Customer-Service-Request-Analysis

In [98]:
```
!python --version
```
```
Python 3.10.5
```

In [7]:
```
import numpy as np
import pandas as pd
```

Import a 311 NYC service request.

In [8]:
```
data =pd.read_csv('311_Service_Requests_from_2010_to_Present.csv')
# To Read Top 5 records
data.head()
```
```
C:\Users\abman\AppData\Local\Temp\ipykernel_11320\1797725720.py:1: DtypeWarning: Columns (48,49) have mixed typ
es. Specify dtype option on import or set low_memory=False.
  data =pd.read_csv('311_Service_Requests_from_2010_to_Present.csv')
```

Out[8]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction
0	32310363	12/31/2015 11:59:45 PM	01-01-16 0:55	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN	NaN
1	32309934	12/31/2015 11:59:44 PM	01-01-16 1:26	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN	NaN
2	32309159	12/31/2015 11:59:29 PM	01-01-16 4:51	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN	NaN
3	32305098	12/31/2015 11:57:46 PM	01-01-16 7:43	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN	NaN
4	32306529	12/31/2015 11:56:58 PM	01-01-16 3:24	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN	NaN

5 rows × 53 columns

In [9]:
```
## check columns names
data.columns
```
Out[9]:
```
Index(['Unique Key', 'Created Date', 'Closed Date', 'Agency', 'Agency Name',
       'Complaint Type', 'Descriptor', 'Location Type', 'Incident Zip',
       'Incident Address', 'Street Name', 'Cross Street 1', 'Cross Street 2',
       'Intersection Street 1', 'Intersection Street 2', 'Address Type',
       'City', 'Landmark', 'Facility Type', 'Status', 'Due Date',
       'Resolution Description', 'Resolution Action Updated Date',
       'Community Board', 'Borough', 'X Coordinate (State Plane)',
       'Y Coordinate (State Plane)', 'Park Facility Name', 'Park Borough',
       'School Name', 'School Number', 'School Region', 'School Code',
       'School Phone Number', 'School Address', 'School City', 'School State',
       'School Zip', 'School Not Found', 'School or Citywide Complaint',
       'Vehicle Type', 'Taxi Company Borough', 'Taxi Pick Up Location',
       'Bridge Highway Name', 'Bridge Highway Direction', 'Road Ramp',
       'Bridge Highway Segment', 'Garage Lot Name', 'Ferry Direction',
       'Ferry Terminal Name', 'Latitude', 'Longitude', 'Location'],
      dtype='object')
```

In [10]:
```
# Check shape of DataFrame
data.shape
```
Out[10]:
```
(300698, 53)
```

In [11]:
```
#To get nan values
data.isnull().sum()
```
Out[11]:
```
Unique Key                             0
Created Date                           0
Closed Date                         2164
Agency                                 0
Agency Name                            0
Complaint Type                         0
Descriptor                          5914
Location Type                        131
Incident Zip                        2615
Incident Address                   44410
Street Name                        44410
Cross Street 1                     49279
Cross Street 2                     49779
Intersection Street 1             256840
Intersection Street 2             257336
Address Type                        2815
City                                2614
Landmark                          300349
Facility Type                       2171
Status                                 0
Due Date                               3
Resolution Description                 0
Resolution Action Updated Date      2187
Community Board                        0
Borough                                0
X Coordinate (State Plane)          3540
Y Coordinate (State Plane)          3540
Park Facility Name                     0
Park Borough                           0
School Name                            0
School Number                          0
School Region                          1
School Code                            1
School Phone Number                    0
School Address                         0
School City                            0
School State                           0
School Zip                             1
School Not Found                       0
School or Citywide Complaint      300698
Vehicle Type                      300698
Taxi Company Borough              300698
Taxi Pick Up Location             300698
Bridge Highway Name               300455
Bridge Highway Direction          300455
Road Ramp                         300485
Bridge Highway Segment            300485
Garage Lot Name                   300698
Ferry Direction                   300697
Ferry Terminal Name               300696
Latitude                            3540
Longitude                           3540
Location                            3540
dtype: int64
```

In [12]:
```
# As we seen Closed Date is important column and have many missing values
data[data['Closed Date'].isnull()]
```
Out[12]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name
416	32305700	12/31/2015 02:16:04 PM	NaN	NYPD	New York City Police Department	Illegal Parking	Posted Parking Sign Violation	Street/Sidewalk	NaN	5426-5526 90TH ST	...	NaN
611	32309308	12/31/2015 09:58:06 AM	NaN	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	NaN	30 STREET	...	NaN
1648	32303348	12/30/2015 05:13:42 AM	NaN	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	NaN	21600-2169 91ST AVE	...	NaN
1816	32294519	12/29/2015 10:44:50 AM	NaN	NYPD	New York City Police Department	Derelict Vehicle	With License Plate	Street/Sidewalk	NaN	127 STREET	...	NaN
1965	32296487	12/29/2015 07:09:13 PM	NaN	NYPD	New York City Police Department	Derelict Vehicle	With License Plate	Street/Sidewalk	NaN	5201-5299 68TH ST	...	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
300273	30287350	03/29/2015 02:40:19 AM	NaN	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	NaN	3801-3999 23RD AVE	...	NaN
300492	30284963	03/29/2015 08:50:15 AM	NaN	NYPD	New York City Police Department	Vending	Unlicensed	Street/Sidewalk	NaN	COOPER AVE	...	NaN
300496	30285492	03/29/2015 08:44:13 AM	NaN	NYPD	New York City Police Department	Vending	Unlicensed	Street/Sidewalk	NaN	80 STREET	...	NaN
300620	30282717	03/29/2015 01:55:35 AM	NaN	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	NaN	CRESCENT AVENUE	...	NaN
300693	30281872	03/29/2015 12:33:41 AM	NaN	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	NaN	CRESCENT AVENUE	...	NaN

2164 rows × 53 columns

In [13]:
```
# We check data type of each column
data.dtypes
```
Out[13]:
```
Unique Key                          int64
Created Date                       object
Closed Date                        object
Agency                             object
Agency Name                        object
Complaint Type                     object
Descriptor                         object
Location Type                      object
Incident Zip                      float64
Incident Address                   object
Street Name                        object
Cross Street 1                     object
Cross Street 2                     object
Intersection Street 1              object
Intersection Street 2              object
Address Type                       object
City                               object
Landmark                           object
Facility Type                      object
Status                             object
Due Date                           object
Resolution Description             object
Resolution Action Updated Date     object
Community Board                    object
Borough                            object
X Coordinate (State Plane)        float64
Y Coordinate (State Plane)        float64
Park Facility Name                 object
Park Borough                       object
School Name                        object
School Number                      object
School Region                      object
School Code                        object
School Phone Number                object
School Address                     object
School City                        object
School State                       object
School Zip                         object
School Not Found                   object
School or Citywide Complaint      float64
Vehicle Type                      float64
Taxi Company Borough              float64
Taxi Pick Up Location             float64
Bridge Highway Name                object
Bridge Highway Direction           object
Road Ramp                          object
Bridge Highway Segment             object
Garage Lot Name                   float64
Ferry Direction                    object
Ferry Terminal Name                object
Latitude                          float64
Longitude                         float64
Location                           object
dtype: object
```

Read or convert the columns 'Created Date' and Closed Date' to datetime datatype and create a new column 'Request_Closing_Time' as the time elapsed between request creation and request closing.

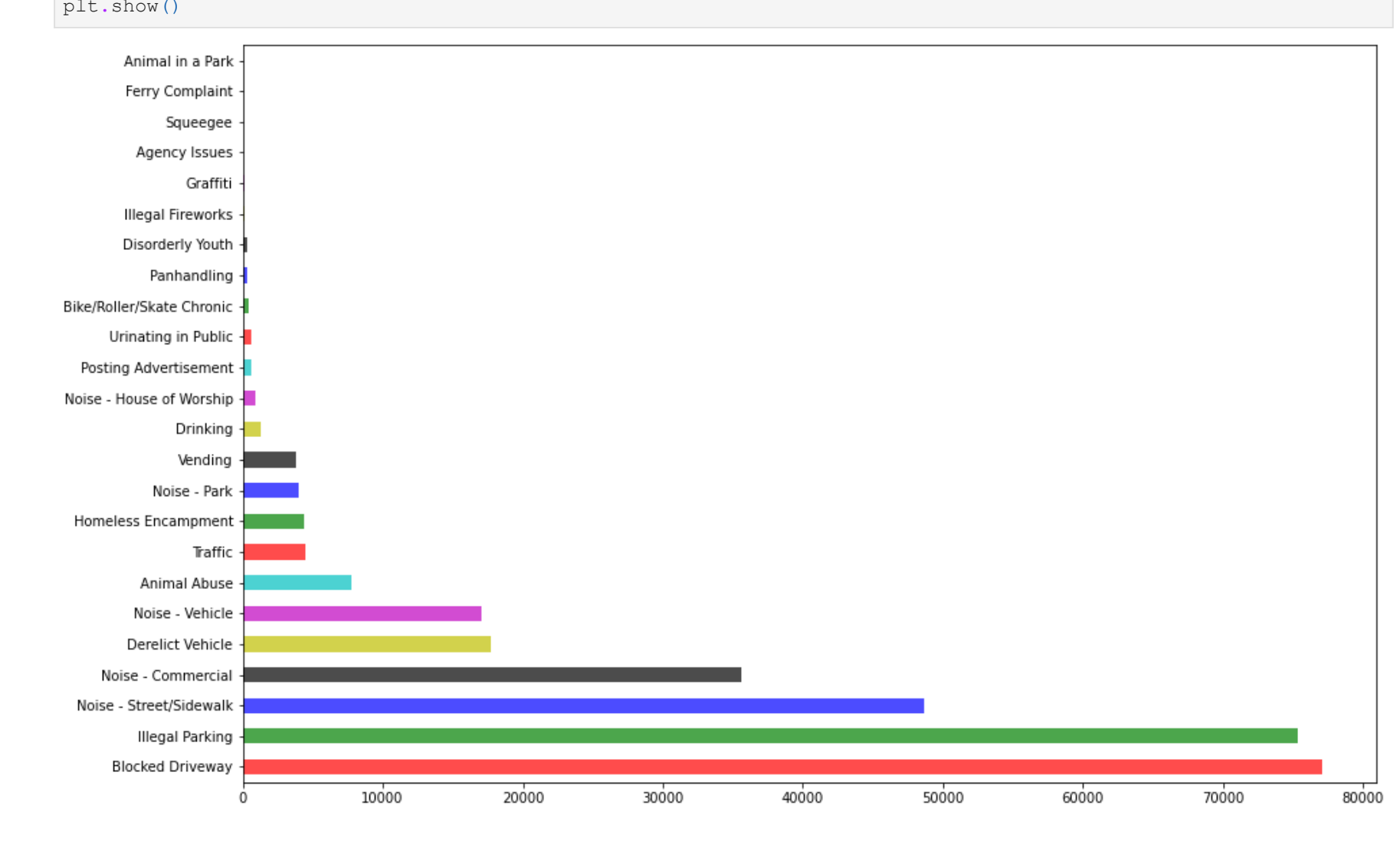
In [15]:
```
import datetime as dt
import time, datetime

# Convert "Closed Date" to datetime dtype
data['Closed Date'] = pd.to_datetime(data['Closed Date'])
data['Closed Date'].dtype
data['Created Date'] = pd.to_datetime(data['Created Date'])
data['Created Date'].dtype

# Create new column Request_Closing_Time with time taken to close complain
data['Request_Closing_Time'] = data['Closed Date'] - data['Created Date']

data['Request_Closing_Time'].head()
```
Out[15]:
```
0   0 days 00:55:15
1   0 days 01:26:16
2   0 days 04:51:31
3   0 days 07:45:14
4   0 days 03:27:02
Name: Request_Closing_Time, dtype: timedelta64[ns]
```

Provide major insights/patterns that you can offer in a visual format (graphs or tables); at least 4 major conclusions that you can come up with after generic data mining

Insight - 1

In [16]:
```
# Check count in each complain type - sorted decreasing order
data['Complaint Type'].value_counts()
```
Out[16]:
```
Blocked Driveway             77044
Illegal Parking              75361
Noise - Street/Sidewalk      48612
Noise - Commercial           35577
Derelict Vehicle             17718
Noise - Vehicle              17083
Animal Abuse                  7778
Traffic                       4498
Homeless Encampment           4416
Noise - Park                  4042
Vending                       3802
Drinking                      1280
Noise - House of Worship       931
Posting Advertisement          650
Urinating in Public            592
Bike/Roller/Skate Chronic      427
Panhandling                    307
Disorderly Youth               286
Illegal Fireworks              168
Graffiti                       113
Agency Issues                    6
Squeegee                         4
Ferry Complaint                  2
Animal in a Park                 1
Name: Complaint Type, dtype: int64
```

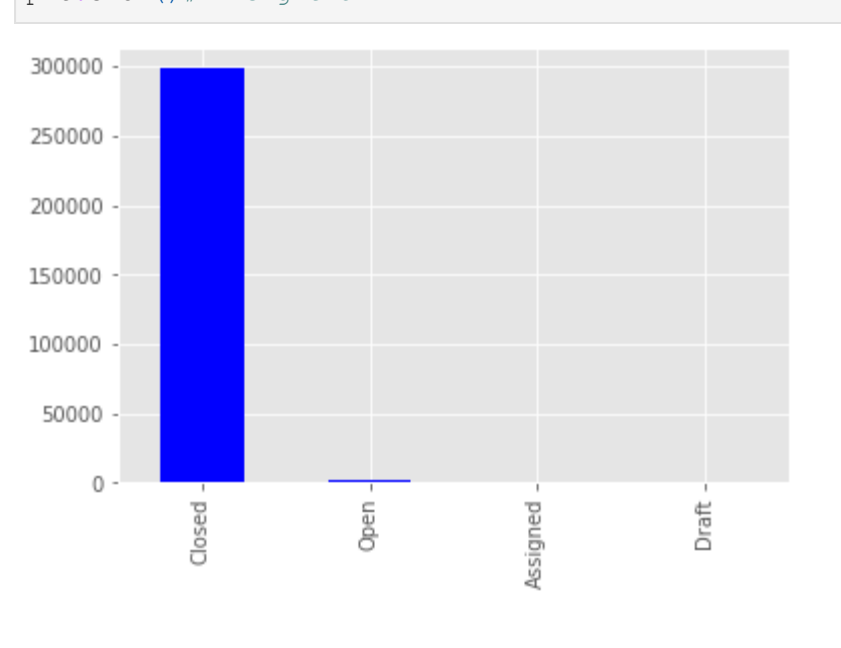
In [17]:
```
#Import required libraries
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Create Bar plot for complain type to check frequency in Complain Type
data['Complaint Type'].value_counts().plot(kind="barh", color=list('rgbkymc'), alpha=0.7, figsize=(15,10))
plt.show()
```



Insight 2

In [19]:
```
# Let's check count for status type
data['Status'].value_counts()
```
Out[19]:
```
Closed      298471
Open          1439
Assigned       786
Draft            2
Name: Status, dtype: int64
```

In [20]:
```
# Draw Bar lot for Status for color = ggbkymc
from matplotlib import style
style.use('ggplot')
data['Status'].value_counts().plot(kind='bar', color=list('b'))
plt.show()# Insight 3
```



Insight 3

In [22]:
```
## counts complaints type group by city and complaints Type
complaintTypecity = pd.DataFrame({'count':
                                  data.groupby(['Complaint Type','City']).size()}).reset_index()
complaintTypecity.head(2)
```
Out[22]:

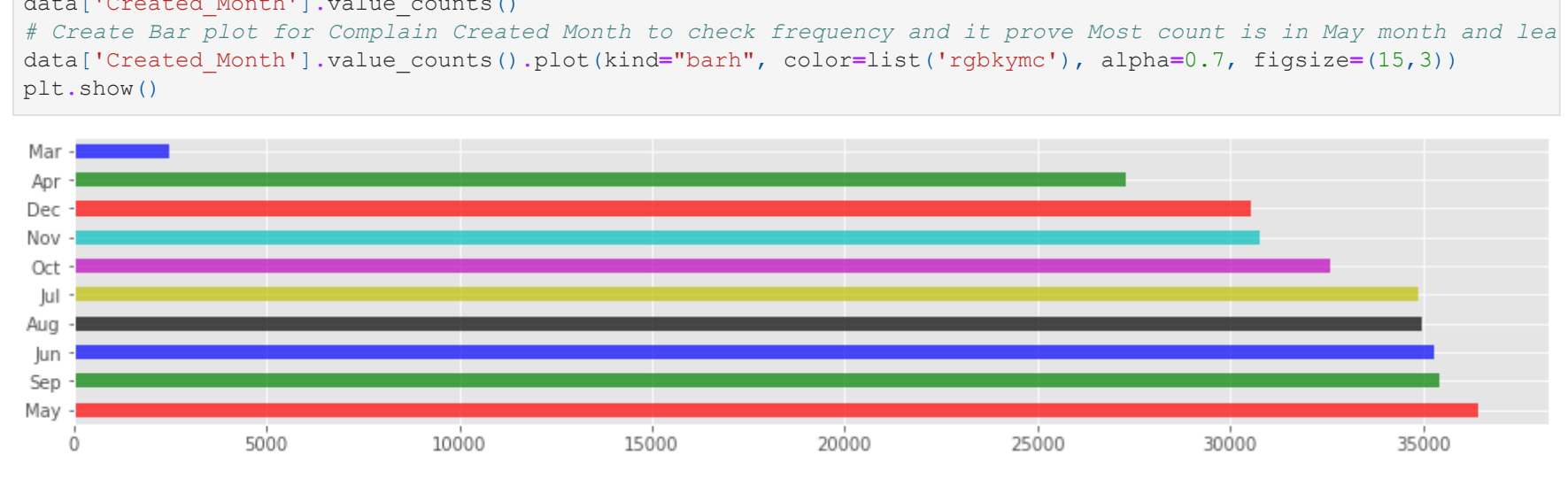
	Complaint Type	City	count
0	Animal Abuse	ARVERNE	38
1	Animal Abuse	ASTORIA	125

Insight 4

In [23]:
```
##  To check with Month have Complain creation most and least

# Created Series for months in text format
monthSeries = pd.Series({1: 'Jan', 2: 'Feb', 3: 'Mar', 4: 'Apr', 5: 'May', 6: 'Jun', 7: 'Jul', 8: 'Aug', 9: 'Se
print(monthSeries)
print(monthSeries[12])
```
```
1     Jan
2     Feb
3     Mar
4     Apr
5     May
6     Jun
7     Jul
8     Aug
9     Sep
10    Oct
11    Nov
12    Dec
dtype: object
Dec
```

In [24]:
```
import datetime as dt
import time, datetime
data['Created Date'].dtype

# Function to fetch month from Created Date column

def getMonth(cDate):
    a = str(cDate)
    datee = datetime.datetime.strptime(a, "%Y-%m-%d %H:%M:%S")
    return monthSeries[datee.month]

# Test function getMonth
print(data['Created Date'][0])
print(getMonth(data['Created Date'][0]))
```
```
2015-12-31 23:59:45
Dec
```

In [25]:
```
# Created new column Created_Month and kept all text format months in that column

data['Created_Month'] = data['Created Date'].apply(getMonth)
data['Created_Month'].value_counts()
# Create Bar plot for Complain Created Month to check frequency and it prove Most count is in May month and lea
data['Created_Month'].value_counts().plot(kind="barh", color=list('rgbkymc'), alpha=0.7, figsize=(15,3))
plt.show()
```



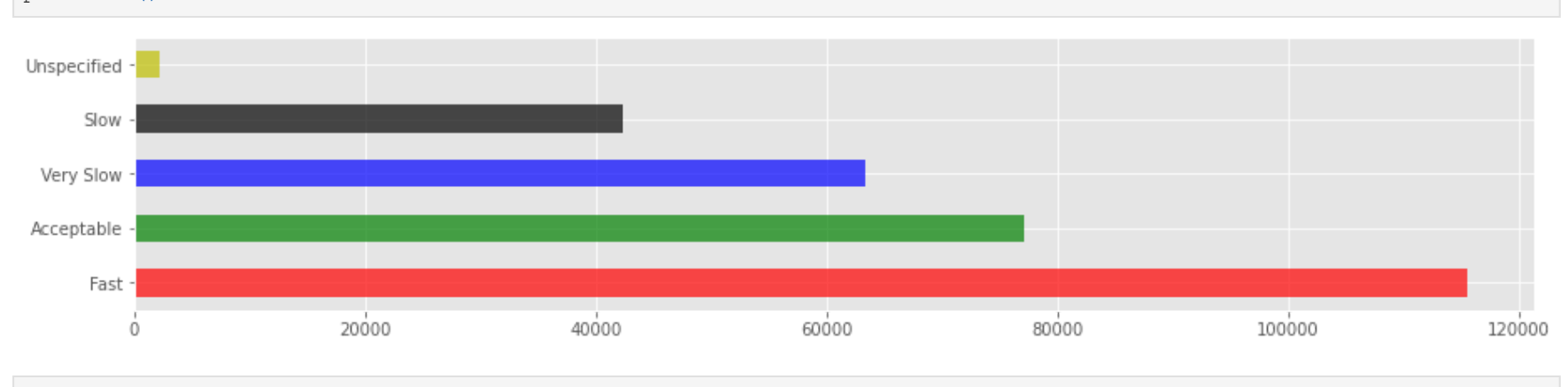
Insight 5

In [26]:
```
## Categorize Request_Closing_Time as follows -
# Below 2 hours - Fast, Between 2 to 4 hours - Acceptable, Between 4 to 6 - Slow, More than 6 hours - Very Slow
# For this, first will create new column Request_Closing_In_Hr and then create new column - Request_Closing_Tim

# Function to convert TimeDelta in Hour
def toHr(timeDel):
    days = timeDel.days
    hours = round(timeDel.seconds/3600, 2)
    result = (days * 24) + hours
    #print(days)
    #print(hours)
    return result
    #return round(pd.Timedelta(timeDel).seconds / 3600, 2)
    # Testing of function with days
```

In [27]:
```
# Apply this function to every row of column Request_Closing_Time
data['Request_Closing_In_Hr'] = data['Request_Closing_Time'].apply(toHr)

data['Request_Closing_In_Hr'].head()
```
Out[27]:
```
0    0.92
1    1.44
2    4.86
3    7.75
4    3.45
Name: Request_Closing_In_Hr, dtype: float64
```

In [28]:
```
import math
# Function to categorize hours - Less than 2 hours - Fast, Between 2 to 4 hours - Acceptable, Between 4 to 6 -
def hrToCategory(hr):
    if (math.isnan(hr)):
        return 'Unspecified'
    elif (hr < 2.0):
        return 'Fast'
    elif (4.0 > hr >= 2.0):
        return 'Acceptable'
    elif (6.0 > hr >= 4.0):
        return 'Slow'
    else:
        return 'Very Slow'

# Testing function
print(hrToCategory(1.99))

# Create new column Request_Closing_Time_Category and apply function on column Request_Closing_In_Hr

data['Request_Closing_Time_Category'] = data['Request_Closing_In_Hr'].apply(hrToCategory)

data['Request_Closing_Time_Category'].head()
```
Out[28]:
```
Fast
0          Fast
1          Fast
2          Slow
3     Very Slow
4    Acceptable
Name: Request_Closing_Time_Category, dtype: object
```

In [29]:
```
#Import required libraries
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
data['Request_Closing_Time_Category'].value_counts()

# Create Bar plot for Request_Closing_Time_Category to check frequency in Request_Closing_Time_Category and it
data['Request_Closing_Time_Category'].value_counts().plot(kind="barh", color=list('rgbkymc'), alpha=0.7, figsiz
plt.show()
```



In [30]:
```
data['Descriptor'].unique()
```
Out[30]:
```
array(['Loud Music/Party', 'No Access', 'Commercial Overnight Parking',
       'Blocked Sidewalk', 'Posted Parking Sign Violation',
       'Blocked Hydrant', 'With License Plate', 'Partial Access',
       'Unauthorized Bus Layover', 'Double Parked Blocking Vehicle',
       'Double Parked Blocking Traffic', 'Vehicle', 'Loud Talking',
       'Banging/Pounding', 'Car/Truck Music', 'Tortured',
       'In Prohibited Area', 'Congestion/Gridlock', 'Neglected',
       'Car/Truck Horn', 'In Public', 'Other (complaint details)', nan,
       'No Shelter', 'Truck Route Violation', 'Unlicensed',
       'Overnight Commercial Storage', 'Engine Idling',
       'After Hours - Licensed Est', 'Detached Trailer',
       'Underage - Licensed Est', 'Chronic Stoplight Violation',
       'Loud Television', 'Chained', 'Building', 'In Car',
       'Police Report Requested', 'Chronic Speeding',
       'Playing in Unsuitable Place', 'Drag Racing',
       'Police Report Not Requested', 'Nuisance/Truant', 'Homeless Issue',
       'Language Access Complaint', 'Disruptive Passenger',
       'Animal Waste'], dtype=object)
```

In [31]:
```
complaintTypecity = pd.DataFrame({'Total Number':
                                  data.groupby(['Complaint Type','Descriptor']).size()})
complaintTypecity
```
Out[31]:

Complaint Type	Descriptor	Total Number
Agency Issues	Language Access Complaint	6
Animal Abuse	Chained	535
	In Car	251
	Neglected	3787
	No Shelter	382
	Other (complaint details)	1969
	Tortured	854
Animal in a Park	Animal Waste	1
Blocked Driveway	No Access	56976
	Partial Access	20068
Derelict Vehicle	With License Plate	17718
Disorderly Youth	Nuisance/Truant	41
	Playing in Unsuitable Place	245
Drinking	After Hours - Licensed Est	77
	In Public	932
	Underage - Licensed Est	271
Ferry Complaint	Disruptive Passenger	1
	Homeless Issue	1
Graffiti	Police Report Not Requested	23
	Police Report Requested	90
Illegal Parking	Blocked Hydrant	16081
	Blocked Sidewalk	11121
	Commercial Overnight Parking	12189
	Detached Trailer	464
	Double Parked Blocking Traffic	5731
	Double Parked Blocking Vehicle	4211
	Overnight Commercial Storage	1757
	Posted Parking Sign Violation	22440
	Unauthorized Bus Layover	1367
Noise - Commercial	Banging/Pounding	3917
	Car/Truck Horn	991
	Car/Truck Music	899
	Loud Music/Party	25787
	Loud Talking	3891
	Loud Television	92
Noise - House of Worship	Banging/Pounding	248
	Loud Music/Party	504
	Loud Talking	178
	Loud Television	1
Noise - Park	Loud Music/Party	3109
	Loud Talking	933
Noise - Street/Sidewalk	Loud Music/Party	32030
	Loud Talking	16582
Noise - Vehicle	Car/Truck Horn	2520
	Car/Truck Music	10374
	Engine Idling	4189
Posting Advertisement	Building	60
	Vehicle	590
Traffic	Chronic Speeding	268
	Chronic Stoplight Violation	280
	Congestion/Gridlock	2761
	Drag Racing	175
	Truck Route Violation	1014
Vending	In Prohibited Area	2025
	Unlicensed	1777

Order the complaint types based on the average 'Request_Closing_Time', grouping them for different locations.

In [39]:
```
# For location we can choose here City, so first check if there is missing values there
data['City'].isnull().sum()
# Fill all missing values
data['City'].fillna('Not Available', inplace=True)
data['City'].head(1)
```
Out[39]:
```
0    NEW YORK
Name: City, dtype: object
```

In [40]:
```
# Group them for City then Complain Type
data_grouped = data.groupby(['City', 'Complaint Type'])
# get average and get Request_Closing_Time column
data_mean = data_grouped.mean()['Request_Closing_In_Hr']
data_mean.isnull().sum()
```
Out[40]:
```
4
```

In [41]:
```
# Group by City first & then Complain Type and showing average of Request Closing in Hour
data_grouped = data.groupby(['City','Complaint Type']).agg({'Request_Closing_In_Hr': 'mean'})
data_grouped
```
Out[41]:

City	Complaint Type	Request_Closing_In_Hr
ARVERNE	Animal Abuse	2.153158
	Blocked Driveway	2.526000
	Derelict Vehicle	2.968889
	Disorderly Youth	3.595000
	Drinking	0.240000
...	...	...
Woodside	Blocked Driveway	6.405455
	Derelict Vehicle	4.965000
	Illegal Parking	5.219500
	Noise - Commercial	2.390000
	Noise - Street/Sidewalk	3.410000

782 rows × 1 columns

In [42]:
```
# Check if any value is null
data_grouped[data_grouped['Request_Closing_In_Hr'].isnull()]
```
Out[42]:

City	Complaint Type	Request_Closing_In_Hr
Not Available	Ferry Complaint	NaN
	Noise - House of Worship	NaN
	Panhandling	NaN
	Posting Advertisement	NaN

In [43]:
```
# Check total rows
print(data_grouped)
```
```
                                    Request_Closing_In_Hr
City     Complaint Type
ARVERNE  Animal Abuse                            2.153158
         Blocked Driveway                        2.526000
         Derelict Vehicle                        2.968889
         Disorderly Youth                        3.595000
         Drinking                                0.240000
...                                                   ...
Woodside Blocked Driveway                        6.405455
         Derelict Vehicle                        4.965000
         Illegal Parking                         5.219500
         Noise - Commercial                      2.390000
         Noise - Street/Sidewalk                 3.410000

[782 rows x 1 columns]
```

In [44]:
```
# drop null values from this group
data_grouped_withoutna = data_grouped.dropna()
# verify if new group has null values
data_grouped_withoutna.isnull().sum()
```
Out[44]:
```
Request_Closing_In_Hr    0
dtype: int64
```

In [45]:
```
# verify number of rows after dropping null values
print(data_grouped_withoutna)
```
```
                                    Request_Closing_In_Hr
City     Complaint Type
ARVERNE  Animal Abuse                            2.153158
         Blocked Driveway                        2.526000
         Derelict Vehicle                        2.968889
         Disorderly Youth                        3.595000
         Drinking                                0.240000
...                                                   ...
Woodside Blocked Driveway                        6.405455
         Derelict Vehicle                        4.965000
         Illegal Parking                         5.219500
         Noise - Commercial                      2.390000
         Noise - Street/Sidewalk                 3.410000

[778 rows x 1 columns]
```

In [46]:
```
# Sorting by column - Request_Closing_In_Hr for City on grouped
data_sorted = data_grouped_withoutna.sort_values(['City', 'Request_Closing_In_Hr'])
data_sorted
```
Out[46]:

City	Complaint Type	Request_Closing_In_Hr
ARVERNE	Drinking	0.240000
	Vending	0.480000
	Urinating in Public	0.690000
	Panhandling	1.030000
	Noise - Park	1.285000
...	...	...
Woodside	Noise - Commercial	2.390000
	Noise - Street/Sidewalk	3.410000
	Derelict Vehicle	4.965000
	Illegal Parking	5.219500
	Blocked Driveway	6.405455

778 rows × 1 columns

In [47]:
```
# Sorting
sort_data = data_grouped_withoutna.sort_values(['City', 'Request_Closing_In_Hr'])
```

Perform a statistical test for the following:

In [48]:
```
# Please note: For the below statements you need to state the Null and Alternate and then provide a statistical

import scipy.stats as stats
from math import sqrt

# H0 : All Complain Types average response time mean is similar
# H1 : Not similar

data['Complaint Type'].value_counts()
```
Out[48]:
```
Blocked Driveway             77044
Illegal Parking              75361
Noise - Street/Sidewalk      48612
Noise - Commercial           35577
Derelict Vehicle             17718
Noise - Vehicle              17083
Animal Abuse                  7778
Traffic                       4498
Homeless Encampment           4416
Noise - Park                  4042
Vending                       3802
Drinking                      1280
Noise - House of Worship       931
Posting Advertisement          650
Urinating in Public            592
Bike/Roller/Skate Chronic      427
Panhandling                    307
Disorderly Youth               286
Illegal Fireworks              168
Graffiti                       113
Agency Issues                    6
Squeegee                         4
Ferry Complaint                  2
Animal in a Park                 1
Name: Complaint Type, dtype: int64
```

In [49]:
```
complaints_type_5top = data['Complaint Type'].value_counts()[:5]
complaints_type_5top
```
Out[49]:
```
Blocked Driveway           77044
Illegal Parking            75361
Noise - Street/Sidewalk    48612
Noise - Commercial         35577
Derelict Vehicle           17718
Name: Complaint Type, dtype: int64
```

In [50]:
```
complaints_type_top5_names = complaints_type_5top.index
complaints_type_top5_names
```
Out[50]:
```
Index(['Blocked Driveway', 'Illegal Parking', 'Noise - Street/Sidewalk',
       'Noise - Commercial', 'Derelict Vehicle'],
      dtype='object')
```

In [51]:
```
sample_data = data.loc[data['Complaint Type'].isin(complaints_type_top5_names), ['Complaint Type', 'Request_Clo
sample_data.head()
```
Out[51]:

	Complaint Type	Request_Closing_In_Hr
0	Noise - Street/Sidewalk	0.92
1	Blocked Driveway	1.44
2	Blocked Driveway	4.86
3	Illegal Parking	7.75
4	Illegal Parking	3.45

In [52]:
```
sample_data.isnull().sum()
```
Out[52]:
```
Complaint Type              0
Request_Closing_In_Hr    2059
dtype: int64
```

In [53]:
```
sample_data.shape
```
Out[53]:
```
(254312, 2)
```

```
In [ ]: #sample_data[~sample_data.isin(['NaN', 'NaT'])].any(axis=1)
        #sample_data[sample_data.isnull()]

        sample_data.dropna(how='any', inplace=True)
        sample_data.isnull().sum()
        # sample_data_without_null(sample_data_without_null.isnull())
```

```
Out[54]: Complaint Type           0
         Request_Closing_In_Hr    0
         dtype: int64
```

```
In [55]: s1 = sample_data[sample_data['Complaint Type'] == complaints_type_top5_names[0]].Request_Closing_In_Hr
         s1.head()
```

```
Out[55]: 1     1.44
         2     4.86
         4     1.80
         9     1.38
         10    7.80
         Name: Request_Closing_In_Hr, dtype: float64
```

```
In [56]: s2 = sample_data[sample_data['Complaint Type'] == complaints_type_top5_names[1]].Request_Closing_In_Hr
         s2.head()
```

```
Out[56]: 3    7.75
         4    3.45
         5    1.98
         6    1.96
         8    8.55
         Name: Request_Closing_In_Hr, dtype: float64
```

```
In [57]: s3 = sample_data[sample_data['Complaint Type'] == complaints_type_top5_names[2]].Request_Closing_In_Hr
         s3.head()
```

```
Out[57]: 0     0.92
         12    2.48
         19    0.78
         38    0.49
         54    1.50
         Name: Request_Closing_In_Hr, dtype: float64
```

```
In [58]: s4 = sample_data[sample_data['Complaint Type'] == complaints_type_top5_names[3]].Request_Closing_In_Hr
         s4.head()
```

```
Out[58]: 17    0.85
         18    2.93
         22    1.26
         29    2.50
         30    1.99
         Name: Request_Closing_In_Hr, dtype: float64
```

```
In [59]: print(s1.isnull().sum())
         print(s2.isnull().sum())
         print(s3.isnull().sum())
         print(s4.isnull().sum())

         0
         0
         0
         0
```

```
In [60]: stats.f_oneway(s1, s2, s3, s4)
```

```
Out[60]: F_onewayResult(statistic=1055.48716871555, pvalue=0.0)
```

We can see pvalue is less than 0.05 so we reject null hypothesis and average response time is not same

```
In [ ]: ### Try ChiSquare Test for second one = # Are the type of complaint or service requested and location related?

        # H0 : 2 categories - Complain Type and Location is is independent means not related
        # Ha : 2 categories - Complain Type and Location is is dependent means related
```

```
In [71]: top_10_city = data['City'].value_counts()[:10]
         top_10_city
```

```
Out[71]: BROOKLYN         98307
         NEW YORK         65994
         BRONX            40702
         STATEN ISLAND    12343
         JAMAICA           7296
         ASTORIA           6330
         FLUSHING          5971
         RIDGEWOOD         5163
         CORONA            4295
         WOODSIDE          3544
         Name: City, dtype: int64
```

```
In [72]: top10_city_names = top_10_city.index
         top10_city_names
```

```
Out[72]: Index(['BROOKLYN', 'NEW YORK', 'BRONX', 'STATEN ISLAND', 'JAMAICA', 'ASTORIA',
                'FLUSHING', 'RIDGEWOOD', 'CORONA', 'WOODSIDE'],
               dtype='object')
```

```
In [81]: sample_data_city = data.loc[(data['Complaint Type'].isin(top10_city_names)) & (data['City'].isin(top10_city_nam
         sample_data_city.head()
```

Out[81]:

Complaint Type	City

```
In [172]: chi2, p_value, df, exp_freq = stats.chi2_contingency(pd.crosstab(sample_data_city['Complaint Type'], sample_data_
          print(chi2)
          print(p_value)

          50406.906171855495
          0.0
```

Second way to check

Null and Alternate and then provide a statistical test to accept or reject the Null Hypothesis along with the corresponding 'p-value'.

```
In [84]: date_data = pd.read_csv("311_Service_Requests_from_2010_to_Present.csv", parse_dates=["Created Date", "Closed D
         date_data["Request_Closing_Time"] = date_data["Closed Date"] - date_data["Created Date"]
         date_data.Request_Closing_Time = date_data.Request_Closing_Time.astype('int64') / 10**9
         date_data

         C:\Users\abman\AppData\Local\Temp\ipykernel_11320\793570299.py:1: DtypeWarning: Columns (48,49) have mixed type
         s. Specify dtype option on import or set low_memory=False.
           date_data = pd.read_csv("311_Service_Requests_from_2010_to_Present.csv", parse_dates=["Created Date", "Closed
```

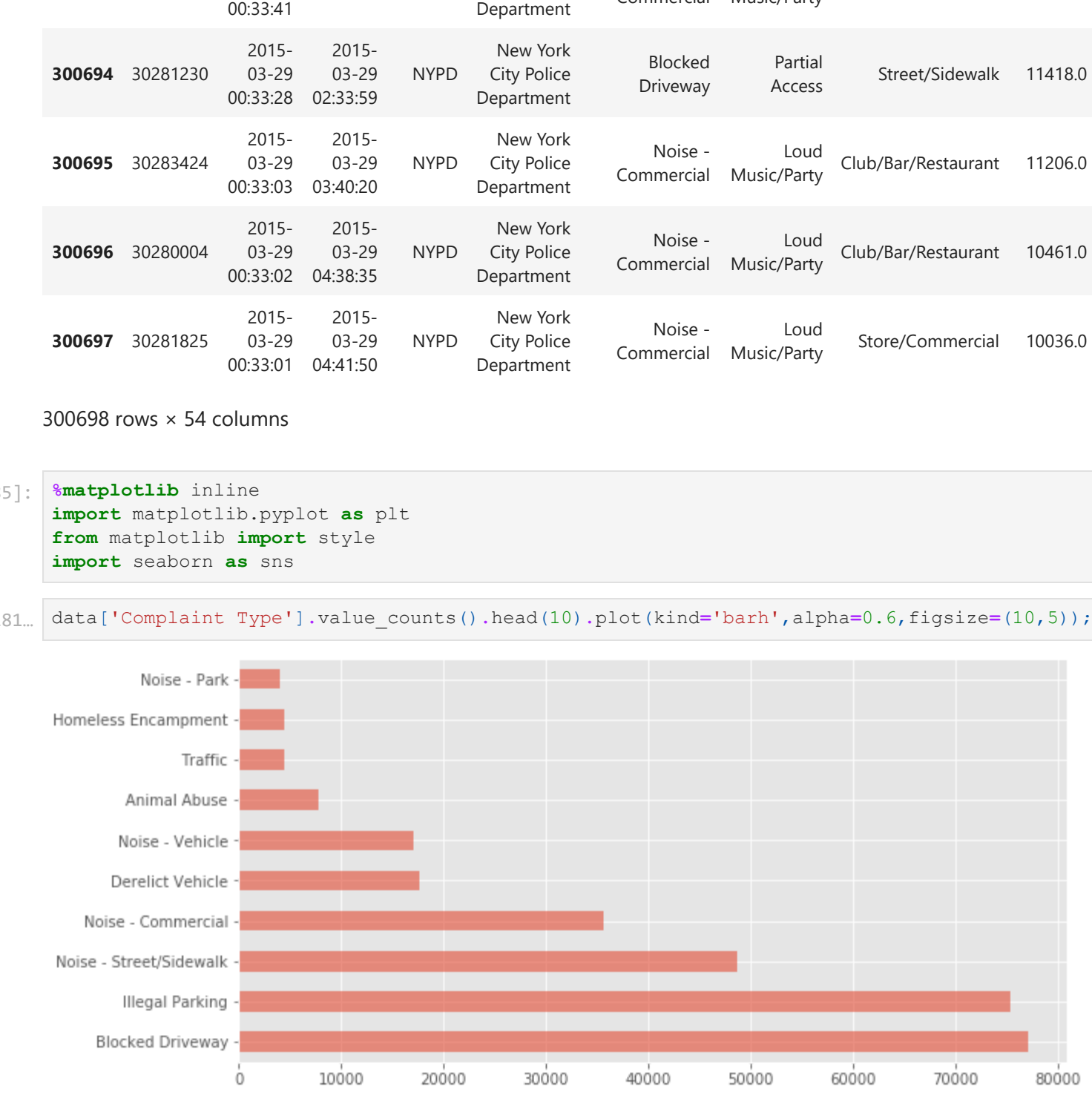
Out[84]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Direction
0	32310363	2015-12-31 23:59:45	2016-01-01 00:55:00	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN
1	32309934	2015-12-31 23:59:44	2016-01-01 01:26:00	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN
2	32309159	2015-12-31 23:59:29	2016-01-01 04:51:00	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN
3	32305098	2015-12-31 23:57:46	2016-01-01 07:43:00	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN
4	32306529	2015-12-31 23:56:58	2016-01-01 03:24:00	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
300693	30281872	2015-03-29 00:33:41	NaT	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	NaN	CRESCENT AVENUE	...	NaN
300694	30281230	2015-03-29 00:33:28	2015-03-29 02:33:59	NYPD	New York City Police Department	Blocked Driveway	Partial Access	Street/Sidewalk	11418.0	100-17 87 AVENUE	...	NaN
300695	30283424	2015-03-29 00:33:03	2015-03-29 03:40:20	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	11206.0	162 THROOP AVENUE	...	NaN
300696	30280004	2015-03-29 00:33:02	2015-03-29 04:38:35	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	10461.0	3151 EAST TREMONT AVENUE	...	NaN
300697	30281825	2015-03-29 00:33:01	2015-03-29 04:41:50	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Store/Commercial	10036.0	251 WEST 48 STREET	...	NaN

300698 rows × 54 columns

```
In [85]: %matplotlib inline
         import matplotlib.pyplot as plt
         from matplotlib import style
         import seaborn as sns
```

```
In [181]: data['Complaint Type'].value_counts().head(10).plot(kind='barh',alpha=0.6,figsize=(10,5))
```




```
In [86]: meanclose1= date_data['Request_Closing_Time'].mean()
         meanclose1
```

```
Out[86]: -66361400.168607704
```

```
In [119]: from scipy.stats import normaltest
          meanclose= date_data['Request_Closing_Time'].mean()
          stat, p = normaltest(date_data['Request_Closing_Time'])
          meanclose
```

```
Out[119]: -5742847521.905661
```

```
In [87]: from scipy.stats import ttest_1samp
         tset, pval = ttest_1samp(date_data['Request_Closing_Time'], meanclose1)
         pval
```

```
Out[87]: 1.0
```

```
In [94]: from scipy import stats
         from statsmodels.stats import weightstats as stests
         ztest ,pval = stests.ztest(date_data['Request_Closing_Time'], x2=None, value=date_data['Request_Closing_Time'].mean())
         print(float(pval))

         1.0
```

```
In [88]: contingency_table=pd.crosstab(data["Complaint Type"],data["Location Type"])
         print('contingency_table :-\n',contingency_table)#Observed Values

         Observed_Values = contingency_table.values
         print("Observed Values :-\n",Observed_Values)

         b=stats.chi2_contingency(contingency_table)
         Expected_Values = b[3]
         print("Expected Values :-\n",Expected_Values)

contingency_table :-
 Location Type              Bridge  Club/Bar/Restaurant  Commercial  Ferry  \
Complaint Type
Animal Abuse                    0                    0          62      0
Animal in a Park                0                    0           0      0
Bike/Roller/Skate Chronic       0                    0           0      0
Blocked Driveway                0                    0           0      0
Derelict Vehicle                0                    0           0      0
Disorderly Youth                0                    0           0      0
Drinking                        0                  366           0      0
Ferry Complaint                 0                    0           0      1
Graffiti                        0                    0           0      0
Homeless Encampment             2                    0           0      0
Illegal Fireworks               0                    0           0      0
Illegal Parking                 0                    0           0      0
Noise - Commercial              0                16973           0      0
Noise - House of Worship        0                    0           0      0
Noise - Park                    0                    0           0      0
Noise - Street/Sidewalk         0                    0           0      0
Noise - Vehicle                 0                    0           0      0
Panhandling                     0                    0           0      0
Posting Advertisement           0                    0           0      0
Squeegee                        0                    0           0      0
Traffic                         0                    0           0      0
Urinating in Public             0                   21           0      0
Vending                         0                    0           0      0

Location Type              Highway  House and Store  House of Worship  Park  \
Complaint Type
Animal Abuse                     0               93                 0     0
Animal in a Park                 0                0                 0     1
Bike/Roller/Skate Chronic        0                0                 0     0
Blocked Driveway                 0                0                 0     0
Derelict Vehicle                14                0                 0     0
Disorderly Youth                 0                0                 0     0
Drinking                         0                0                 0     0
Ferry Complaint                  0                0                 0     0
Graffiti                         0                0                 0     0
Homeless Encampment             15                0                 0     0
Illegal Fireworks                0                0                 0     0
Illegal Parking                  0                0                 0     0
Noise - Commercial               0                0                 0     0
Noise - House of Worship         0                0               929     0
Noise - Park                     0                0                 0     0
Noise - Street/Sidewalk          0                0                 0     0
Noise - Vehicle                  0                0                 0     0
Panhandling                      0                0                 0     0
Posting Advertisement            0                0                 0     0
Squeegee                         0                0                 0     0
Traffic                        186                0                 0     0
Urinating in Public              0                0                 0     0
Vending                          0                0                 0     0

Location Type              Park/Playground  Parking Lot  Residential Building  \
Complaint Type
Animal Abuse                           123          110                   227
Animal in a Park                         0            0                     0
Bike/Roller/Skate Chronic                0            0                     0
Blocked Driveway                         0            0                     0
Derelict Vehicle                         0            0                     0
Disorderly Youth                         0            0                     0
Drinking                                98            0                     0
Ferry Complaint                          0            0                     0
Graffiti                                 0            0                     0
Homeless Encampment                    353            0                     0
Illegal Fireworks                        8            0                     0
Illegal Parking                          0            0                     0
Noise - Commercial                       0            0                     0
Noise - House of Worship                 0            0                     0
Noise - Park                          4041            0                     0
Noise - Street/Sidewalk                  0            0                     0
Noise - Vehicle                          0            0                     0
Panhandling                              6            0                     0
Posting Advertisement                    0            7                     0
Squeegee                                 0            0                     0
Traffic                                  0            0                     0
Urinating in Public                     38            0                     0
Vending                                106            0                     0

Location Type              Residential Building/House  Roadway Tunnel  \
Complaint Type
Animal Abuse                                     5085               0
Animal in a Park                                    0               0
Bike/Roller/Skate Chronic                          26               0
Blocked Driveway                                    0               0
Derelict Vehicle                                    0               5
Disorderly Youth                                   77               0
Drinking                                          291               0
Ferry Complaint                                     0               0
Graffiti                                           56               0
Homeless Encampment                               983               1
Illegal Fireworks                                  33               0
Illegal Parking                                     0               0
Noise - Commercial                                  0               0
Noise - House of Worship                            0               0
Noise - Park                                        0               0
Noise - Street/Sidewalk                             0               0
Noise - Vehicle                                     0               0
Panhandling                                        16               0
Posting Advertisement                              54               0
Squeegee                                            0               0
Traffic                                             0              29
Urinating in Public                               138               0
Vending                                           201               0

Location Type              Store/Commercial  Street/Sidewalk  Subway Station  \
Complaint Type
Animal Abuse                            522             1531              22
Animal in a Park                          0                0               0
Bike/Roller/Skate Chronic                53              348               0
Blocked Driveway                          0            77007               0
Derelict Vehicle                          0            17614               0
Disorderly Youth                          8              201               0
Drinking                                 90              434               0
Ferry Complaint                           0                0               0
Graffiti                                 32               25               0
Homeless Encampment                     512             2541               0
Illegal Fireworks                         2              125               0
Illegal Parking                           0            75326               0
Noise - Commercial                    18598                0               0
Noise - House of Worship                  0                0               0
Noise - Park                              0                0               0
Noise - Street/Sidewalk                   0            48601               0
Noise - Vehicle                           0            17080               0
Panhandling                              60              225               0
Posting Advertisement                     6              582               0
Squeegee                                  0                4               0
Traffic                                   0             4278               0
Urinating in Public                      66              316              12
Vending                                 432             3061               0

Location Type              Terminal  Vacant Lot
Complaint Type
Animal Abuse                      0           0
Animal in a Park                  0           0
Bike/Roller/Skate Chronic         0           0
Blocked Driveway                  0           0
Derelict Vehicle                  0          77
Disorderly Youth                  0           0
Drinking                          0           0
Ferry Complaint                   1           0
Graffiti                          0           0
Homeless Encampment               0           0
Illegal Fireworks                 0           0
Illegal Parking                   0           0
Noise - Commercial                0           0
Noise - House of Worship          0           0
Noise - Park                      0           0
Noise - Street/Sidewalk           0           0
Noise - Vehicle                   0           0
Panhandling                       0           0
Posting Advertisement             0           0
Squeegee                          0           0
Traffic                           0           0
Urinating in Public               0           0
Vending                           0           0
Observed Values :-
 [[    0     0    62     0     0    93     0     0   123   110   227  5085
      0   522  1531    22     0     0]
 [    0     0     0     0     0     0     0     1     0     0     0     0
