In [1]:
```python
# importing libraries

import pandas as pd
import numpy as np
```

In [40]:
```python
import seaborn as sns
pd.options.mode.chained_assignment = None
```

In [3]:
```python
import matplotlib.pyplot as plt
%matplotlib inline
```

Import a 311 NYC service request.

In [4]:
```python
df = pd.read_csv('311_dataset.csv',low_memory=False)
```

In [5]:
```python
df.head()
```

Out[5]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name
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
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
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
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
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

5 rows × 53 columns

In [6]:
```python
# shape of the dataframe
df.shape
```

Out[6]:
```
(300698, 53)
```

In [7]:
```python
# columns in df
df.columns
```

Out[7]:
```
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 [8]:
```python
# converting the columns into lower case and replacing the space with '_'.

df.columns = df.columns.str.lower().str.replace(' ','_')
```

In [9]:
```python
# checking for NaN or null values in df
df.isna().sum()
```

Out[9]:
```
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
```

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. (Hint: Explore the package/module datetime)

In [10]:
```python
df['closed_date'] = pd.to_datetime(df['closed_date'])
```

In [11]:
```python
df['created_date'] = pd.to_datetime(df['created_date'])
```

In [12]:
```python
requested_closing_time = df['closed_date'] - df['created_date']
```

In [13]:
```python
df['requested_closing_time'] = requested_closing_time
```

In [14]:
```python
# converting the date difference into minutes

requested_closing_time_min = df['requested_closing_time']/np.timedelta64(1,'m')
```

In [15]:
```python
df['requested_closing_time_min'] = requested_closing_time_min
```

In [16]:
```python
df.head()
```

Out[16]:

	unique_key	created_date	closed_date	agency	agency_name	complaint_type	descriptor	location_type	incident_zip	incident_addre
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 VERMILY AVEN
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 AVEN
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 VALENTI AVEN
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 BAISL AVEN
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 RO

5 rows × 55 columns

In [17]:
```python
df.columns
```

Out[17]:
```
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',
       'requested_closing_time', 'requested_closing_time_min'],
      dtype='object')
```

In [18]:
```python
# creating new dataframe

