Uber Trip Analysis Using Python

About The DataSet

The dataset consists of 6 files which is related to Trip details and Pickups details of Uber Taxi for Month April-2014 to Sept.-2014.This dataset contain various details of Uber Trips.

Objective of This Data Analysis

The objective of this Analysis as follow:-

1) To find out the Most Busy Hours of Trips, 2) Most Busy Day of Uber Taxi etc

STEP 1 - Importing Necessary Python Libraries

In [1]:
```
import pandas as pd # for the data processing,CSV file reading and Data cleaning
import numpy as np # for the N-dimensional array and linear algebra
import plotly.express as px # to visualize a variety of types of data
from datetime import datetime
from plotly.offline import iplot # to display the plot when working on offline
import plotly
plotly.offline.init_notebook_mode(connected=True)
```

STEP 2 - Loading Dataset and making single Dataset/DataFrame

- Reading all the 6 file and merging all file in single data frame.

In [2]:
```
apr=pd.read_csv("C:/Users/Lenovo/OneDrive/Desktop/Choice-Data/Uber/uber-raw-data-apr14.csv")
may=pd.read_csv("C:/Users/Lenovo/OneDrive/Desktop/Choice-Data/Uber/uber-raw-data-may14.csv")
june=pd.read_csv("C:/Users/Lenovo/OneDrive/Desktop/Choice-Data/Uber/uber-raw-data-jun14.csv")
july=pd.read_csv("C:/Users/Lenovo/OneDrive/Desktop/Choice-Data/Uber/uber-raw-data-jul14.csv")
aug=pd.read_csv("C:/Users/Lenovo/OneDrive/Desktop/Choice-Data/Uber/uber-raw-data-aug14.csv")
sep=pd.read_csv("C:/Users/Lenovo/OneDrive/Desktop/Choice-Data/Uber/uber-raw-data-sep14.csv")
```

In [3]:
```
apr["Month"]="Apr"
may["Month"]="May"
june["Month"]="June"
july["Month"]="July"
aug["Month"]="Aug"
sep["Month"]="Sep"
```

In [4]:
```
#Now concatenating the all  data in one DataFrame called as df
df = pd.concat([apr,may,june,july,aug,sep],axis=0)
df.head()
```

Out[4]:

	Date/Time	Lat	Lon	Base	Month
0	4/1/2014 0:11:00	40.7690	-73.9549	B02512	Apr
1	4/1/2014 0:17:00	40.7267	-74.0345	B02512	Apr
2	4/1/2014 0:21:00	40.7316	-73.9873	B02512	Apr
3	4/1/2014 0:28:00	40.7588	-73.9776	B02512	Apr
4	4/1/2014 0:33:00	40.7594	-73.9722	B02512	Apr

STEP 3 - Basic Descriptions of the Data and handling Null values

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
Date/Time    0
Lat          0
Lon          0
Base         0
Month        0
dtype: int64
```

In [6]:
```
df.shape
```

Out[6]:
```
(4534327, 5)
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4534327 entries, 0 to 1028135
Data columns (total 5 columns):
 #   Column     Dtype 
---  ------     ----- 
 0   Date/Time  object
 1   Lat        float64
 2   Lon        float64
 3   Base       object
 4   Month      object
dtypes: float64(2), object(3)
memory usage: 207.6+ MB
```

In [8]:
```
df['Date/Time'] = pd.to_datetime(df['Date/Time'], errors='coerce')
```

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4534327 entries, 0 to 1028135
Data columns (total 5 columns):
 #   Column     Dtype         
---  ------     -----         
 0   Date/Time  datetime64[ns]
 1   Lat        float64
 2   Lon        float64
 3   Base       object
 4   Month      object
dtypes: datetime64[ns](1), float64(2), object(2)
memory usage: 207.6+ MB
```

In [10]:
```
df['weekday']=df['Date/Time'].dt.day_name()
df['day']=df['Date/Time'].dt.day
df['hour']=df['Date/Time'].dt.hour
df['minute']=df['Date/Time'].dt.minute
```

In [11]:
```
df.head()
```

Out[11]:

	Date/Time	Lat	Lon	Base	Month	weekday	day	hour	minute
0	2014-04-01 00:11:00	40.7690	-73.9549	B02512	Apr	Tuesday	1	0	11
1	2014-04-01 00:17:00	40.7267	-74.0345	B02512	Apr	Tuesday	1	0	17
2	2014-04-01 00:21:00	40.7316	-73.9873	B02512	Apr	Tuesday	1	0	21
3	2014-04-01 00:28:00	40.7588	-73.9776	B02512	Apr	Tuesday	1	0	28
4	2014-04-01 00:33:00	40.7594	-73.9722	B02512	Apr	Tuesday	1	0	33

In [12]:
```
df.duplicated().sum()
```

Out[12]:
```
82581
```

Duplicate pickups do exist in the data set but we not know the accuracy of a pickup's Latitude/Longitude or Time, these pickups may be just have happened around the same time and around the same location. Therefore,due to not any conclusive reason here, we will assume that duplicate pickups are valid.

In [13]:
```
df.describe().T
```

Out[13]:

	count	mean	std	min	25%	50%	75%	max
Lat	4534327.0	40.739261	0.039950	39.6569	40.7211	40.7422	40.7610	42.1166
Lon	4534327.0	-73.973019	0.057267	-74.9290	-73.9965	-73.9834	-73.9653	-72.0666
day	4534327.0	15.943368	8.744902	1.0000	9.0000	16.0000	23.0000	31.0000
hour	4534327.0	14.218310	5.958759	0.0000	10.0000	15.0000	19.0000	23.0000
minute	4534327.0	29.400709	17.322384	0.0000	14.0000	29.0000	44.0000	59.0000

In [14]:
```
num_pickups = df.shape[0]
num_days = len(df[['Month', 'day']].drop_duplicates())
daily_avg = np.round(num_pickups/num_days, 0)

stats_raw = 'Number of Pickups: {}\nNumber of Days: {}\nAvg Daily Pickups: {}'
print(stats_raw.format(num_pickups, num_days, daily_avg))
```

```
Number of Pickups: 4534327
Number of Days: 183
Avg Daily Pickups: 24778.0
```

- According to the dataset, there were over **4.5 million Uber pickups between the April-2014 to September-2014. This corresponds to 24,778**

pickups every single day on average.

In [15]:
```
df.head()
```

Out[15]:

	Date/Time	Lat	Lon	Base	Month	weekday	day	hour	minute
0	2014-04-01 00:11:00	40.7690	-73.9549	B02512	Apr	Tuesday	1	0	11
1	2014-04-01 00:17:00	40.7267	-74.0345	B02512	Apr	Tuesday	1	0	17
2	2014-04-01 00:21:00	40.7316	-73.9873	B02512	Apr	Tuesday	1	0	21
3	2014-04-01 00:28:00	40.7588	-73.9776	B02512	Apr	Tuesday	1	0	28
4	2014-04-01 00:33:00	40.7594	-73.9722	B02512	Apr	Tuesday	1	0	33

Uber Trip Data Analysis and Visualization

Q.1. Which days of the week have the highest trip/fare? Why do you think that particular day receives the highest trip request?

In [16]:
```
weekday = df[['Month', 'weekday']].groupby(['weekday']).value_counts().reset_index()
weekday.columns = ['weekday', 'Month','Total Trip']
weekday.head()
```

Out[16]:

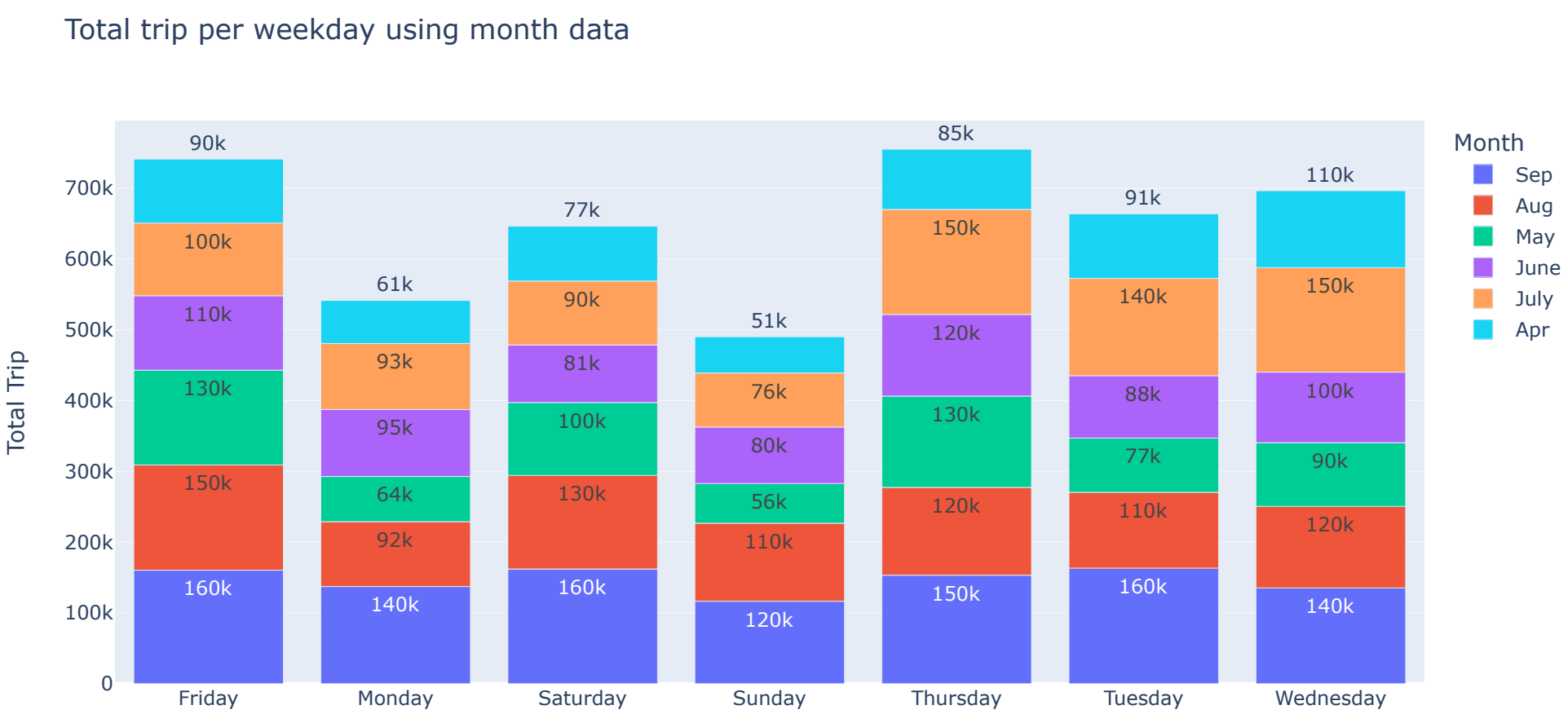
	weekday	Month	Total Trip
0	Friday	Sep	160380
1	Friday	Aug	148674
2	Friday	May	133991
3	Friday	June	105056
4	Friday	July	102735

In [17]:
```
weekday_1 = df[['weekday']].groupby(['weekday']).value_counts().reset_index()
weekday_1.columns = ['weekday','Total Trip']
weekday_1.head()
```

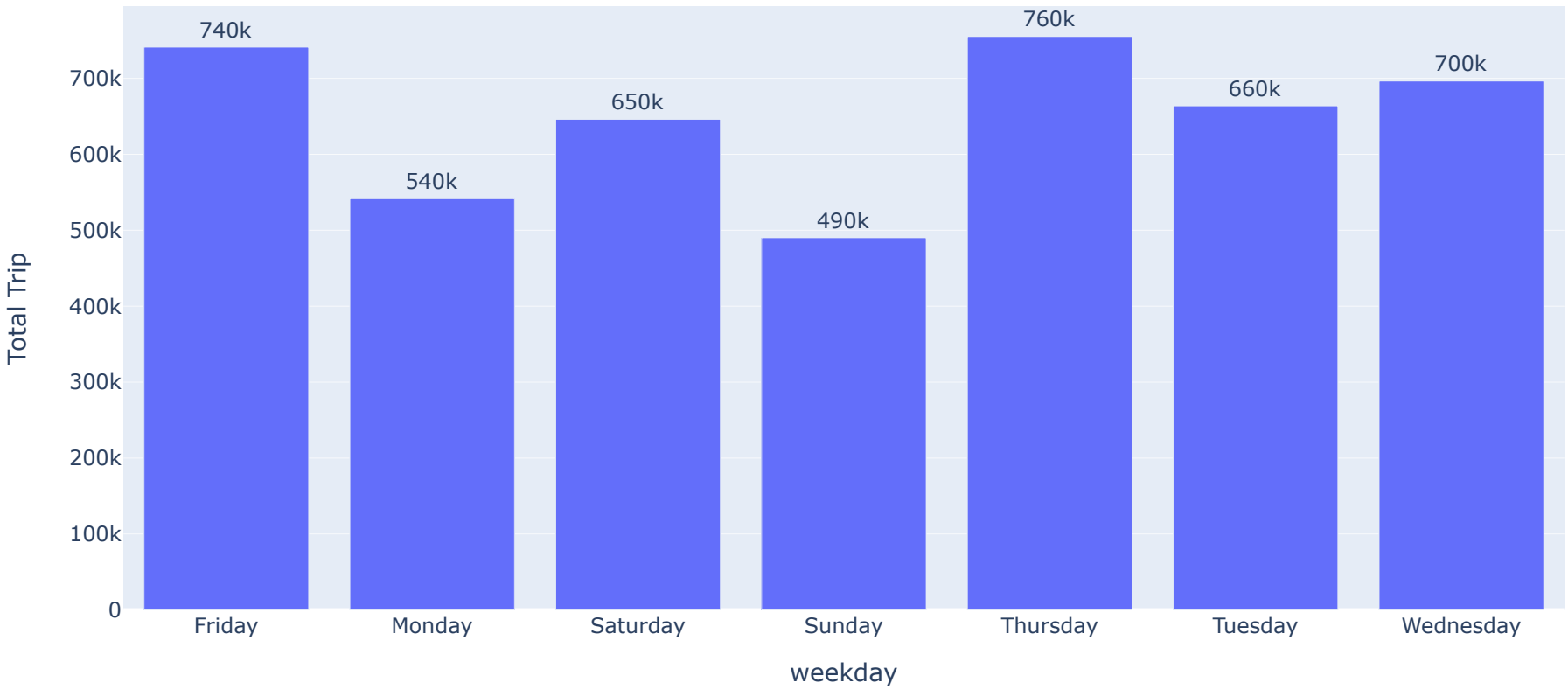
Out[17]:

	weekday	Total Trip
0	Friday	741139
1	Monday	541472
2	Saturday	646114
3	Sunday	490180
4	Thursday	755145

In [18]:
```
fig = px.bar(weekday, x='weekday',y='Total Trip',text_auto='.2s',title="Total trip per weekday using month data",color='Month')
fig.update_traces(textfont_size=12, textangle=0, textposition="outside", cliponaxis=False)
fig.show()
```



In [19]:
```
fig = px.bar(weekday_1, x='weekday',y='Total Trip',text_auto='.2s',title="Total Pickup per weekday")
fig.update_traces(textfont_size=12, textangle=0, textposition="outside", cliponaxis=False)
fig.show()
```

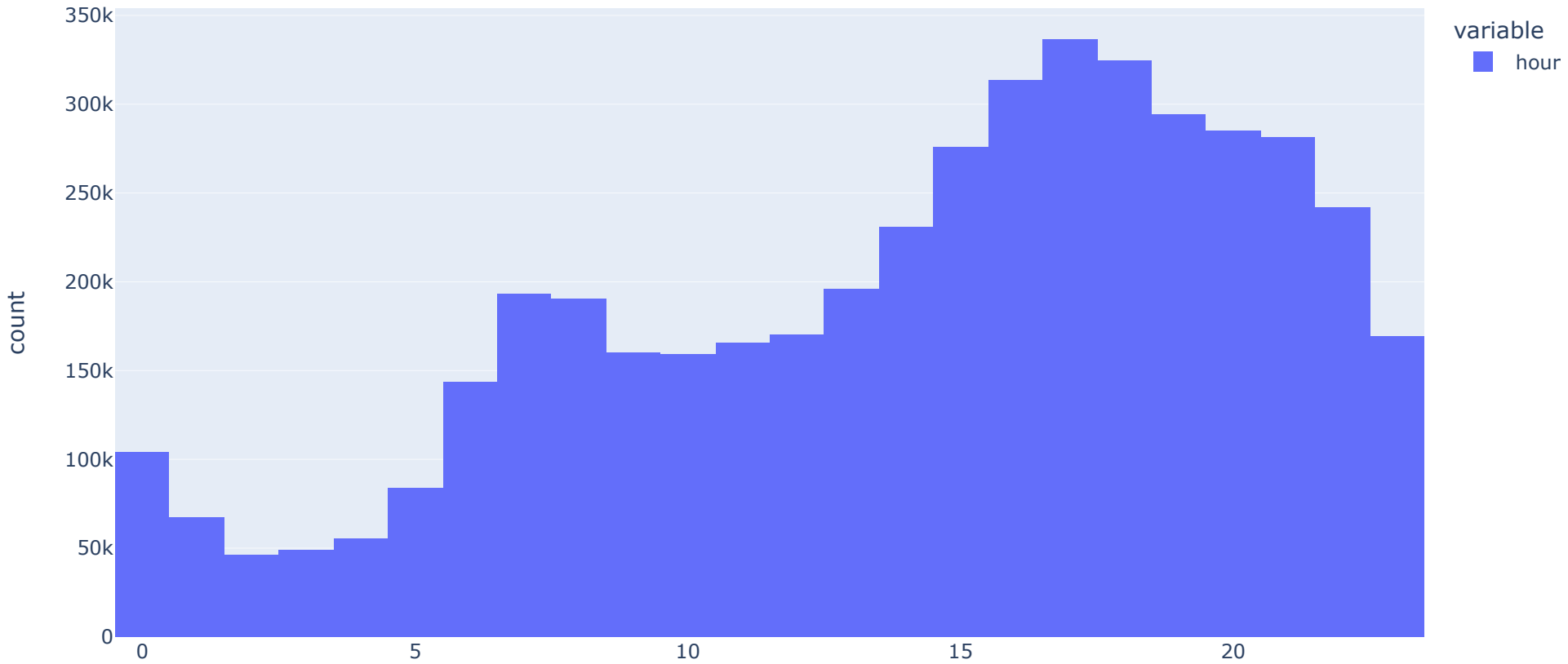


Conclusion for Question-1

The Most Pickups happened most during Thursdays and after Thursdays, Fridays closely follow. Even more interesting is the fact that more pickups occurred during Tuesdays and Wednesdays than on Saturdays or Sundays. **Uber being used as a means to get to work during the week could be a possible