In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
sns.set()
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.
23.0 is required for this version of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

This is taxi fare data where we are analysing the highest trip days finding out the outliers

Type *Markdown* and LaTeX: α^2

In [5]:
```
df = pd.read_csv('/kaggle/input/nyc-taxi-trip-data-google-public-data/taxi_trip_data.csv')
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000000 entries, 0 to 9999999
Data columns (total 17 columns):
 #   Column               Dtype  
---  ------               -----  
 0   vendor_id            int64  
 1   pickup_datetime      object 
 2   dropoff_datetime     object 
 3   passenger_count      int64  
 4   trip_distance        float64
 5   rate_code            int64  
 6   store_and_fwd_flag   object 
 7   payment_type         int64  
 8   fare_amount          float64
 9   extra                float64
 10  mta_tax              float64
 11  tip_amount           float64
 12  tolls_amount         float64
 13  imp_surcharge        float64
 14  total_amount         float64
 15  pickup_location_id   int64  
 16  dropoff_location_id  int64  
dtypes: float64(8), int64(6), object(3)
memory usage: 1.3+ GB
```

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

Out[7]:

	vendor_id	passenger_count	trip_distance	rate_code	payment_type	fare_amount	extra	mta_tax	tip_a
count	1.000000e+07	1.000000e+07	1.000000e+07	1.000000e+07	1.000000e+07	1.000000e+07	1.000000e+07	1.000000e+07	1.0000
mean	1.614328e+00	1.602949e+00	8.849280e+00	1.201239e+00	1.189299e+00	3.165255e+01	3.383781e-01	4.819289e-01	5.5985
std	5.146576e-01	1.245782e+00	5.882028e+00	1.250733e+00	4.339876e-01	1.606011e+02	5.512911e-01	1.207282e-01	4.8405
min	1.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00	1.000000e+00	-8.000000e+02	-8.000000e+01	-5.000000e-01	-3.2242
25%	1.000000e+00	1.000000e+00	5.820000e+00	1.000000e+00	1.000000e+00	2.350000e+01	0.000000e+00	5.000000e-01	2.0000
50%	2.000000e+00	1.000000e+00	8.480000e+00	1.000000e+00	1.000000e+00	2.850000e+01	0.000000e+00	5.000000e-01	5.5600
75%	2.000000e+00	2.000000e+00	1.110000e+01	1.000000e+00	1.000000e+00	3.700000e+01	5.000000e-01	5.000000e-01	7.9600
max	4.000000e+00	9.000000e+00	7.655760e+03	9.900000e+01	5.000000e+00	3.984600e+05	8.400000e+01	1.500000e+02	4.9600

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
vendor_id              0
pickup_datetime        0
dropoff_datetime       0
passenger_count        0
trip_distance          0
rate_code              0
store_and_fwd_flag     0
payment_type           0
fare_amount            0
extra                  0
mta_tax                0
tip_amount             0
tolls_amount           0
imp_surcharge          0
total_amount           0
pickup_location_id     0
dropoff_location_id    0
dtype: int64
```

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

Out[9]:

	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code	store_and_fwd_flag	payment_type	fare_am
0	2	2018-03-29 13:37:13	2018-03-29 14:17:01	1	18.15	3	N	1	
1	2	2018-03-29 13:37:18	2018-03-29 14:15:33	1	4.59	1	N	1	
2	2	2018-03-29 13:26:57	2018-03-29 13:28:03	1	0.30	1	N	1	
3	2	2018-03-29 13:07:48	2018-03-29 14:03:05	2	16.97	1	N	1	
4	2	2018-03-29 14:19:11	2018-03-29 15:19:59	5	14.45	1	N	1	

In [10]:
```
df.columns
```

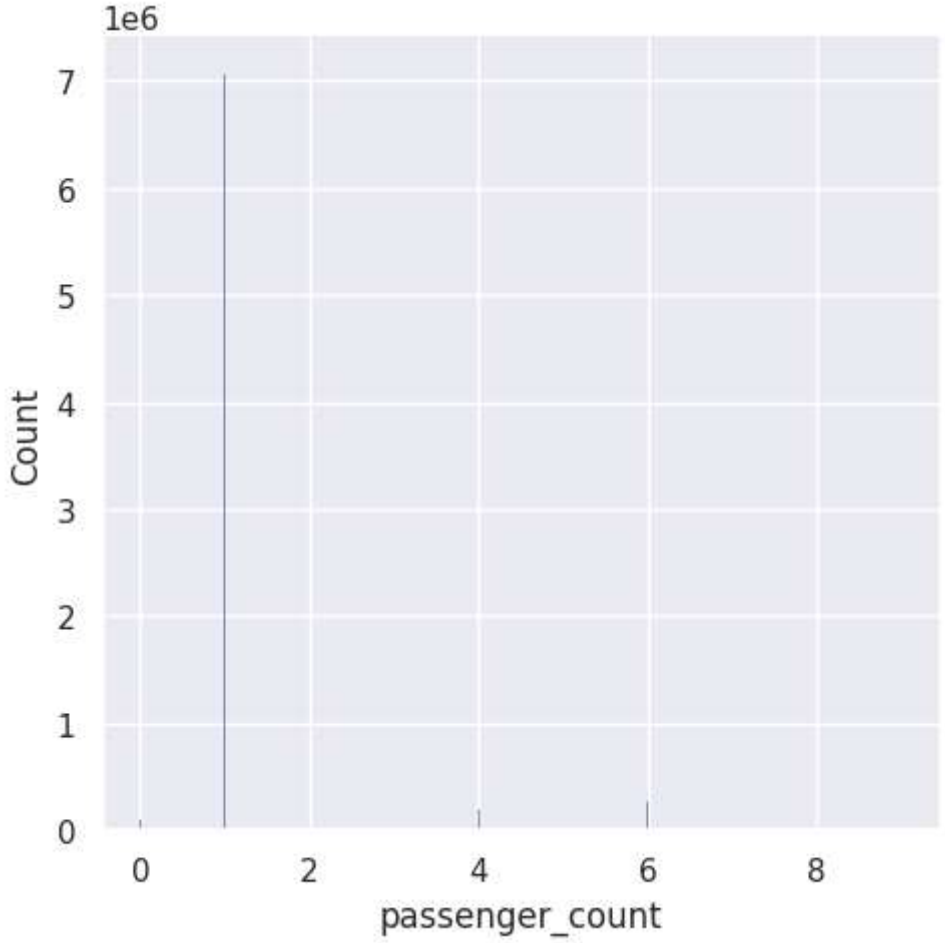
Out[10]:
```
Index(['vendor_id', 'pickup_datetime', 'dropoff_datetime', 'passenger_count',
       'trip_distance', 'rate_code', 'store_and_fwd_flag', 'payment_type',
       'fare_amount', 'extra', 'mta_tax', 'tip_amount', 'tolls_amount',
       'imp_surcharge', 'total_amount', 'pickup_location_id',
       'dropoff_location_id'],
      dtype='object')
```

In [11]:
```
df['pickup_datetime']=pd.to_datetime(df['pickup_datetime'])
df['dropoff_datetime']=pd.to_datetime(df['dropoff_datetime'])
```

We will now create a displot which is used to show univariant set of collected data show passenger count data distribution as we want to show one variable against another variable.

In [13]:
```
sns.displot(df['passenger_count'])
```

Out[13]:
```
<seaborn.axisgrid.FacetGrid at 0x7de8a7a95b10>
```



We will now convert the dates into days of the week to see on which day the most and least trips are taken

In [20]:
```
df['pickup_day']=df['pickup_datetime'].dt.day_name()
df['dropoff_day']=df['dropoff_datetime'].dt.day_name()
```

In [21]:
```
df['pickup_day'].value_counts()
```

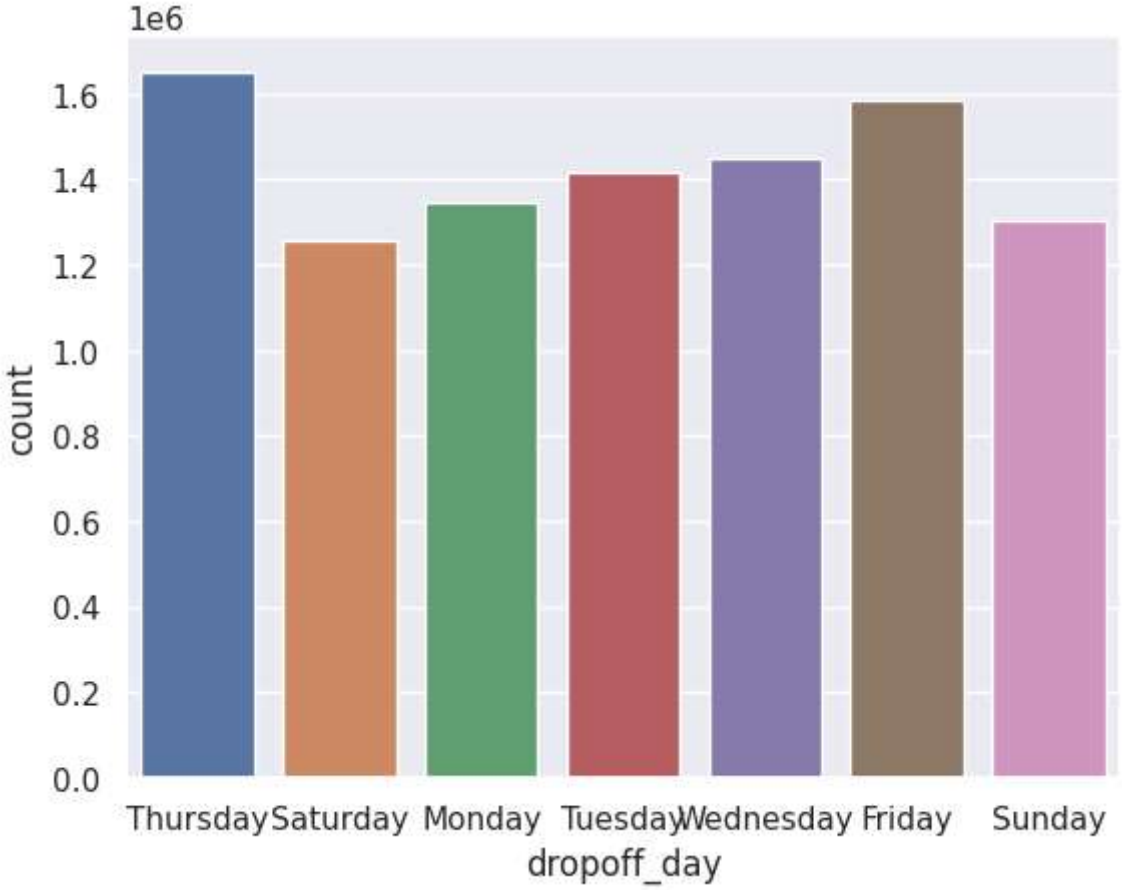
Out[21]:
```
Thursday     1664099
Friday       1588090
Wednesday    1451808
Tuesday      1422420
Monday       1343747
Sunday       1284224
Saturday     1245612
Name: pickup_day, dtype: int64
```

In [22]:
```
df['dropoff_day'].value_counts()
```

Out[22]:
```
Thursday     1650531
Friday       1584692
Wednesday    1447014
Tuesday      1418264
Monday       1343468
Sunday       1301192
Saturday     1254839
Name: dropoff_day, dtype: int64
```

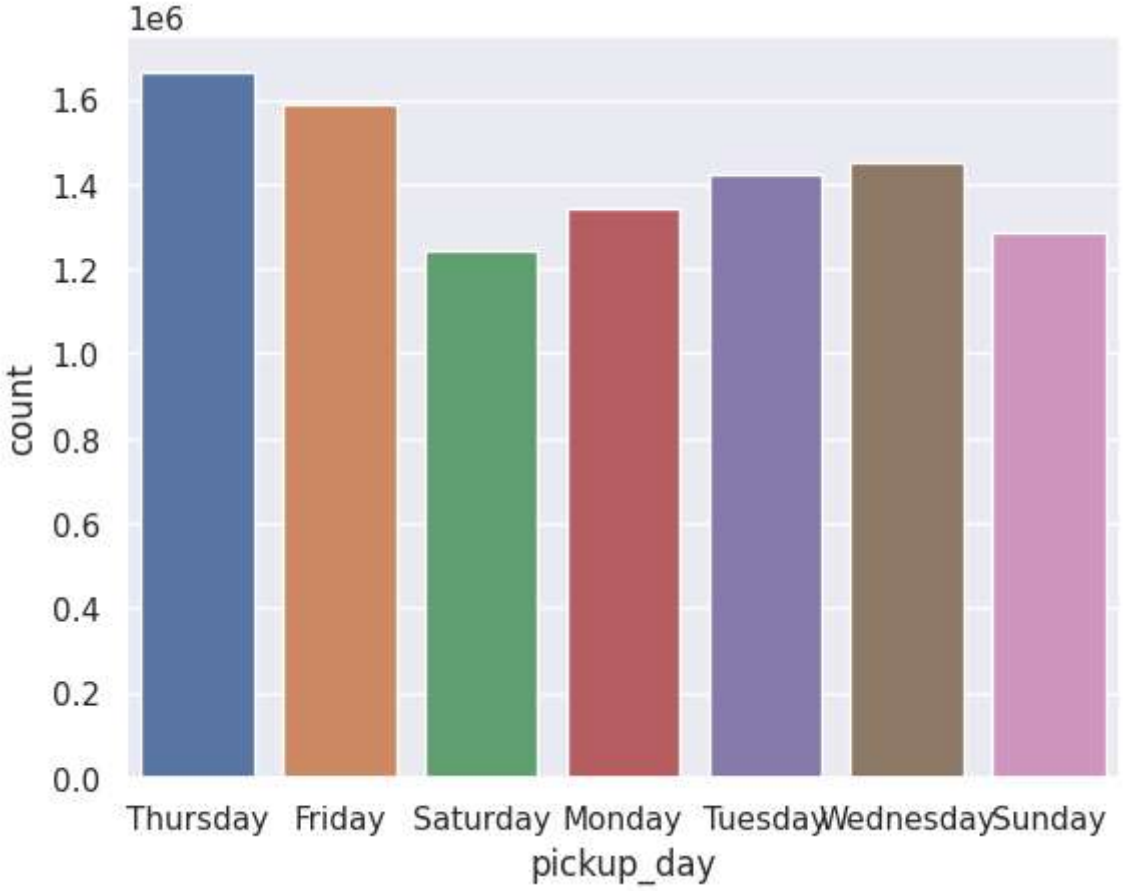
In [24]:
```
sns.countplot(x='dropoff_day', data=df)
```

Out[24]: <AxesSubplot: xlabel='dropoff_day', ylabel='count'>



In [25]:
```
sns.countplot(x='pickup_day', data=df)
```

Out[25]:
```
<AxesSubplot: xlabel='pickup_day', ylabel='count'>
```



We found out that maximum trips were taken on Thursday and least were taken on Saturday

In [29]:
```
df['pickup_time']=df['pickup_datetime'].dt.time
```

In [30]:
```
df['dropoff_time']=df['dropoff_datetime'].dt.time
```

In [32]:
```
df.dtypes
```

Out[32]:
```
vendor_id                     int64
pickup_datetime        datetime64[ns]
dropoff_datetime       datetime64[ns]
passenger_count                int64
trip_distance                float64
rate_code                      int64
store_and_fwd_flag            object
payment_type                   int64
fare_amount                  float64
extra                        float64
mta_tax                      float64
tip_amount                   float64
tolls_amount                 float64
imp_surcharge                float64
total_amount                 float64
pickup_location_id             int64
dropoff_location_id            int64
pickup_day                    object
dropoff_day                   object
pickup_time                   object
dropoff_time                  object
dtype: object
```

In [35]:
```
def timezone(x):
    if x>=datetime.time(4, 0, 1) and x <=datetime.time(10, 0, 0):
        return 'morning'
    elif x>=datetime.time(10, 0, 1) and x <=datetime.time(16, 0, 0):
        return 'midday'
    elif x>=datetime.time(16, 0, 1) and x <=datetime.time(22, 0, 0):
        return 'evening'
    elif x>=datetime.time(22, 0, 1) or x <=datetime.time(4, 0, 0):
        return 'late night'

df['pickup_timezone']=df['pickup_datetime'].apply(lambda x :timezone(datetime.datetime.strptime(str(x), "%Y-%
df['dropoff_timezone']=df['dropoff_datetime'].apply(lambda x :timezone(datetime.datetime.strptime(str(x), "%Y
```

In [36]:
```
df.head()
```

Out[36]:

	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code	store_and_fwd_flag	payment_type	fare_am
0	2	2018-03-29 13:37:13	2018-03-29 14:17:01	1	18.15	3	N	1	
1	2	2018-03-29 13:37:18	2018-03-29 14:15:33	1	4.59	1	N	1	
2	2	2018-03-29 13:26:57	2018-03-29 13:28:03	1	0.30	1	N	1	
3	2	2018-03-29 13:07:48	2018-03-29 14:03:05	2	16.97	1	N	1	
4	2	2018-03-29 14:19:11	2018-03-29 15:19:59	5	14.45	1	N	1	

5 rows × 23 columns

In [39]:
```
df['duration']=df['dropoff_datetime'] - df['pickup_datetime']
```

In [42]:
```
df.drop(['trip_duration'], axis=1)
```

Out[42]:

	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code	store_and_fwd_flag	payment_type	f
0	2	2018-03-29 13:37:13	2018-03-29 14:17:01	1	18.15	3	N	1	
1	2	2018-03-29 13:37:18	2018-03-29 14:15:33	1	4.59	1	N	1	
2	2	2018-03-29 13:26:57	2018-03-29 13:28:03	1	0.30	1	N	1	
3	2	2018-03-29 13:07:48	2018-03-29 14:03:05	2	16.97	1	N	1	
4	2	2018-03-29 14:19:11	2018-03-29 15:19:59	5	14.45	1	N	1	
...	...	...	...	...	...	...	...	...	
9999995	2	2018-03-29 12:16:01	2018-03-29 13:03:31	1	3.45	1	N	1	
9999996	1	2018-03-29 12:26:25	2018-03-29 13:09:54	1	6.80	1	N	1	
9999997	2	2018-03-29 12:22:12	2018-03-29 13:01:07	2	8.46	1	N	1	
9999998	2	2018-03-29 13:57:40	2018-03-29 15:07:42	1	14.43	1	N	1	
9999999	2	2018-03-29 13:29:40	2018-03-29 14:02:51	1	4.42	1	N	1	

10000000 rows × 24 columns

In [43]:
```
df.head()
```

Out[43]:

	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code	store_and_fwd_flag	payment_type	fare_am
0	2	2018-03-29 13:37:13	2018-03-29 14:17:01	1	18.15	3	N	1	
1	2	2018-03-29 13:37:18	2018-03-29 14:15:33	1	4.59	1	N	1	
2	2	2018-03-29 13:26:57	2018-03-29 13:28:03	1	0.30	1	N	1	
3	2	2018-03-29 13:07:48	2018-03-29 14:03:05	2	16.97	1	N	1	
4	2	2018-03-29 14:19:11	2018-03-29 15:19:59	5	14.45	1	N	1	

5 rows × 25 columns

In [51]:
```
df.drop(["duration"], axis = 1, inplace = True)
```

In [52]:
```
df.head()
```

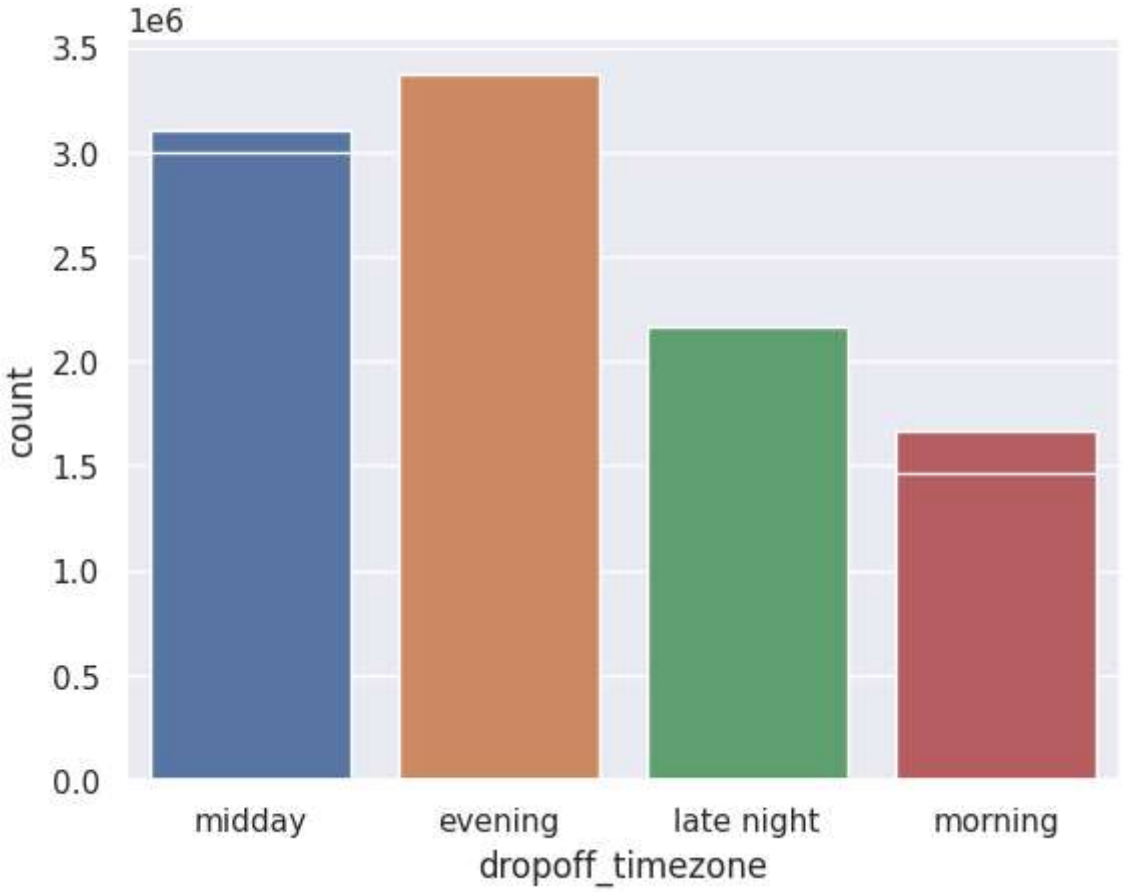
Out[52]:

	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code	store_and_fwd_flag	payment_type	fare_am
0	2	2018-03-29 13:37:13	2018-03-29 14:17:01	1	18.15	3	N	1	
1	2	2018-03-29 13:37:18	2018-03-29 14:15:33	1	4.59	1	N	1	
2	2	2018-03-29 13:26:57	2018-03-29 13:28:03	1	0.30	1	N	1	
3	2	2018-03-29 13:07:48	2018-03-29 14:03:05	2	16.97	1	N	1	
4	2	2018-03-29 14:19:11	2018-03-29 15:19:59	5	14.45	1	N	1	

5 rows × 24 columns

In [56]:
```
sns.countplot(x='pickup_timezone', data=df)
sns.countplot(x='dropoff_timezone', data=df)
```

Out[56]: <AxesSubplot: xlabel='dropoff_timezone', ylabel='count'>



In [57]:
```
df.dtypes
```

Out[57]:
```
vendor_id                      int64
pickup_datetime        datetime64[ns]
dropoff_datetime       datetime64[ns]
passenger_count                int64
trip_distance                float64
rate_code                      int64
store_and_fwd_flag            object
payment_type                   int64
fare_amount                  float64
extra                        float64
mta_tax                      float64
tip_amount                   float64
tolls_amount                 float64
imp_surcharge                float64
total_amount                 float64
pickup_location_id             int64
dropoff_location_id            int64
pickup_day                    object
dropoff_day                   object
pickup_time                   object
dropoff_time                  object
pickup_timezone               object
dropoff_timezone              object
trip_duration         timedelta64[ns]
dtype: object
```

In [60]:
```
df['trip_duration_hour']=df['trip_duration']/3600
```

In [62]:
```
df['triphours'] = df['trip_duration_hour'] / pd.Timedelta(hours=1)
```