new_df = df[['unique_key','created_date','closed_date','agency', 'agency_name','complaint_type', 'descr
iptor', 'location_type',
          'address_type','city','facility_type', 'status', 'due_date','borough', 'location',
       'requested_closing_time', 'requested_closing_time_min']]
```

In [19]:
```python
new_df.head()
```

Out[19]:

	unique_key	created_date	closed_date	agency	agency_name	complaint_type	descriptor	location_type	address_type	city
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	ADDRESS	NEW YORK
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	ADDRESS	ASTORIA
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	ADDRESS	BRONX
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	ADDRESS	BRONX
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	ADDRESS	ELMHURST

Converting the series city and borough into lower case to avoid redudancies

In [20]:
```python
new_df['city'] = new_df['city'].str.lower()
pd.options.mode.chained_assignment = None
```

In [21]:
```python
new_df['borough'] = new_df['borough'].str.lower()
pd.options.mode.chained_assignment = None
```

In [22]:
```python
new_df.head()
new_df.shape
```

Out[22]:
```
(300698, 17)
```

In [23]:
```python
new_df.duplicated().sum()
```

Out[23]:
```
0
```

In [24]:
```python
new_df.isnull().sum()
```

Out[24]:
```
unique_key                        0
created_date                      0
closed_date                    2164
agency                            0
agency_name                       0
complaint_type                    0
descriptor                     5914
location_type                   131
address_type                   2815
city                           2614
facility_type                  2171
status                            0
due_date                          3
borough                           0
location                       3540
requested_closing_time         2164
requested_closing_time_min     2164
dtype: int64
```

In [25]:
```python
pd.set_option('mode.chained_assignment', None)
new_df.dropna(inplace=True)
```

In [26]:
```python
new_df.isnull().sum()
```

Out[26]:
```
unique_key                    0
created_date                  0
closed_date                   0
agency                        0
agency_name                   0
complaint_type                0
descriptor                    0
location_type                 0
address_type                  0
city                          0
facility_type                 0
status                        0
due_date                      0
borough                       0
location                      0
requested_closing_time        0
requested_closing_time_min    0
dtype: int64
```

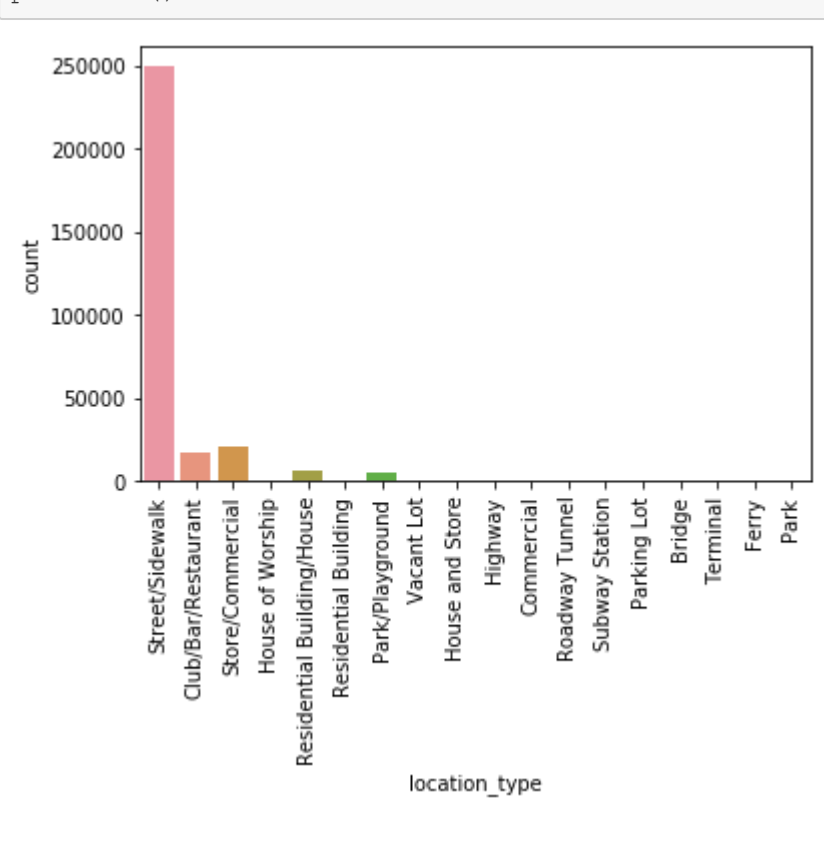
In [27]:
```python
new_df.dtypes
```

Out[27]:
```
unique_key                              int64
created_date                   datetime64[ns]
closed_date                    datetime64[ns]
agency                                 object
agency_name                            object
complaint_type                         object
descriptor                             object
location_type                          object
address_type                           object
city                                   object
facility_type                          object
status                                 object
due_date                               object
borough                                object
location                               object
requested_closing_time        timedelta64[ns]
requested_closing_time_min            float64
dtype: object
```

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.

1)

In [28]:
```python
df['location_type'].value_counts()
plot=sns.countplot(df['location_type'])
plot.set_xticklabels(plot.get_xticklabels(),rotation=90)
plt.show()
```

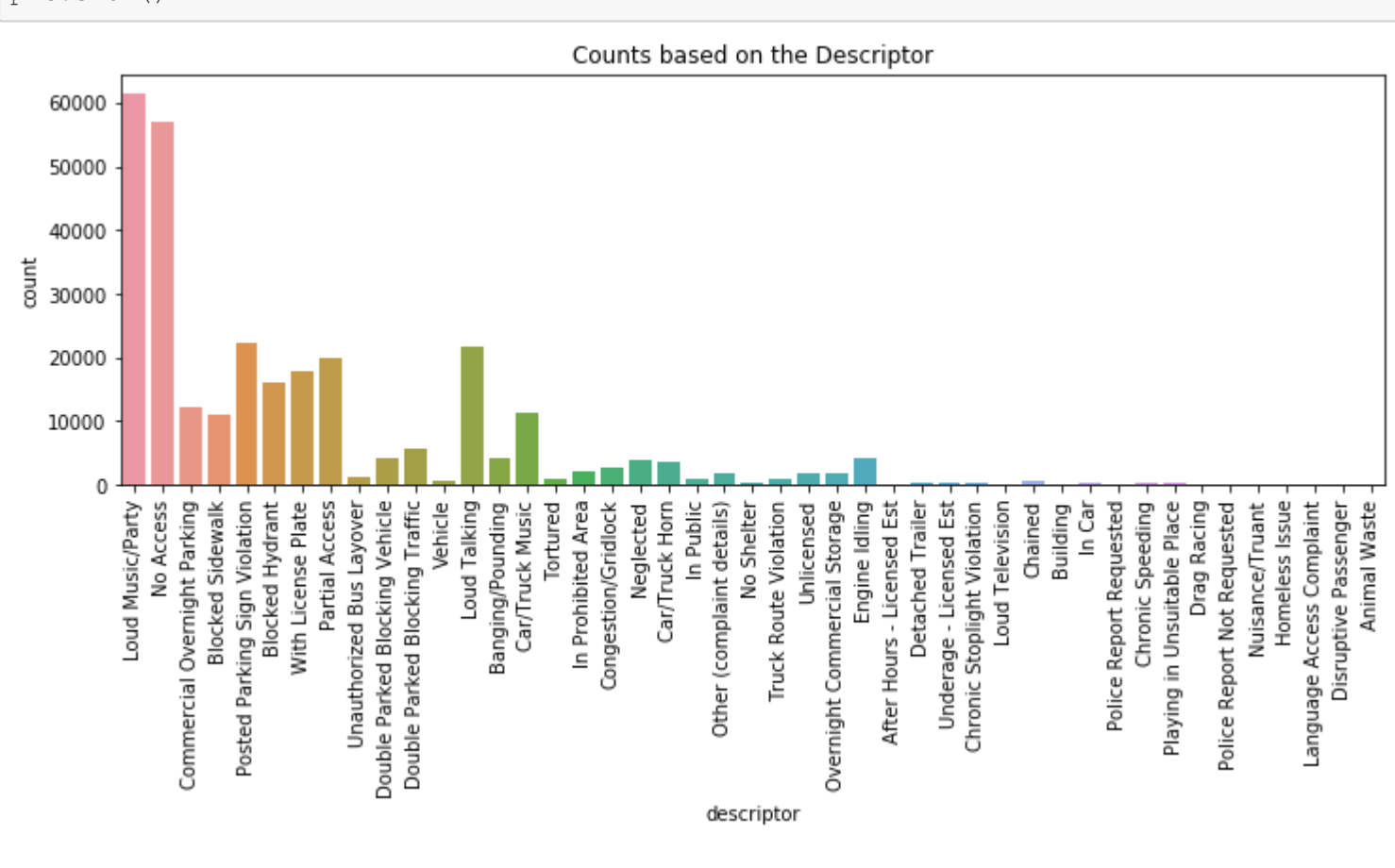


Insight :

```
- Street/ SideWalk is the most used Location Type by New York people. From the plot, we can see
 it out-counts most other
types
```

2)

In [29]:
```python
plt.figure(figsize=(12,4))
plot2=sns.countplot(df['descriptor'])
plt.title('Counts based on the Descriptor')
plot2.set_xticklabels(plot2.get_xticklabels(),rotation=90)
plt.show()
```

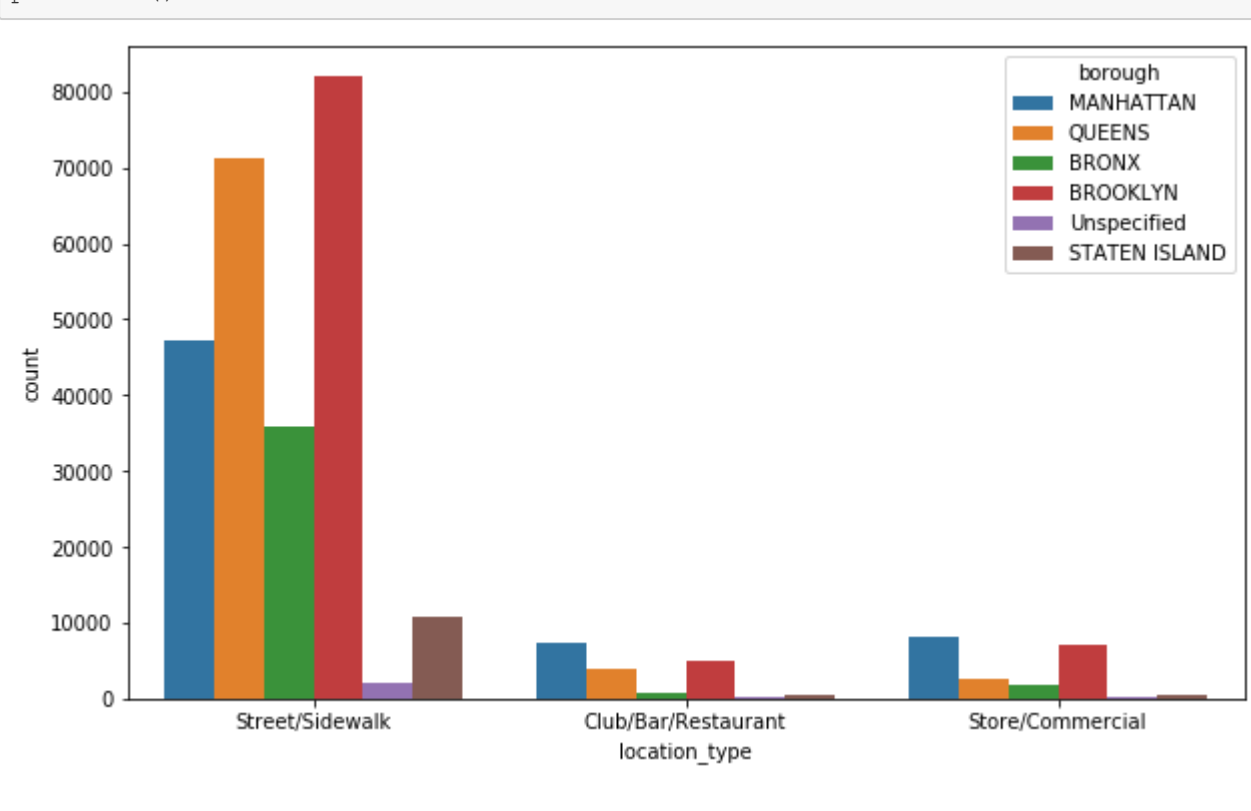


Insights:

```
- "Descriptor" variable plotted here is based on the Complaint type
- From the above, it is seen that Loud Music/Party is highest registered count by the people aga
inst others.
- It comes under the "Noise" Complaint type.
```

3)

In [30]:
```python
plt.figure(figsize=(10,6))
sns.countplot(df.loc[df['location_type'].isin(['Street/Sidewalk','Store/Commercial','Club/Bar/Restauran
t'])]
            ['location_type'],data=df,hue='borough')
plt.show()
```

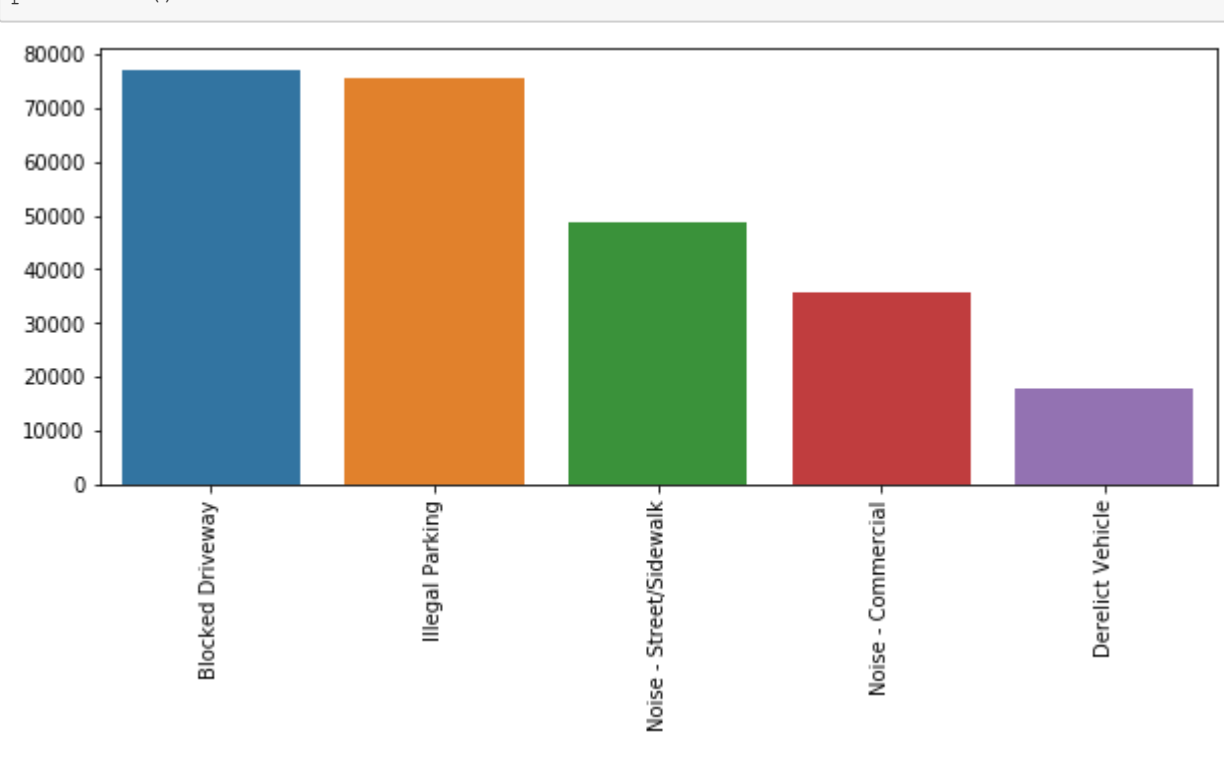


Conclusion:

```
- The plot is used to measure of usage count of location type's in the Cities of Newyork
- From above, we can infer that, Street/SideWalks of BROOKLYN is most used by the people of Newy
ork
- Footfall in Sidewalks is higher than anyother compared in overall Borough.
```

4)

In [31]:
```python
plt.figure(figsize=(10,4))
series=df['complaint_type'].value_counts()
series.nlargest().index
plot4=sns.barplot(x=series.nlargest().index,y=series.nlargest().values)
plot4.set_xticklabels(plot4.get_xticklabels(),rotation =90)
plt.show()
```



Insights :

```
- The above graph is used to find out which complaint type has been registered high in count.
- As visualised, since Driveway Block is has high count, it is the problem faced most, by the pe
ople around.
```

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

In [32]:
```python
df.pivot_table(index = 'complaint_type',
                    columns='city',
                    values= 'requested_closing_time_min',
                    aggfunc= lambda x : x.mean(),
                     fill_value='-')
```

Out[32]:

city / complaint_type	ARVERNE	ASTORIA	Astoria	BAYSIDE	BELLEROSE	BREEZY POINT	BRONX	BROOKLYN	CAMBRIA HEIGHTS	CENTRAL PARK	...	SAIN ALBAN
Animal Abuse	129.218	300.01	-	196.471	763.481	156.808	440.13	289.95	681.35	-	...	442.3
Animal in a Park	-	-	-	-	-	-	-	-	-	-	...	
Bike/Roller/Skate Chronic	-	104.359	-	-	294	-	207.538	300.282	-	-	...	
Blocked Driveway	151.558	288.967	294.878	153.784	605.965	79.6611	375.706	264.647	461.759	-	...	281.2
Derelict Vehicle	178.093	581.355	374.169	201.587	1030.09	428.711	553.654	356.849	966.647	-	...	354.
Disorderly Youth	215.475	174.156	-	178.383	111	-	254.304	249.031	-	-	...	105.
Drinking	14.3167	283.345	-	114	235.083	158	347.615	212.434	-	-	...	57.68
Graffiti	92	845.704	-	273.094	-	-	533.943	494.592	-	-	...	
Homeless Encampment	108.888	295.055	-	172.533	2348	-	446.514	281.689	1367.37	-	...	454.2
Illegal Fireworks	-	166.408	-	-	400	-	336.492	140.375	91.8333	-	...	
Illegal Parking	138.982	290.003	282.677	153.76	492.188	236.533	394.851	256.334	674.606	275.658	...	251.0
Noise - Commercial	137.233	187.982	212.513	134.02	404.446	152.371	281.841	179.166	228.601	-	...	186.0
Noise - House of Worship	93.7318	121.333	-	212.092	131.817	-	273.573	184.192	158.458	-	...	278.
Noise - Park	77	179.707	-	196.354	84.85	-	281.912	189.032	-	-	...	49.21
Noise - Street/Sidewalk	119.544	207.07	222.794	91.86	544.094	60	313.579	197.76	276.483	196.016	...	201.9
Noise - Vehicle	111.593	210.551	-	102.505	154.99	79.4667	333.648	197.007	415.026	-	...	207.5
Panhandling	62	69	-	-	449	-	852.824	258.989	-	-	...	
Posting Advertisement	-	352.2	-	-	135.6	-	207.72	201.657	-	-	...	
Squeegee	-	-	-	-	-	-	-	-	-	-	...	
Traffic	-	324.611	-	91.6167	345.531	-	295.421	186.776	512.803	-	...	237.9
Urinating in Public	41.5167	277.543	-	-	452.483	-	323.406	233.959	-	-	...	390
Vending	29	296.125	-	112.675	-	-	409.542	271.686	-	-	...	143.3

22 rows × 53 columns

For the below statements you need to state the Null and Alternate and then provide a statistical test to accept or reject the Null Hypothesis along with the corresponding 'p-value'.

Whether the average response time across complaint types is similar or not (overall)

Hypothesis:

```
- Ho: Overall ComplaintType and Average Response Time doesn't have any significant difference
- Ha: Atleast one of the ComplaintType's and Avg. Response Time has significant difference
```

In [33]:
```python
import scipy.stats as stats
```

In [34]:
```python
top = new_df['complaint_type'].value_counts().index
top
```

Out[34]:
```
Index(['Blocked Driveway', 'Illegal Parking', 'Noise - Street/Sidewalk',
       'Noise - Commercial', 'Derelict Vehicle', 'Noise - Vehicle',
       'Animal Abuse', 'Traffic', 'Noise - Park', 'Vending', 'Drinking',
       'Noise - House of Worship', 'Posting Advertisement', 'Disorderly Youth',
       'Graffiti'],
      dtype='object')
```

In [35]:
```python
from statsmodels.formula.api import ols
import statsmodels.api as sm
```

In [36]:
```python
sample = new_df[new_df['complaint_type'].isin(top)]
mod = ols('requested_closing_time_min ~ complaint_type', data = sample).fit()
sm.stats.anova_lm(mod)
```

Out[36]:

	df	sum_sq	mean_sq	F	PR(>F)
complaint_type	14.0	1.043265e+09	7.451890e+07	578.768616	0.0
Residual	290869.0	3.745061e+10	1.287542e+05	NaN	NaN

F-stat

Conclusion:

```
- From the F-stat, the p_value < 0.05
- Hence, we reject null hypothesis
```

Are the type of complaint or service requested and location related?¶

Hypothesis:

```
- Ho: The complaint or service requested and location are not related/ independent
- Ha: They are related/ dependent
```

In [37]:
```python
topall = df['complaint_type'].value_counts().index
sample = df[df['complaint_type'].isin(topall)]
ct = pd.crosstab(sample['complaint_type'], sample.location)
ct
```

Out[37]:

location / complaint_type	(40.49913462101514, -74.24348482977875)	(40.49967332981336, -74.2379063249761)	(40.49994886080869, -74.23740031497493)	(40.49999700116009, -74.23801175120917)	(40.50002168207532, -74.23802262609722)	(40.5000491077 -74.23803351
Animal Abuse	0	0	0	0	0	
Bike/Roller/Skate Chronic	0	0	0	0	0	
Blocked Driveway	0	0	0	0	0	
Derelict Vehicle	0	1	0	0	0	
Disorderly Youth	0	0	0	0	0	
Drinking	0	0	0	0	0	
Graffiti	0	0	0	0	0	
Homeless Encampment	0	0	0	0	0	
Illegal Fireworks	0	0	0	0	0	
Illegal Parking	1	0	1	1	1	
Noise - Commercial	0	0	0	0	0	
Noise - House of Worship	0	0	0	0	0	
Noise - Park	0	0	0	0	0	
Noise - Street/Sidewalk	0	0	0	0	0	
Noise - Vehicle	0	0	0	0	0	
Panhandling	0	0	0	0	0	
Posting Advertisement	0	0	0	0	0	
Squeegee	0	0	0	0	0	
Traffic	0	0	0	0	0	
Urinating in Public	0	0	0	0	0	
Vending	0	0	0	0	0	

21 rows × 126048 columns

In [38]:
```python
chisquare , p , df, et = stats.chi2_contingency(ct)
```

In [39]:
```python
print('Results : statistic={:.4f}, pvalue={}'.format(chisquare,p))
```

```
Results : statistic=4161473.2250, pvalue=0.0
```

Conclusion:

```
- Here, p_value < 0.05
- Thus " we reject null hypothesis"
```