explanation** but, unfortunately, the purpose of these pickups is not available in the data.

Q.2. Find day wise busy hours for uber and why?

In [20]:
```
px.histogram(df['hour'])
```



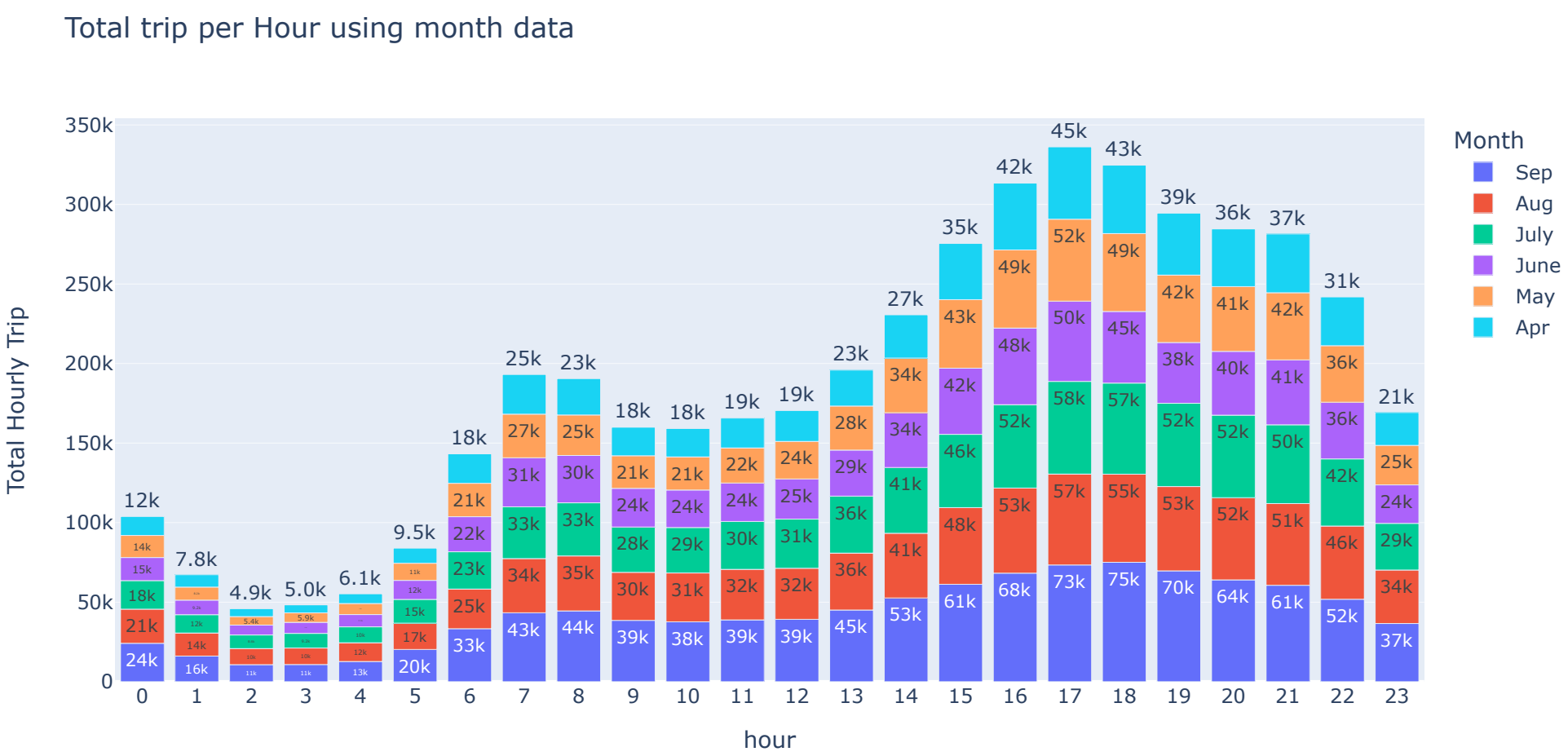
In [21]:
```
hour_pickup = df[['hour','Month']].groupby(['hour']).value_counts().reset_index()
hour_pickup.columns = ['hour','Month','Total Hourly Trip']
hour_pickup
```

Out[21]:

	hour	Month	Total Hourly Trip
0	0	Sep	24133
1	0	Aug	21451
2	0	July	17953
3	0	June	14514
4	0	May	13875
...	...	...	...
139	23	Aug	33609
140	23	July	29346
141	23	May	24836
142	23	June	24182
143	23	Apr	20649

144 rows × 3 columns

In [22]:
```
fig = px.bar(hour_pickup, x='hour',y='Total Hourly Trip',text_auto='.2s',
             title="Total trip per Hour using month data",color='Month')
fig.update_traces(textfont_size=12, textangle=0, textposition="outside", cliponaxis=False)
fig.update_layout(xaxis = dict(tickmode = 'linear',tick0 = 0,dtick = 1))
fig.show()
```



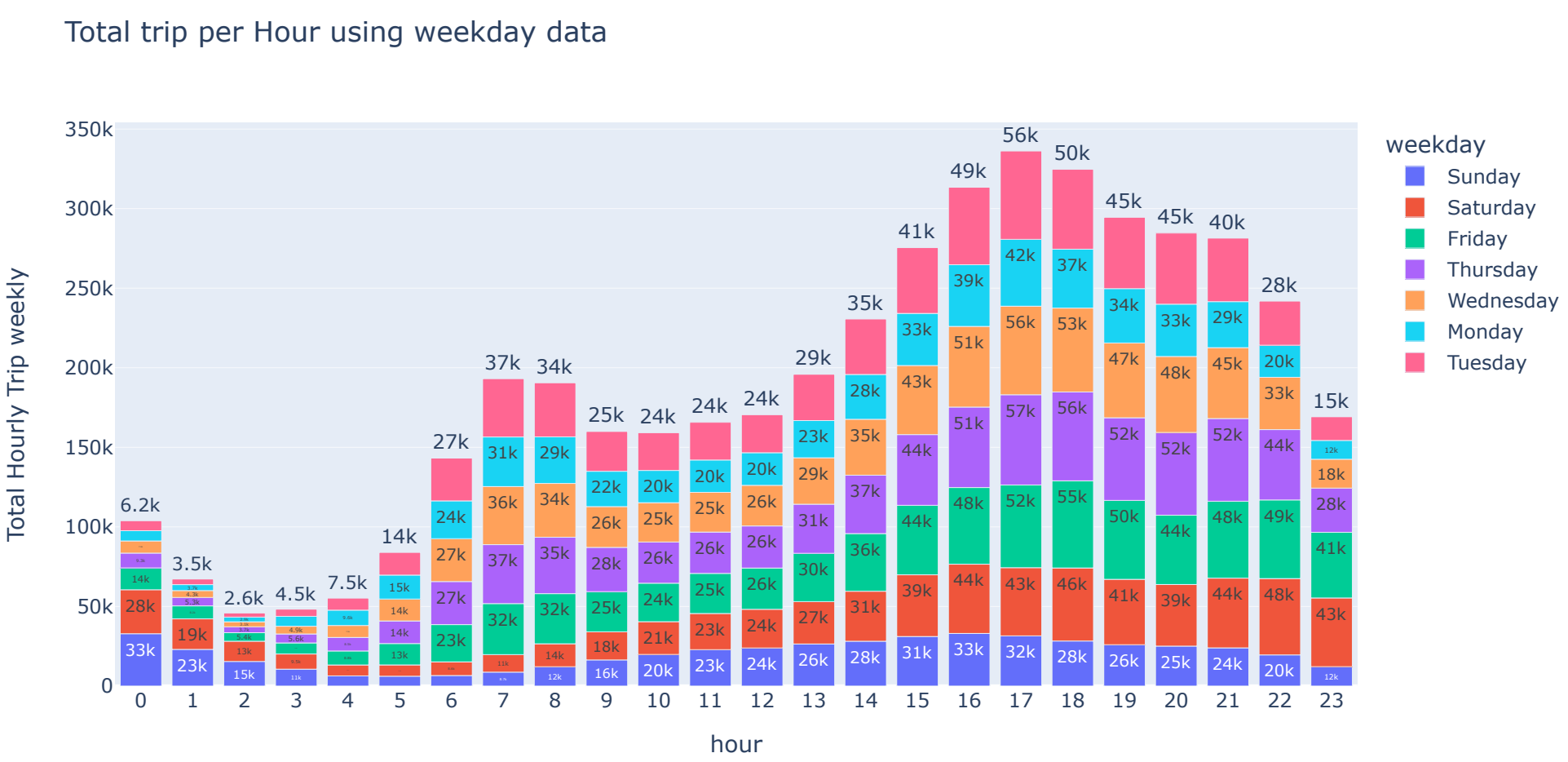
In [23]:
```
hour_pickup_weekly = df[['hour','weekday']].groupby(['hour']).value_counts().reset_index()
hour_pickup_weekly.columns = ['hour','weekday','Total Hourly Trip weekly']
hour_pickup_weekly
```

Out[23]:

	hour	weekday	Total Hourly Trip weekly
0	0	Sunday	32877
1	0	Saturday	27633
2	0	Friday	13716
3	0	Thursday	9293
4	0	Wednesday	7644
...	...	...	...
163	23	Thursday	27764
164	23	Wednesday	18146
165	23	Tuesday	14869
166	23	Sunday	12166
167	23	Monday	11811

168 rows × 3 columns

In [24]:
```
fig = px.bar(hour_pickup_weekly, x='hour',y='Total Hourly Trip weekly',text_auto='.2s',
             title="Total trip per Hour using weekday data",color='weekday')
fig.update_traces(textfont_size=12, textangle=0, textposition="outside", cliponaxis=False)
fig.update_layout(xaxis = dict(tickmode = 'linear',tick0 = 0,dtick = 1))
fig.show()
```

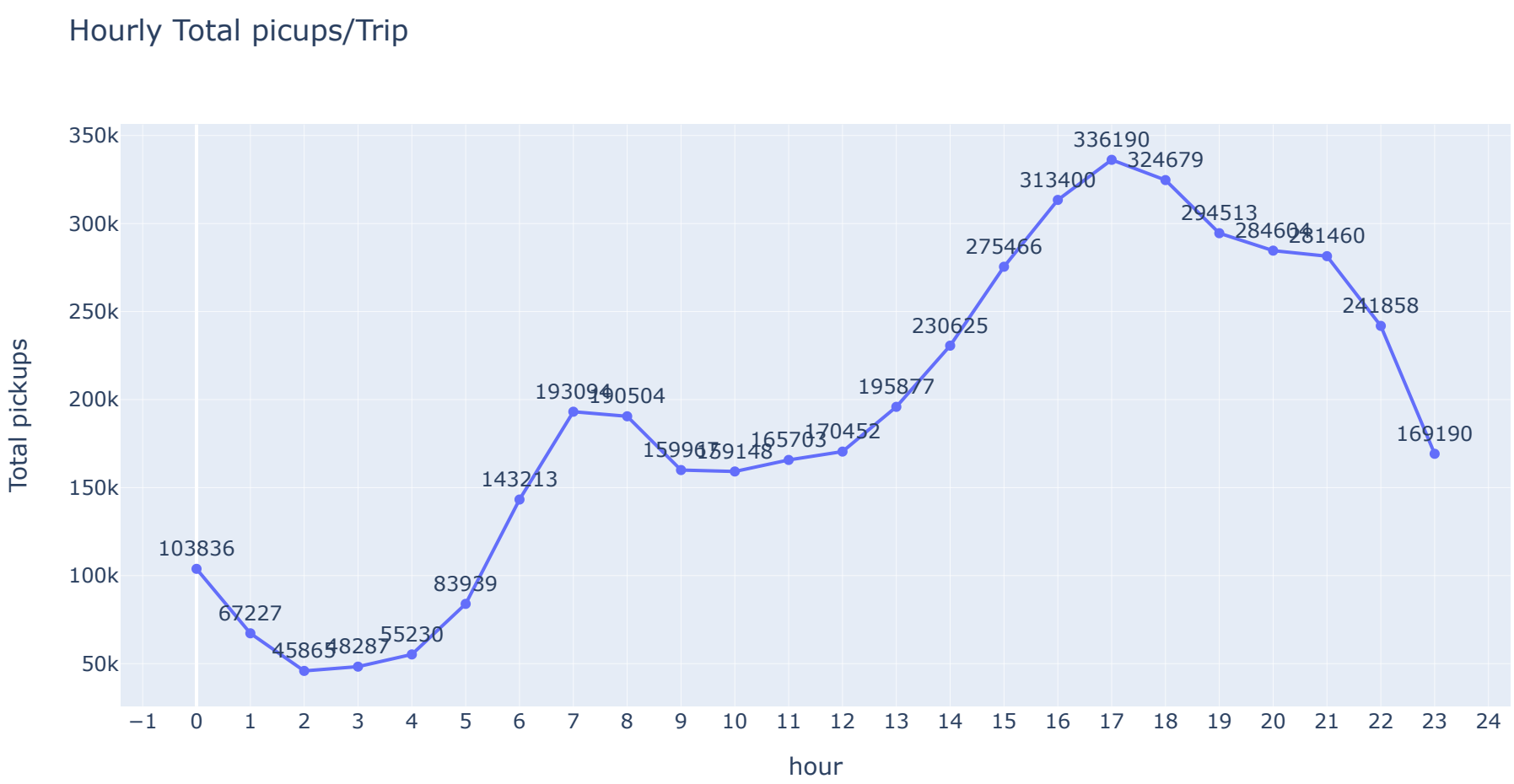


In [25]:
```
hourly_pickups = df[['hour']].groupby(['hour']).value_counts().reset_index()
hourly_pickups.columns = ['hour','Total pickups']
hourly_pickups.head()
```

Out[25]:

	hour	Total pickups
0	0	103836
1	1	67227
2	2	45865
3	3	48287
4	4	55230

In [26]:
```
fig = px.line(hourly_pickups, x = 'hour', y ='Total pickups', title="Hourly Total picups/Trip",
              markers=True,text='Total pickups')
fig.update_layout(xaxis = dict(tickmode = 'linear',tick0 = 0,dtick = 1))
fig.update_traces(textposition = "top center")
fig.show()
```



Conclusion for Question-2

1) The **highest number of trips by hour is 336190 trip, that corresponds to the peak hour 17:00.** Also from the plot, we can observe that **between 12–4am, there is a gradual drop in pickups**, then a steep increase between 4–8am before it starts to drop steadily and **flattens between 9am-12pm.** It then steadily rises after that to reach its **peak at most days is between 4–8pm** then it decreases steadily again throughout the night to the next morning, apart from **on Fridays and saturdays night between 8–11pm when there is a slight increase.**

2) **We can say that the majority of Uber's clients are workers.**

Q.3. How many trips were completed or canceled? Why do you think that % trip was canceled?

In [27]:
```
df.columns
```

Out[27]:
```
Index(['Date/Time', 'Lat', 'Lon', 'Base', 'Month', 'weekday', 'day', 'hour',
       'minute'],
      dtype='object')
```

Conclusion for Question-3

As per the above details of the column names, there is no information regarding the status of a journey or trip which was completed or canceled. There is also no other relevant information available. So, here we are not able to find out how many trips were completed or cancelled from the above dataset information.