      0     0     0     0     0     0]
 [    0     0     0     0     0     0     0     0     0     0     0    26
      0    53   348     0     0     0]
 [    0     0     0     0     0     0     0     0     0     0     0     0
      0     0 77007     0     0     0]
 [    0     0     0     0    14     0     0     0     0     0     0     0
      5     0 17614     0     0    77]
 [    0     0     0     0     0     0     0     0     0     0     0    77
      0     8   201     0     0     0]
 [    0   366     0     0     0     0     0     0    98     0     0   291
      0    90   434     0     0     0]
 [    0     0     0     1     0     0     0     0     0     0     0     0
      0     0     0     0     1     0]
 [    0     0     0     0     0     0     0     0     0     0     0    56
      0    32    25     0     0     0]
 [    2     0     0     0    15     0     0     0   353     0     0   983
      1   512  2541     0     0     0]
 [    0     0     0     0     0     0     0     0     8     0     0    33
      0     2   125     0     0     0]
 [    0     0     0     0     0     0     0     0     0     0     0     0
      0     0 75326     0     0     0]
 [    0 16973     0     0     0     0     0     0     0     0     0     0
      0 18598     0     0     0     0]
 [    0     0     0     0     0     0   929     0     0     0     0     0
      0     0     0     0     0     0]
 [    0     0     0     0     0     0     0     0  4041     0     0     0
      0     0     0     0     0     0]
 [    0     0     0     0     0     0     0     0     0     0     0     0
      0     0 48601     0     0     0]
 [    0     0     0     0     0     0     0     0     0     0     0     0
      0     0 17080     0     0     0]
 [    0     0     0     0     0     0     0     0     6     0     0    16
      0    60   225     0     0     0]
 [    0     0     0     0     0     0     0     0     0     7     0    54
      0     6   582     0     0     0]
 [    0     0     0     0     0     0     0     0     0     0     0     0
      0     0     4     0     0     0]
 [    0     0     0     0   186     0     0     0     0     0     0     0
     29     0  4278     0     0     0]
 [    0    21     0     0     0     0     0     0    38     0     0   138
      0    66   316    12     0     0]
 [    0     0     0     0     0     0     0     0   106     0     0   201
      0   432  3061     0     0     0]]
Expected Values :-
 [[5.17355531e-02 4.49064601e+02 1.60380215e+00 2.58677766e-02
  5.56157196e+00 2.40570322e+00 2.40311644e+01 2.58677766e-02
  1.23466898e+02 3.02652986e+00 5.87198528e+00 1.80039725e+02
  9.05372180e-01 5.27211154e+02 6.44881083e+03 8.79504403e-01
  2.58677766e-02 1.99181880e+00]
 [6.65409044e-06 5.77575050e-02 2.06276804e-04 3.32704522e-06
  7.15314722e-04 3.09415205e-04 3.09082501e-03 3.32704522e-06
  1.58799868e-02 3.89264290e-04 7.55239264e-04 2.31562347e-02
  1.16446583e-04 6.78085086e-02 8.29429046e-01 1.13119537e-04
  3.32704522e-06 2.56182482e-04]
 [2.84129662e-03 2.46624546e+01 8.80801951e-02 1.42064831e-03
  3.05439386e-01 1.32120293e-01 1.31978228e+00 1.42064831e-03
  6.78075437e+00 1.66215852e-01 3.22487166e-01 9.88771222e+00
  4.97226908e-02 2.89542332e+01 3.54166203e+02 4.83020425e-02
  1.42064831e-03 1.09389920e-01]
 [5.12411542e-01 4.44773219e+03 1.58847578e+01 2.56205771e-01
  5.50842408e+01 2.38271367e+01 2.38015161e+02 2.56205771e-01
  1.22287015e+03 2.99760752e+01 5.81587100e+01 1.78319217e+03
  8.96720199e+00 5.22172982e+03 6.38718425e+04 8.71099622e+00
  2.56205771e-01 1.97278444e+01]
 [1.17843942e-01 1.02288541e+03 3.65316219e+00 5.89219708e-02
  1.26682237e+01 5.47974329e+00 5.47385109e+01 5.89219708e-02
  2.81234567e+02 6.89387058e+00 1.33752874e+01 4.10096917e+02
  2.06226898e+00 1.20088869e+03 1.46891884e+04 2.00334701e+00
  5.89219708e-02 4.53699175e+00]
 [1.90306986e-03 1.65186464e+01 5.89951658e-02 9.51534932e-04
  2.04580010e-01 8.84927487e-02 8.83975952e-01 9.51534932e-04
  4.54167623e+00 1.11329587e-01 2.15998430e-01 6.62268313e+00
  3.33037226e-02 1.93932335e+01 2.37216707e+02 3.23521877e-02
  9.51534932e-04 7.32681898e-02]
 [8.51058167e-03 7.38718489e+01 2.63828032e-01 4.25529083e-03
  9.14887529e-01 3.95742048e-01 3.95316518e+00 4.25529083e-03
  2.03105031e+01 4.97869028e-01 9.65951019e-01 2.96168242e+01
  1.48935179e-01 8.67270825e+01 1.06083975e+03 1.44679888e-01
  4.25529083e-03 3.27657394e-01]
 [1.33081809e-05 1.15515010e-01 4.12553607e-04 6.65409044e-06
  1.43062944e-03 6.18830411e-04 6.18165001e-03 6.65409044e-06
  3.17599736e-02 7.78528581e-04 1.51047853e-03 4.63124694e-02
  2.32893165e-04 1.35617017e-01 1.65885809e+00 2.26239075e-04
  6.65409044e-06 5.12364964e-04]
 [7.51912219e-04 6.52659806e+00 2.33092788e-02 3.75956110e-04
  8.08305636e-02 3.49639182e-02 3.49263226e-01 3.75956110e-04
  1.79443851e+00 4.39868648e-02 8.53420369e-02 2.61665452e+00
  1.31584638e-02 7.66236147e+00 9.37254822e+01 1.27825077e-02
  3.75956110e-04 2.89486204e-02]
 [2.93245766e-02 2.54537324e+02 9.09061873e-01 1.46622883e-02
  3.15239198e+00 1.36359281e+00 1.36212658e+01 1.46622883e-02
  6.99831019e+01 1.71548773e+00 3.32833944e+00 1.02049526e+02
  5.13180090e-01 2.98832097e+02 3.65529380e+03 4.98517801e-01
  1.46622883e-02 1.12899620e+00]
 [1.11788719e-03 9.70326084e+00 3.46545030e-02 5.58943597e-04
  1.20172873e-01 5.19817545e-02 5.19258601e-01 5.58943597e-04
  2.66783779e+00 6.53964008e-02 1.26880196e-01 3.89024743e+00
  1.95630259e-02 1.13918294e+01 1.39344080e+02 1.90040823e-02
  5.58943597e-04 4.30386569e-02]
 [5.01226016e-01 4.35064182e+03 1.55380065e+01 2.50613008e-01
  5.38817967e+01 2.33070098e+01 2.32819485e+02 2.50613008e-01
  1.19617589e+03 2.93217219e+01 5.68891528e+01 1.74426654e+03
  8.77145528e+00 5.10774372e+03 6.24775723e+04 8.52084227e+00
  2.50613008e-01 1.92972016e+01]
 [2.36692651e-01 2.05449221e+03 7.33747218e+00 1.18346325e-01
  2.54444600e+01 1.10062083e+01 1.09943736e+02 1.18346325e-01
  5.64867011e+02 1.38465201e+01 2.68646159e+01 8.23690425e+02
  4.14212139e+00 2.41201646e+03 2.95036206e+04 4.02377507e+00
  1.18346325e-01 9.11266706e+00]
 [6.18165001e-03 5.36567221e+01 1.91631150e-01 3.09082501e-03
  6.64527377e-01 2.87446726e-01 2.87137643e+00 3.09082501e-03
  1.47525078e+01 3.61626526e-01 7.01617277e-01 2.15121421e+01
  1.08178875e-01 6.29941045e+01 7.70539584e+02 1.05088050e-01
  3.09082501e-03 2.37993526e-01]
 [2.68891795e-02 2.33398078e+02 8.33564563e-01 1.34445897e-02
  2.89058679e+00 1.25034684e+00 1.24900239e+01 1.34445897e-02
  6.41710268e+01 1.57301700e+00 3.05192187e+00 9.35743445e+01
  4.70560640e-01 2.74014183e+02 3.35172277e+03 4.57116051e-01
  1.34445897e-02 1.03523341e+00]
 [3.23395449e-01 2.80707250e+03 1.00252589e+01 1.61697725e-01
  3.47650108e+01 1.50378884e+01 1.50217186e+02 1.61697725e-01
  7.71783240e+02 1.89186338e+01 3.67053835e+01 1.12541616e+03
  5.65942036e+00 3.29556133e+03 4.03110811e+04 5.49772264e+00
  1.61697725e-01 1.24507248e+01]
 [1.13651865e-01 9.86498185e+02 3.52320780e+00 5.68259323e-02
  1.22175754e+01 5.28481171e+00 5.27912911e+01 5.68259323e-02
  2.71230175e+02 6.64863408e+00 1.28994866e+01 3.95508489e+02
  1.98890763e+00 1.15816933e+03 1.41666481e+04 1.93208170e+00
  5.68259323e-02 4.37559679e+00]
 [2.04280576e-03 1.77315540e+01 6.33269787e-02 1.02140288e-03
  2.19601620e-01 9.49904680e-02 9.48883277e-01 1.02140288e-03
  4.87515596e+00 1.19504137e-01 2.31858454e-01 7.10896406e+00
  3.57491009e-02 2.08172121e+01 2.54634717e+02 3.47276980e-02
  1.02140288e-03 7.86480219e-02]
 [4.31850469e-03 3.74846207e+01 1.33873645e-01 2.15925235e-03
  4.64239254e-01 2.00810468e-01 2.00594543e+00 2.15925235e-03
  1.03061114e+01 2.52632525e-01 4.90150283e-01 1.50283963e+01
  7.55738321e-02 4.40077221e+01 5.38299451e+02 7.34145798e-02
  2.15925235e-03 1.66262431e-01]
 [2.66163617e-05 2.31030020e-01 8.25107214e-04 1.33081809e-05
  2.86125889e-03 1.23766082e-03 1.23633000e-02 1.33081809e-05
  6.35199473e-02 1.55705716e-03 3.02095706e-03 9.26249389e-02
  4.65786331e-04 2.71234034e-01 3.31771618e+00 4.52478150e-04
  1.33081809e-05 1.02472993e-03]
 [2.98968283e-02 2.59504470e+02 9.26801678e-01 1.49484142e-02
  3.21390905e+00 1.39020252e+00 1.38870768e+01 1.49484142e-02
  7.13487808e+01 1.74896446e+00 3.39329002e+00 1.04040963e+02
  5.23194496e-01 3.04663629e+02 3.72662470e+03 5.08246082e-01
  1.49484142e-02 1.15102789e+00]
 [3.93256745e-03 3.41346854e+01 1.21909591e-01 1.96628372e-03
  4.22751001e-01 1.82864386e-01 1.82667758e+00 1.96628372e-03
  9.38507221e+00 2.30055196e-01 4.46346405e-01 1.36853347e+01
  6.88199303e-02 4.00748286e+01 4.90192566e+02 6.68536466e-02
  1.96628372e-03 1.51403847e-01]
 [2.52855437e-02 2.19478519e+02 7.83851853e-01 1.26427718e-02
  2.71819594e+00 1.17577778e+00 1.17451350e+01 1.26427718e-02
  6.03439499e+01 1.47920430e+00 2.86990920e+00 8.79936919e+01
  4.42497014e-01 2.57672333e+02 3.15183037e+03 4.29854242e-01
  1.26427718e-02 9.73493431e-01]]
```

```
In [ ]: no_of_rows=len(contingency_table.iloc[0:2,0])
        print("no_of_rows:-",no_of_rows)
        no_of_columns=len(contingency_table.iloc[0,0:2])
        print("no_of_columns:-",no_of_columns)
        ddof=(no_of_rows-1)*(no_of_columns-1)
        print("Degree of Freedom:-",ddof)
        alpha = 0.05
```

```
In [ ]: from scipy.stats import chi2
        chi_square=sum([(o-e)**2./e for o,e in zip(Observed_Values,Expected_Values)])
        chi_square_statistic=chi_square[0]+chi_square[1]
        print("chi-square statistic:-",chi_square_statistic)
```

```
In [ ]: critical_value=chi2.ppf(q=1-alpha,df=ddof)
        print('critical_value:',critical_value) #p-value
        p_value=1-chi2.cdf(x=chi_square_statistic,df=ddof)
        critical_value=chi2.ppf(q=1-alpha,df=ddof)
        print('critical_value:',critical_value) #p-value

        p_value=1-chi2.cdf(x=chi_square_statistic,df=ddof)
        print('p-value:',p_value)
        print('Significance level: ',alpha)
        print('Degree of Freedom: ',ddof)
        print('chi-square statistic:',chi_square_statistic)
        print('critical_value:',critical_value)
        print('p-value:',p_value)
        if chi_square_statistic>=critical_value:
            print("Reject H0,There is a relationship between 2 categorical variables")
        else:
            print("Retain H0,There is no relationship between 2 categorical variables")

        if p_value<=alpha:
            print("Reject H0,There is a relationship between 2 categorical variables")
        else:
            print("Retain H0,There is no relationship between 2 categorical variables")
```

```
In [ ]: from scipy.stats import chi2_contingency
        chi2_contingency(pd.crosstab(data["Complaint Type"],data["Location Type"]))
```

```
In [ ]: from scipy.stats import chi2_contingency
        from scipy.stats import chi2
        # contingency table
        table=pd.crosstab(data["Complaint Type"],data["Location Type"])
        print(table)
        stat, p, dof, expected = chi2_contingency(table)
        print('dof=%d' % dof)
        print(expected)
        # interpret test-statistic
        prob = 0.95
        critical = chi2.ppf(prob, dof)
        print('probability=%.3f, critical=%.3f, stat=%.3f' % (prob, critical, stat))
        if abs(stat) >= critical:
                print('Dependent (reject H0)')
        else:
                print('Independent (fail to reject H0)')
        # interpret p-value
        alpha = 1.0 - prob
        print('significance=%.3f, p=%.3f' % (alpha, p))
        if p <= alpha:
                print('Dependent (reject H0)')
        else:
                print('Independent (fail to reject H0)')
```

```
In [90]: majorcomplints=data .dropna(subset=["Complaint Type"])
         majorcomplints=data .groupby("Complaint Type")

         sortedComplaintType = majorcomplints.size().sort_values(ascending = False)
         sortedComplaintType = sortedComplaintType.to_frame('count').reset_index()

         sortedComplaintType
         sortedComplaintType.head(12)
```

Out[90]:

	Complaint Type	count
0	Blocked Driveway	77044
1	Illegal Parking	75361
2	Noise - Street/Sidewalk	48612
3	Noise - Commercial	35577
4	Derelict Vehicle	17718
5	Noise - Vehicle	17083
6	Animal Abuse	7778
7	Traffic	4498
8	Homeless Encampment	4416
9	Noise - Park	4042
10	Vending	3802
11	Drinking	1280

```
In [91]: data.head()
```

Out[91]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Garage Lot Name	Ferry Direction	T
0	32310363	2015-12-31 23:59:45	2016-01-01 00:55:00	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN	NaN	
1	32309934	2015-12-31 23:59:44	2016-01-01 01:26:00	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN	NaN	
2	32309159	2015-12-31 23:59:29	2016-01-01 04:51:00	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN	NaN	
3	32305098	2015-12-31 23:57:46	2016-01-01 07:43:00	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN	NaN	
4	32306529	2015-12-31 23:56:58	2016-01-01 03:24:00	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN	NaN	

5 rows × 57 columns

```
In [92]: sortedComplaintType = sortedComplaintType.head()
         plt.figure(figsize=(6,6))
         plt.pie(sortedComplaintType['count'],labels=sortedComplaintType["Complaint Type"], autopct="%1.1f%%")
         plt.show()
```



```
In [ ]:
```