In [1]:

```
#importing needed libraries

import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from scipy import stats


#importing dataset into dataframe. Specified data type manually due to error message
raw = pd.read_csv(r'C:\Users\karol\Desktop\data analyst\caltech bootcmap\course 5 - applied data science with python\Project 2\311_Service_R

#check the size of the dataset
print(raw.shape)
```

```
(364558, 53)
```

In [2]:

```
#visualize the first 5 rows of the dataset
raw.head()
```

Out[2]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	Bridge Highway Segment	Garage Lot Name
0	32310363	12/31/2015 11:59:45 PM	01/01/2016 12:55:15 AM	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN	NaN	NaN	NaN	NaN
1	32309934	12/31/2015 11:59:44 PM	01/01/2016 01:26:57 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN	NaN	NaN	NaN	NaN
2	32309159	12/31/2015 11:59:29 PM	01/01/2016 04:51:03 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN	NaN	NaN	NaN	NaN
3	32305098	12/31/2015 11:57:46 PM	01/01/2016 07:43:13 AM	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN	NaN	NaN	NaN	NaN
4	32306529	12/31/2015 11:56:58 PM	01/01/2016 03:24:42 AM	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN	NaN	NaN	NaN	NaN

5 rows × 53 columns

In [3]:

```
#visualize the last 10 rows of the dataset
raw.tail(10)
```

Out[3]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	Bridg Highwa Segmer
364548	29613386	01/01/2015 12:08:34 AM	01/01/2015 02:42:23 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10467.0	800 EAST 219 STREET	...	NaN	NaN	NaN	Nal
364549	29610965	01/01/2015 12:08:02 AM	01/01/2015 01:17:43 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11368.0	NaN	...	NaN	NaN	NaN	Nal
364550	29610950	01/01/2015 12:06:43 AM	01/01/2015 06:05:18 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10473.0	616 COMMONWEALTH AVENUE	...	NaN	NaN	NaN	Nal
364551	29607567	01/01/2015 12:06:02 AM	01/01/2015 12:43:41 AM	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10453.0	NaN	...	NaN	NaN	NaN	Nal
364552	29610051	01/01/2015 12:05:05 AM	01/01/2015 01:22:10 AM	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10002.0	NaN	...	NaN	NaN	NaN	Nal
364553	29609918	01/01/2015 12:04:44 AM	01/01/2015 10:22:31 AM	NYPD	New York City Police Department	Illegal Parking	Blocked Hydrant	Street/Sidewalk	11421.0	84-25 85 ROAD	...	NaN	NaN	NaN	Nal
364554	29608392	01/01/2015 12:04:28 AM	01/01/2015 02:25:02 AM	NYPD	New York City Police Department	Noise - Vehicle	Car/Truck Horn	Street/Sidewalk	10468.0	2555 SEDGWICK AVENUE	...	NaN	NaN	NaN	Nal
364555	29607589	01/01/2015 12:01:30 AM	01/01/2015 12:20:33 AM	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10031.0	508 WEST 139 STREET	...	NaN	NaN	NaN	Nal
364556	29610889	01/01/2015 12:01:29 AM	01/01/2015 02:42:22 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10466.0	931 EAST 226 STREET	...	NaN	NaN	NaN	Nal
364557	29611816	01/01/2015 12:00:50 AM	01/01/2015 02:47:50 AM	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11420.0	123-19 135 STREET	...	NaN	NaN	NaN	Nal

10 rows × 53 columns

In [4]:

```
#column titles of the dataset
raw.columns
```

Out[4]:

```
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 [5]:

```
#variables (columns) with null values and how many in each column
raw.isna().sum()
```

Out[5]:

```
Unique Key                             0
Created Date                           0
Closed Date                         2381
Agency                                 0
Agency Name                            0
Complaint Type                         0
Descriptor                          6501
Location Type                        133
Incident Zip                        2998
Incident Address                   51699
Street Name                        51699
Cross Street 1                     57188
Cross Street 2                     57805
Intersection Street 1             313438
Intersection Street 2             314046
Address Type                        3252
City                                2997
Landmark                          364183
Facility Type                       2389
Status                                 0
Due Date                               3
Resolution Description                 0
Resolution Action Updated Date      2402
Community Board                        0
Borough                                0
X Coordinate (State Plane)          4030
Y Coordinate (State Plane)          4030
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      364558
Vehicle Type                      364558
Taxi Company Borough              364558
Taxi Pick Up Location             364558
Bridge Highway Name               364261
Bridge Highway Direction          364261
Road Ramp                         364296
Bridge Highway Segment            364296
Garage Lot Name                   364558
Ferry Direction                   364557
Ferry Terminal Name               364556
Latitude                            4030
Longitude                           4030
Location                            4030
dtype: int64
```

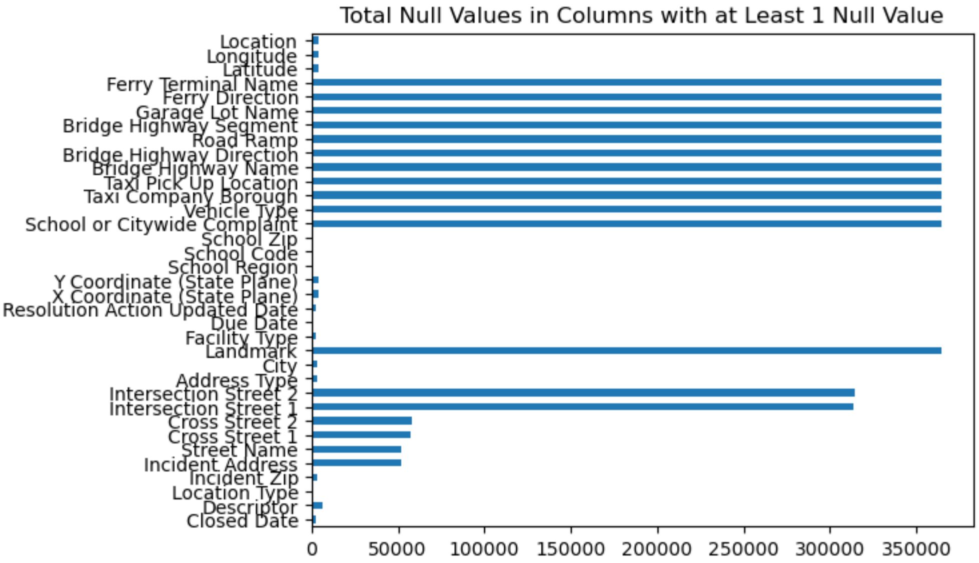
In [6]:

```python
#bar graph to show number of null values in each columnn of the dataframe.
naplotdata = raw.isna().sum()
nozerodata = naplotdata[naplotdata > 0]


nozerodata.plot(kind = 'barh')
plt.title('Total Null Values in Columns with at Least 1 Null Value')
```

Out[6]:

```
Text(0.5, 1.0, 'Total Null Values in Columns with at Least 1 Null Value')
```



In [7]:

```python
#missing value treatment

#removing the records whose Closed Date values are null
raw = raw.dropna(subset = 'Closed Date')
raw.shape
```

Out[7]:

```
(362177, 53)
```

In [9]:

```python
#2.3 analysis of the Date columns

raw['Created Date'] = pd.to_datetime(raw['Created Date'])
raw['Closed Date'] = pd.to_datetime(raw['Closed Date'])
```

In [10]:
```
#check that object to datetime conversion is correct
raw.dtypes
```

Out[10]:
```
Unique Key                                 int64
Created Date                      datetime64[ns]
Closed Date                       datetime64[ns]
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
```

In [12]:
```
#check if any dates are before 2010 (outside of the timeline)
raw[raw['Created Date']<'2010-01-01']
```

Out[12]:

Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	Bridge Highway Segment	Garage Lot Name	Ferry Direction	Ferry Terminal Name	Latitu

0 rows × 53 columns

In [13]:
```
#check if any dates are after 2023 (outside of timeline)
raw[raw['Created Date']>'2023-01-01']
```

Out[13]:

Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	Bridge Highway Segment	Garage Lot Name	Ferry Direction	Ferry Terminal Name	Latitu

0 rows × 53 columns

In [15]:
```
#creating a column with the time elapsed for service request to be completed

raw['Elapsed Time'] = raw['Closed Date'] - raw['Created Date']
```

In [16]:
```
#converting Elapsed Time to seconds into a new column Elapsed Time (sec)
raw['Elapsed Time (sec)'] = raw['Elapsed Time']/np.timedelta64(1, 's')
```

In [17]:

```
#descriptive statistics of Elapsed Time (sec). using apply lambda to change format output to float from scientific notation

raw['Elapsed Time (sec)'].describe().apply(lambda x: format(x, 'f'))
```

Out[17]:

```
count     362177.000000
mean       15113.299633
std        21102.547520
min           61.000000
25%         4533.000000
50%         9616.000000
75%        18878.000000
max      2134342.000000
Name: Elapsed Time (sec), dtype: object
```

In [18]:

```
#checking number of null values in Complaint_Type and City columns
raw[['Complaint Type', 'City']].isna().sum()
```

Out[18]:

```
Complaint Type      0
City              674
dtype: int64
```

In [19]:

```
#imputing null City values with Unknown City
raw['City'] = raw['City'].fillna('Unknown City')
```

In [20]:

```
#checking that the imputation correctly returns previous number of null values
raw['City'].value_counts()['Unknown City']
```

Out[20]:

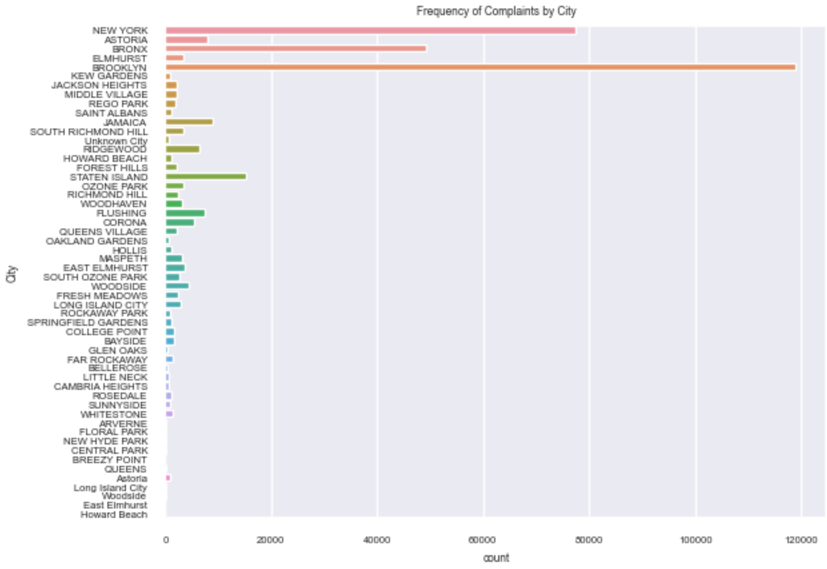
```
674
```

In [21]:

```
#frequency plot of complaints by City. showing seaborn countplot rather than pandas .plot()
sns.set(font_scale = 0.5)
sns.countplot(y='City', data=raw)
plt.title('Frequency of Complaints by City')
```

Out[21]:

```
Text(0.5, 1.0, 'Frequency of Complaints by City')
```

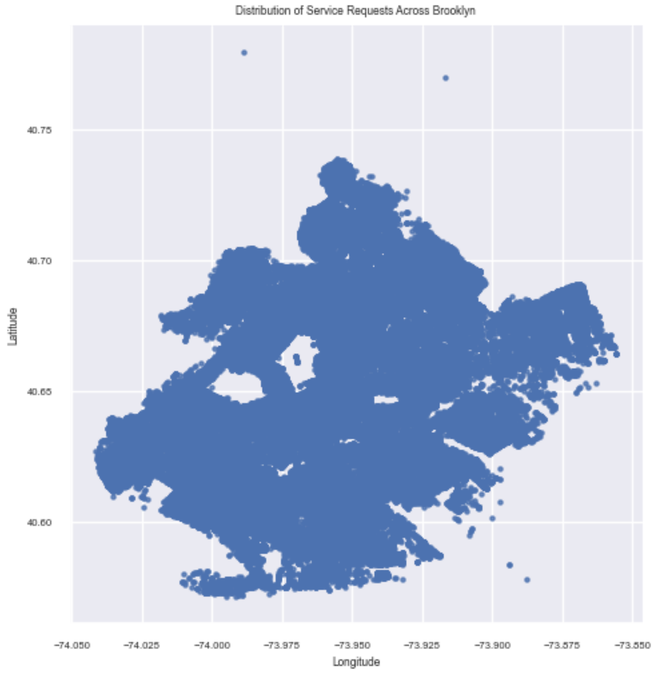


In [22]:

```
#scatterplot of request concentration across Brooklyn using X and Y coordinates, taking off trend line and decreasing point size
sns.lmplot(data = raw.loc[raw['City'] == 'BROOKLYN'], x = 'Longitude',
           y = 'Latitude', fit_reg=False, scatter_kws={'s':5}).set(title='Distribution of Service Requests Across Brooklyn')
```

Out[22]:

```
<seaborn.axisgrid.FacetGrid at 0x1c6d39aadf0>
```



Distribution of Service Requests Across Brooklyn

In [23]:

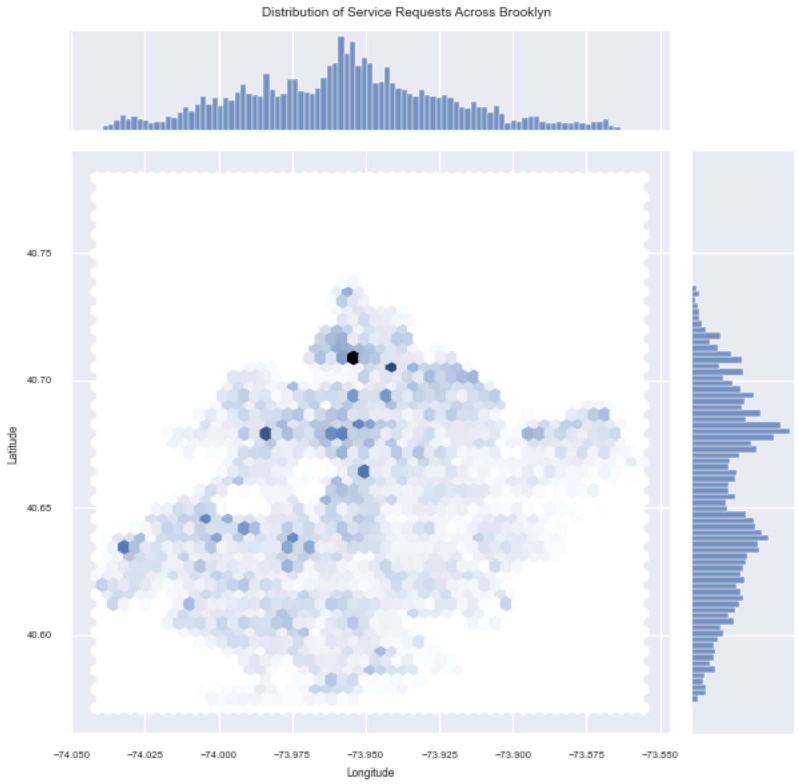
```
#hexbin (jointplot) of request concentration across Brooklyn using X and Y coordinates

hexBrook = sns.jointplot(data = raw.loc[raw['City'] == 'BROOKLYN'], x = 'Longitude',
              y = 'Latitude', kind = 'hex')

#adjust title height
hexBrook.fig.subplots_adjust(top=.95)
#add title
hexBrook.fig.suptitle('Distribution of Service Requests Across Brooklyn')
```

Out[23]:

```
Text(0.5, 0.98, 'Distribution of Service Requests Across Brooklyn')
```



Distribution of Service Requests Across Brooklyn

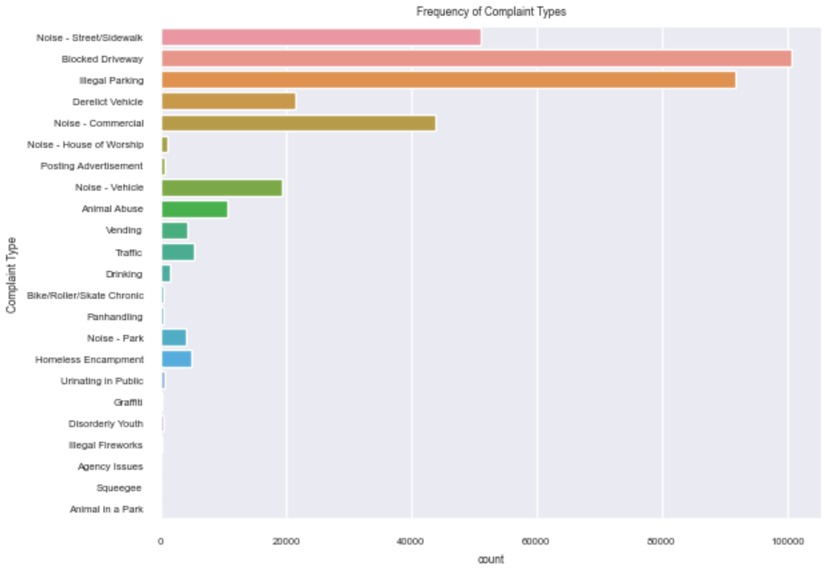
In [24]:

```
#3. find major types of complaints
#bar graph showing types of complaints

sns.countplot(data=raw, y='Complaint Type')
plt.title('Frequency of Complaint Types')
```

Out[24]:

```
Text(0.5, 1.0, 'Frequency of Complaint Types')
```



In [25]:

```
#checking frequency of various complaints in New York City in particular

nycraw = raw[raw['City'] == "NEW YORK"]
nycraw['Complaint Type'].value_counts()
```

Out[25]:

```
Noise - Street/Sidewalk      22245
Noise - Commercial           18686
Illegal Parking              14549
Noise - Vehicle               6294
Homeless Encampment           3060
Blocked Driveway              2705
Vending                       2638
Animal Abuse                  1941
Traffic                       1769
Noise - Park                  1243
Derelict Vehicle               695
Drinking                       321
Urinating in Public            264
Bike/Roller/Skate Chronic      254
Noise - House of Worship       222
Panhandling                    206
Disorderly Youth                81
Posting Advertisement           49
Illegal Fireworks               38
Graffiti                        25
Squeegee                         4
Name: Complaint Type, dtype: int64
```

In [26]:

```
#top 10 complaint types for all cities
raw['Complaint Type'].value_counts().nlargest(10)
```

Out[26]:

```
Blocked Driveway           100624
Illegal Parking             91716
Noise - Street/Sidewalk     51139
Noise - Commercial         43751
Derelict Vehicle           21518
Noise - Vehicle            19301
Animal Abuse               10530
Traffic                     5196
Homeless Encampment         4879
Vending                     4185
Name: Complaint Type, dtype: int64
```

In [27]:

```
#creating new dataframe with City as columns and Complaint Type as rows
#displays various complaint types in each city

df_new = pd.crosstab(index=raw["Complaint Type"], columns=raw["City"])
df_new
```

Out[27]:

City	ARVERNE	ASTORIA	Astoria	BAYSIDE	BELLEROSE	BREEZY POINT	BRONX	BROOKLYN	CAMBRIA HEIGHTS	CENTRAL PARK	...	SOUTH OZONE PARK	SOUTH RICHMOND HILL	SPRINGFIELD GARDENS	STATEN ISLAND	SU
mplaint Type																
gency Issues	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	
ıimal Abuse	46	170	0	53	15	2	1971	3191	15	0	...	74	40	42	786	
mal in a Park	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	
/Roller/Skate Chronic	0	16	0	0	1	0	22	124	0	0	...	1	1	0	10	
Blocked Driveway	50	3436	159	514	138	3	17062	36445	177	0	...	1202	1946	330	2845	
relict Vehicle	32	426	14	231	120	3	2402	6257	148	0	...	425	356	267	2184	
ɔrderly Youth	2	5	0	2	2	0	66	79	0	0	...	2	2	0	25	
Drinking	1	43	0	1	1	1	206	291	0	0	...	14	25	6	188	
Graffiti	1	4	0	3	0	0	15	60	0	0	...	2	0	0	6	
Homeless Encampment	4	32	0	2	1	0	275	948	6	0	...	5	12	7	77	
ɟal Fireworks	0	4	0	0	1	0	24	61	1	0	...	1	2	1	11	
legal Parking	62	1340	277	638	132	16	9889	33532	113	5	...	602	596	291	6224	
Noise - Commercial	2	1653	310	47	38	4	2944	13855	19	0	...	82	223	38	783	
se - House of Worship	14	21	0	3	1	0	90	389	2	0	...	5	3	1	18	
Noise - Park	2	64	0	4	1	0	548	1575	0	0	...	4	2	1	67	
Noise - 'eet/Sidewalk	29	409	145	17	13	1	9144	13982	29	105	...	108	93	42	885	
ɔise - Vehicle	10	236	0	24	11	1	3556	5965	100	0	...	97	93	48	424	
Panhandling	1	2	0	0	1	0	20	49	0	0	...	0	0	2	13	
Posting dvertisement	0	3	0	0	1	0	18	58	0	0	...	1	0	2	516	
Squeegee	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	
Traffic	1	60	0	9	9	0	427	1258	7	0	...	36	12	12	229	
Urinating in Public	1	10	0	0	1	0	54	155	0	0	...	2	1	3	19	
Vending	1	57	0	2	0	0	433	575	0	0	...	5	24	1	25	

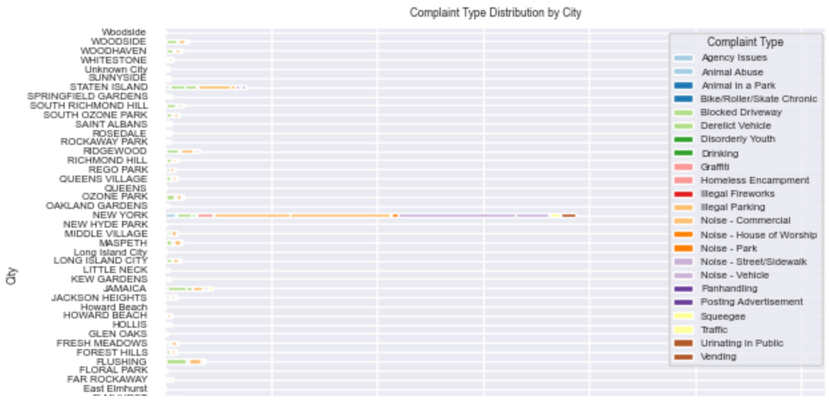
ıws × 54 columns

In [28]:

```
#graph showing the different types of complaints in each City (complaints in color)

citytype= pd.crosstab(index=raw["City"], columns=raw["Complaint Type"])
citytype.plot(kind='barh', stacked=True, colormap="Paired")
plt.title('Complaint Type Distribution by City')
```

Out[28]: Text(0.5, 1.0, 'Complaint Type Distribution by City')



In [29]:

```
#create new dataframe of Elapsed Time (sec), City, Complaint Type
#intention is to check if avg response time varies across complaint types

responseData = raw[['City', 'Complaint Type', 'Elapsed Time (sec)']]
responseData
```

Out[29]:

	City	Complaint Type	Elapsed Time (sec)
0	NEW YORK	Noise - Street/Sidewalk	3330.0
1	ASTORIA	Blocked Driveway	5233.0
2	BRONX	Blocked Driveway	17494.0
3	BRONX	Illegal Parking	27927.0
4	ELMHURST	Illegal Parking	12464.0
...	...	...	...
364553	WOODHAVEN	Illegal Parking	37067.0
364554	BRONX	Noise - Vehicle	8434.0
364555	NEW YORK	Noise - Street/Sidewalk	1143.0
364556	BRONX	Blocked Driveway	9653.0
364557	SOUTH OZONE PARK	Blocked Driveway	10020.0

362177 rows × 3 columns

In [30]:

```
#group Complaint Type by City and show average Elapsed Time to complete service request
pd.options.display.max_rows = None
responseData.groupby(['City','Complaint Type']).mean().astype(str)
```

Out[30]:

		Elapsed Time (sec)
City	**Complaint Type**	
ARVERNE	**Animal Abuse**	8399.195652173914
	Blocked Driveway	8318.84
	Derelict Vehicle	11394.0
	Disorderly Youth	12928.5
	Drinking	859.0
	Graffiti	5508.0
	Homeless Encampment	6541.25
	Illegal Parking	8406.08064516129
	Noise - Commercial	8234.0
	Noise - House of Worship	6653.428571428572

In [31]:

```
#creating new DataFrame to just show avg elapsed time for each complaint type
complainttime = responseData.groupby(['Complaint Type'], as_index=False).mean()
complainttime
```

Out[31]:

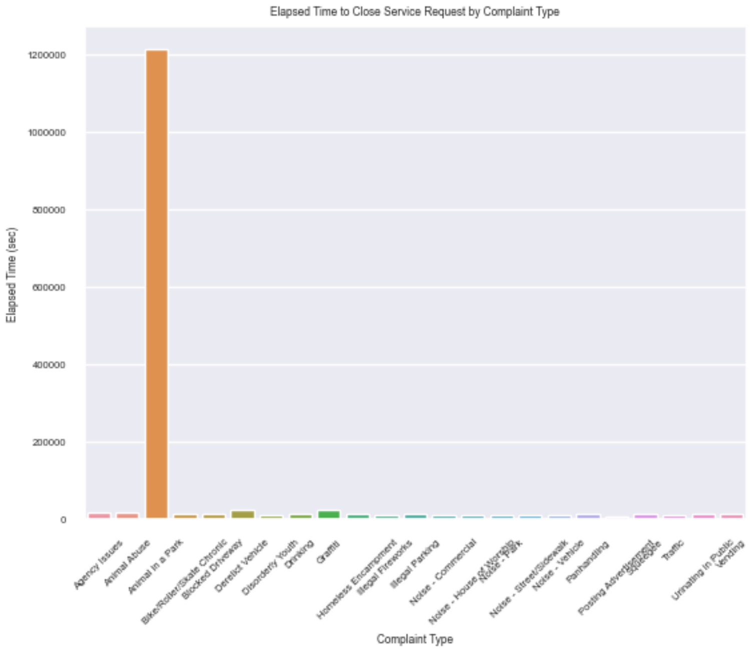
	Complaint Type	Elapsed Time (sec)
0	Agency Issues	1.828912e+04
1	Animal Abuse	1.803256e+04
2	Animal in a Park	1.212634e+06
3	Bike/Roller/Skate Chronic	1.312369e+04
4	Blocked Driveway	1.623252e+04
5	Derelict Vehicle	2.535960e+04
6	Disorderly Youth	1.236375e+04
7	Drinking	1.382130e+04
8	Graffiti	2.327634e+04
9	Homeless Encampment	1.545138e+04
10	Illegal Fireworks	1.011348e+04
11	Illegal Parking	1.565044e+04
12	Noise - Commercial	1.108576e+04
13	Noise - House of Worship	1.139109e+04
14	Noise - Park	1.222606e+04
15	Noise - Street/Sidewalk	1.223130e+04
16	Noise - Vehicle	1.256180e+04
17	Panhandling	1.585355e+04
18	Posting Advertisement	7.286256e+03
19	Squeegee	1.456025e+04
20	Traffic	1.230912e+04
21	Urinating in Public	1.295929e+04
22	Vending	1.436628e+04

In [32]:

```
#graphing the above DataFrame to compare average time to resolve a request between complaint types
sns.barplot(data=complainttime, x='Complaint Type', y='Elapsed Time (sec)')
plt.xticks(rotation=45)
plt.ticklabel_format(style='plain',axis='y')
plt.title('Elapsed Time to Close Service Request by Complaint Type')
```

Out[32]:

```
Text(0.5, 1.0, 'Elapsed Time to Close Service Request by Complaint Type')
```



In [33]:

```
#remove Animal in Park Complaint Type to re-check distribution
complaintime2 = complaintime.drop([2])
complaintime2
```

Out[33]:

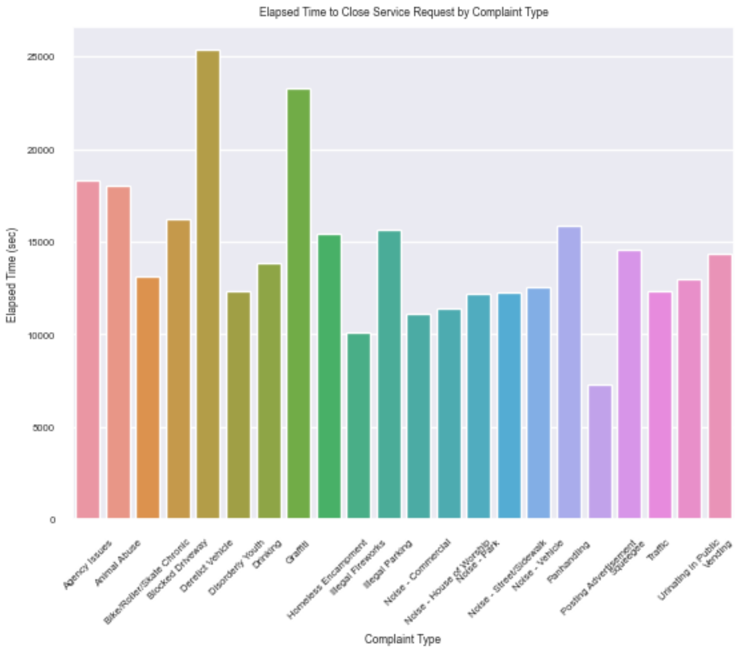
	Complaint Type	Elapsed Time (sec)
0	Agency Issues	18289.125000
1	Animal Abuse	18032.556030
3	Bike/Roller/Skate Chronic	13123.688421
4	Blocked Driveway	16232.521516
5	Derelict Vehicle	25359.600102
6	Disorderly Youth	12363.749206
7	Drinking	13821.300570
8	Graffiti	23276.343949
9	Homeless Encampment	15451.384505
10	Illegal Fireworks	10113.482558
11	Illegal Parking	15650.435671
12	Noise - Commercial	11085.760531
13	Noise - House of Worship	11391.087079
14	Noise - Park	12226.055515
15	Noise - Street/Sidewalk	12231.295411
16	Noise - Vehicle	12561.800010
17	Panhandling	15853.550769
18	Posting Advertisement	7286.256259
19	Squeegee	14560.250000
20	Traffic	12309.120092
21	Urinating in Public	12959.293292
22	Vending	14366.278375

In [34]:

```
#graph distribution wiht Animal in Park dropped
sns.barplot(data=complaintime2, x='Complaint Type', y='Elapsed Time (sec)')
plt.xticks(rotation=45)
plt.ticklabel_format(style='plain',axis='y')
plt.title('Elapsed Time to Close Service Request by Complaint Type')
```

Out[34]:

```
Text(0.5, 1.0, 'Elapsed Time to Close Service Request by Complaint Type')
```



In [35]:

```
#comparison of means test (ANOVA) for all Complaint Types
stats.f_oneway(responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Agency Issues'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Animal Abuse'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Animal in a Park'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Bike/Roller/Skate Chronic'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Blocked Driveway'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Derelict Vehicle'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Disorderly Youth'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Drinking'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Graffiti'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Homeless Encampment'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Illegal Fireworks'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Illegal Parking'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Commercial'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - House of Worship'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Park'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Street/Sidewalk'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Vehicle'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Panhandling'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Posting Advertisement'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Squeegee'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Traffic'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Urinating in Public'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Vending']
              )
```

Out[35]:

```
F_onewayResult(statistic=565.2615700417628, pvalue=0.0)
```

In [36]:

```
#ANOVA interpretation: reject null hypothesis because p-value<0.05. At least one Complaint Type group differs significantly.


#Kruskal-Wallis H Test
stats.kruskal(responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Agency Issues'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Animal Abuse'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Animal in a Park'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Bike/Roller/Skate Chronic'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Blocked Driveway'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Derelict Vehicle'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Disorderly Youth'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Drinking'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Graffiti'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Homeless Encampment'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Illegal Fireworks'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Illegal Parking'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Commercial'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - House of Worship'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Park'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Street/Sidewalk'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Noise - Vehicle'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Panhandling'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Posting Advertisement'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Squeegee'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Traffic'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Urinating in Public'],
               responseData['Elapsed Time (sec)'][responseData['Complaint Type']=='Vending']
              )
```

Out[36]:

```
KruskalResult(statistic=11988.269402358468, pvalue=0.0)
```

In [37]:

```
#results of Kruskal: reject hypothesis because p-value<0.05. One or more sample distributions' medians is not equal.
```

In [70]:

```
#for loop to run an ANOVA on all Complaint Types separately against the mean of all

uniqueTypes=responseData['Complaint Type'].unique()

for t in uniqueTypes:
    print(t)
    s, p = stats.f_oneway(responseData['Elapsed Time (sec)'][responseData['Complaint Type'] == t],
                  responseData['Elapsed Time (sec)'])
    if p<0.05:
        print("The p-value for ", t, "is: ",p)
    else:
        print("The ANOVA for ", t, "is statistically insignificant.")
```

```
Noise - Street/Sidewalk
The p-value for  Noise - Street/Sidewalk is:  1.8816940468126975e-187
Blocked Driveway
The p-value for  Blocked Driveway is:  1.6159916194237022e-52
Illegal Parking
The p-value for  Illegal Parking is:  4.154525537889026e-12
Derelict Vehicle
The p-value for  Derelict Vehicle is:  0.0
Noise - Commercial
The p-value for  Noise - Commercial is:  4.9e-322
Noise - House of Worship
The p-value for  Noise - House of Worship is:  8.432607319794917e-09
Posting Advertisement
The p-value for  Posting Advertisement is:  4.346909898855285e-22
Noise - Vehicle
The p-value for  Noise - Vehicle is:  1.8281897697741582e-61
Animal Abuse
The p-value for  Animal Abuse is:  1.4394553240105054e-43
Vending
The p-value for  Vending is:  0.022509428760841988
Traffic
The p-value for  Traffic is:  1.5596037810939687e-21
Drinking
The p-value for  Drinking is:  0.021977339864342756
Bike/Roller/Skate Chronic
The p-value for  Bike/Roller/Skate Chronic is:  0.03995844859860901
Panhandling
The ANOVA for  Panhandling is statistically insignificant.
Noise - Park
The p-value for  Noise - Park is:  2.646115578080773e-18
Homeless Encampment
The ANOVA for  Homeless Encampment is statistically insignificant.
Urinating in Public
The p-value for  Urinating in Public is:  0.00980519043408918
Graffiti
The p-value for  Graffiti is:  1.2661371844340123e-06
Disorderly Youth
The p-value for  Disorderly Youth is:  0.020771877023639477
Illegal Fireworks
The p-value for  Illegal Fireworks is:  0.0018899660292152176
Agency Issues
The ANOVA for  Agency Issues is statistically insignificant.
Squeegee
The ANOVA for  Squeegee is statistically insignificant.
Animal in a Park
The p-value for  Animal in a Park is:  0.0
```

In [80]:

```
#for loop to run a Kruskal on all Complaint Types separately against the mean of all

for t in uniqueTypes:
    print(t)
    s, p = stats.kruskal(responseData['Elapsed Time (sec)'][responseData['Complaint Type'] == t],
                  responseData['Elapsed Time (sec)'])
    if p<0.05:
        print("The p-value for ", t, "is: ",p)
    else:
        print("The Kruskal-Wallis H Test for ", t, "is statistically insignificant.")
```

```
Noise - Street/Sidewalk
The p-value for  Noise - Street/Sidewalk is:  0.0
Blocked Driveway
The p-value for  Blocked Driveway is:  0.0
Illegal Parking
The p-value for  Illegal Parking is:  9.741214904521316e-55
Derelict Vehicle
The p-value for  Derelict Vehicle is:  0.0
Noise - Commercial
The p-value for  Noise - Commercial is:  0.0
Noise - House of Worship
The p-value for  Noise - House of Worship is:  6.930181176361173e-24
Posting Advertisement
The p-value for  Posting Advertisement is:  2.803234449006454e-70
Noise - Vehicle
The p-value for  Noise - Vehicle is:  8.405737110084953e-106
Animal Abuse
The p-value for  Animal Abuse is:  8.357777072833736e-81
Vending
The Kruskal-Wallis H Test for  Vending is statistically insignificant.
Traffic
The p-value for  Traffic is:  1.2016530910591436e-85
Drinking
The Kruskal-Wallis H Test for  Drinking is statistically insignificant.
Bike/Roller/Skate Chronic
The p-value for  Bike/Roller/Skate Chronic is:  0.006662068093053303
Panhandling
The Kruskal-Wallis H Test for  Panhandling is statistically insignificant.
Noise - Park
The p-value for  Noise - Park is:  5.809958778086615e-33
Homeless Encampment
The p-value for  Homeless Encampment is:  0.0011887571897201124
Urinating in Public
The p-value for  Urinating in Public is:  4.219443417745079e-05
Graffiti
The p-value for  Graffiti is:  1.0746066629867494e-08
Disorderly Youth
The p-value for  Disorderly Youth is:  0.027141055353555394
Illegal Fireworks
The p-value for  Illegal Fireworks is:  3.813645042289103e-09
Agency Issues
The Kruskal-Wallis H Test for  Agency Issues is statistically insignificant.
Squeegee
The Kruskal-Wallis H Test for  Squeegee is statistically insignificant.
Animal in a Park
The Kruskal-Wallis H Test for  Animal in a Park is statistically insignificant.
```

In [ ]: