out[84]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a94ac5788>
```



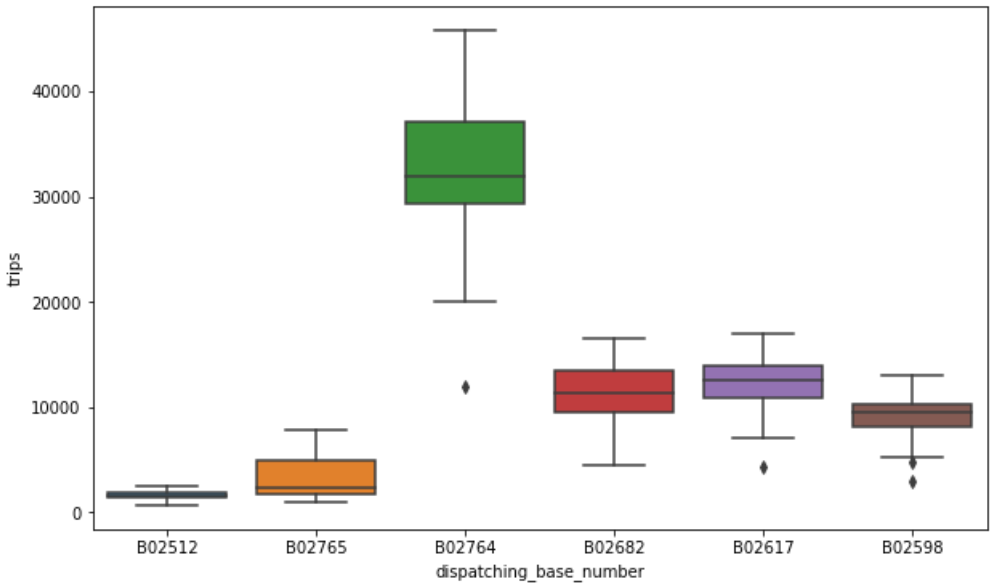
seems to have more number of Active Vehicles in B02764

In [95]:

```
plt.figure(figsize=(10,6))
sns.boxplot(x = 'dispatching_base_number', y = 'trips', data = uber_foil)
```

Out[95]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22a981bc2c8>
```



so Base No. B02764 is highly use on trips

15.how Average trips/vehicle inc/decreases with dates with each of base uber

In [86]:

```
# Finding the ratio of trips/active_vehicles
uber_foil['trips/vehicle'] = uber_foil['trips']/uber_foil['active_vehicles']
```

In [87]:

```
uber_foil.head()
```

Out[87]:

	dispatching_base_number	date	active_vehicles	trips	trips/vehicle
0	B02512	1/1/2015	190	1132	5.957895
1	B02765	1/1/2015	225	1765	7.844444
2	B02764	1/1/2015	3427	29421	8.585060
3	B02682	1/1/2015	945	7679	8.125926
4	B02617	1/1/2015	1228	9537	7.766287

In [88]:

```
uber_foil.set_index('date')
```

Out[88]:

	dispatching_base_number	active_vehicles	trips	trips/vehicle

date	dispatching_base_number	active_vehicles	trips	trips/vehicle
date				
1/1/2015	B02512	190	1132	5.957895
1/1/2015	B02765	225	1765	7.844444
1/1/2015	B02764	3427	29421	8.585060
1/1/2015	B02682	945	7679	8.125926
1/1/2015	B02617	1228	9537	7.766287
...	...	...	...	...
2/28/2015	B02764	3952	39812	10.073887
2/28/2015	B02617	1372	14022	10.220117
2/28/2015	B02682	1386	14472	10.441558
2/28/2015	B02512	230	1803	7.839130
2/28/2015	B02765	747	7753	10.378849

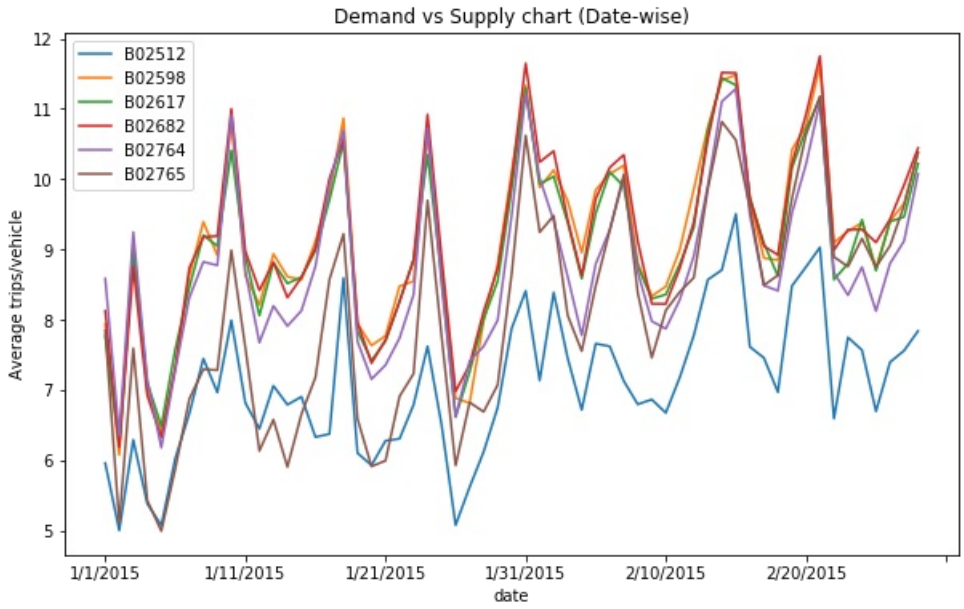
354 rows × 4 columns

In [89]:

```
plt.figure(figsize=(10,6))
uber_foil.set_index('date').groupby(['dispatching_base_number'])['trips/vehicle'].plot()
plt.ylabel('Average trips/vehicle')
plt.title('Demand vs Supply chart (Date-wise)')
plt.legend()
```

Out[89]:

```
<matplotlib.legend.Legend at 0x22a949e0b48>
```



Acc. to Graph base no. B02764 & B02682 & B02617 is avarage highly use

And Base No. B02512 is very low use

In [ ]:

This Complete Analysis of Uber New York trip Data I am find very insides for given row data of uber that are very helpful insides.

**Completed This Project

By Rajat Kumar pancholi

In [ ]: