USA-Accidents Exploratory Data Analysis Project using Python

Exploratory Data Analysis (EDA) deals with the main components of a data set using various statistical approaches. Data visualisation techniques are frequently used to visualise the numbers discovered during exploration.

The dataset is picked from Kaggle.com, contains US accidents data across its 49 States from February 2016 and December 2020. The dataset has 29,06,610 rows and 47 columns.

Here, In this Project, we will be analysing major factors which are responsible for accidents. And, at the end, we will pressent some useful insights from our analysis, which will be helpful to minimize the accidents in US.

Download dataset from kaggle.com

In [1]:
```
!pip install opendatasets
```

```
Requirement already satisfied: opendatasets in c:\users\shivam\appdata\loca
l\programs\python\python311\lib\site-packages (0.1.22)
Requirement already satisfied: tqdm in c:\users\shivam\appdata\local\progra
ms\python\python311\lib\site-packages (from opendatasets) (4.65.0)
Requirement already satisfied: kaggle in c:\users\shivam\appdata\local\prog
rams\python\python311\lib\site-packages (from opendatasets) (1.5.13)
Requirement already satisfied: click in c:\users\shivam\appdata\local\progr
ams\python\python311\lib\site-packages (from opendatasets) (8.1.3)
Requirement already satisfied: colorama in c:\users\shivam\appdata\local\pr
ograms\python\python311\lib\site-packages (from click->opendatasets) (0.4.
6)
Requirement already satisfied: six>=1.10 in c:\users\shivam\appdata\local\p
rograms\python\python311\lib\site-packages (from kaggle->opendatasets) (1.1
6.0)
Requirement already satisfied: certifi in c:\users\shivam\appdata\local\pro
grams\python\python311\lib\site-packages (from kaggle->opendatasets) (2022.
12.7)
Requirement already satisfied: python-dateutil in c:\users\shivam\appdata\l
ocal\programs\python\python311\lib\site-packages (from kaggle->opendataset
s) (2.8.2)
Requirement already satisfied: requests in c:\users\shivam\appdata\local\pr
ograms\python\python311\lib\site-packages (from kaggle->opendatasets) (2.2
8.2)
Requirement already satisfied: python-slugify in c:\users\shivam\appdata\lo
cal\programs\python\python311\lib\site-packages (from kaggle->opendatasets)
(8.0.1)
Requirement already satisfied: urllib3 in c:\users\shivam\appdata\local\pro
grams\python\python311\lib\site-packages (from kaggle->opendatasets) (1.26.
15)
Requirement already satisfied: text-unidecode>=1.3 in c:\users\shivam\appda
ta\local\programs\python\python311\lib\site-packages (from python-slugify->
kaggle->opendatasets) (1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\shivam
\appdata\local\programs\python\python311\lib\site-packages (from requests->
kaggle->opendatasets) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\shivam\appdata\loca
l\programs\python\python311\lib\site-packages (from requests->kaggle->opend
atasets) (3.4)
[notice] A new release of pip available: 22.3.1 -> 23.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [2]:
```
import opendatasets as od
```

In [3]:
```
dataset = "https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents"
```

In [4]:
```
od.download(dataset)
```

```
Skipping, found downloaded files in ".\us-accidents" (use force=True to for
ce download)
```

In [5]:
```
import os
```

In [6]:
```
os.listdir('us-accidents')
```

Out[6]: ['US_Accidents_Dec21_updated.csv']

Data Cleaning & Processing

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

In [8]:
```
# dataset of 2.85 millions users can take upto 30 secs. to load.
df = pd.read_csv(r"C:\Users\Shivam\Desktop\Python\Jupyter_notebook_projects\
df
```

Out[8]:

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lr
0	A-1	3	2016-02-08 00:37:08	2016-02-08 06:37:08	40.108910	-83.092860	40.112060	-83.0318
1	A-2	2	2016-02-08 05:56:20	2016-02-08 11:56:20	39.865420	-84.062800	39.865010	-84.0487
2	A-3	2	2016-02-08 06:15:39	2016-02-08 12:15:39	39.102660	-84.524680	39.102090	-84.5239
3	A-4	2	2016-02-08 06:51:45	2016-02-08 12:51:45	41.062130	-81.537840	41.062170	-81.5354
4	A-5	3	2016-02-08 07:53:43	2016-02-08 13:53:43	39.172393	-84.492792	39.170476	-84.5017
...	...	...	...	...	...	...	...	
2845337	A-2845338	2	2019-08-23 18:03:25	2019-08-23 18:32:01	34.002480	-117.379360	33.998880	-117.3709
2845338	A-2845339	2	2019-08-23 19:11:30	2019-08-23 19:38:23	32.766960	-117.148060	32.765550	-117.1536
2845339	A-2845340	2	2019-08-23 19:00:21	2019-08-23 19:28:49	33.775450	-117.847790	33.777400	-117.8572
2845340	A-2845341	2	2019-08-23 19:00:21	2019-08-23 19:29:42	33.992460	-118.403020	33.983110	-118.3956
2845341	A-2845342	2	2019-08-23 18:52:06	2019-08-23 19:21:31	34.133930	-117.230920	34.137360	-117.2393

2845342 rows × 47 columns

In [9]:
```
df.head(3)
```

Out[9]:

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(mi)
0	A-1	3	2016-02-08 00:37:08	2016-02-08 06:37:08	40.10891	-83.09286	40.11206	-83.03187	3.230
1	A-2	2	2016-02-08 05:56:20	2016-02-08 11:56:20	39.86542	-84.06280	39.86501	-84.04873	0.747
2	A-3	2	2016-02-08 06:15:39	2016-02-08 12:15:39	39.10266	-84.52468	39.10209	-84.52396	0.055

3 rows × 47 columns

In [15]:
```
dataf = pd.DataFrame(df)
dataf
```

Out[15]:

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lr
0	A-1	3	2016-02-08 00:37:08	2016-02-08 06:37:08	40.108910	-83.092860	40.112060	-83.0318
1	A-2	2	2016-02-08 05:56:20	2016-02-08 11:56:20	39.865420	-84.062800	39.865010	-84.0487
2	A-3	2	2016-02-08 06:15:39	2016-02-08 12:15:39	39.102660	-84.524680	39.102090	-84.5239
3	A-4	2	2016-02-08 06:51:45	2016-02-08 12:51:45	41.062130	-81.537840	41.062170	-81.5354
4	A-5	3	2016-02-08 07:53:43	2016-02-08 13:53:43	39.172393	-84.492792	39.170476	-84.5017
...	...	...	...	...	...	...	...	
2845337	A-2845338	2	2019-08-23 18:03:25	2019-08-23 18:32:01	34.002480	-117.379360	33.998880	-117.3709
2845338	A-2845339	2	2019-08-23 19:11:30	2019-08-23 19:38:23	32.766960	-117.148060	32.765550	-117.1536
2845339	A-2845340	2	2019-08-23 19:00:21	2019-08-23 19:28:49	33.775450	-117.847790	33.777400	-117.8572
2845340	A-2845341	2	2019-08-23 19:00:21	2019-08-23 19:29:42	33.992460	-118.403020	33.983110	-118.3956
2845341	A-2845342	2	2019-08-23 18:52:06	2019-08-23 19:21:31	34.133930	-117.230920	34.137360	-117.2393

2845342 rows × 47 columns

In [16]:
```
df.columns
```

Out[16]:
```
Index(['ID', 'Severity', 'Start_Time', 'End_Time', 'Start_Lat', 'Start_Ln
g',
       'End_Lat', 'End_Lng', 'Distance(mi)', 'Description', 'Number', 'Stre
et',
       'Side', 'City', 'County', 'State', 'Zipcode', 'Country', 'Timezone',
       'Airport_Code', 'Weather_Timestamp', 'Temperature(F)', 'Wind_Chill
(F)',
       'Humidity(%)', 'Pressure(in)', 'Visibility(mi)', 'Wind_Direction',
       'Wind_Speed(mph)', 'Precipitation(in)', 'Weather_Condition', 'Amenit
y',
       'Bump', 'Crossing', 'Give_Way', 'Junction', 'No_Exit', 'Railway',
       'Roundabout', 'Station', 'Stop', 'Traffic_Calming', 'Traffic_Signa
l',
       'Turning_Loop', 'Sunrise_Sunset', 'Civil_Twilight', 'Nautical_Twilig
ht',
       'Astronomical_Twilight'],
      dtype='object')
```

In [17]:
```
len(df.columns)
```

Out[17]:
```
47
```

In [18]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2845342 entries, 0 to 2845341
Data columns (total 47 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   ID                     object 
 1   Severity               int64  
 2   Start_Time             object 
 3   End_Time               object 
 4   Start_Lat              float64
 5   Start_Lng              float64
 6   End_Lat                float64
 7   End_Lng                float64
 8   Distance(mi)           float64
 9   Description            object 
 10  Number                 float64
 11  Street                 object 
 12  Side                   object 
 13  City                   object 
 14  County                 object 
 15  State                  object 
 16  Zipcode                object 
 17  Country                object 
 18  Timezone               object 
 19  Airport_Code           object 
 20  Weather_Timestamp      object 
 21  Temperature(F)         float64
 22  Wind_Chill(F)          float64
 23  Humidity(%)            float64
 24  Pressure(in)           float64
 25  Visibility(mi)         float64
 26  Wind_Direction         object 
 27  Wind_Speed(mph)        float64
 28  Precipitation(in)      float64
 29  Weather_Condition      object 
 30  Amenity                bool   
 31  Bump                   bool   
 32  Crossing               bool   
 33  Give_Way               bool   
 34  Junction               bool   
 35  No_Exit                bool   
 36  Railway                bool   
 37  Roundabout             bool   
 38  Station                bool   
 39  Stop                   bool   
 40  Traffic_Calming        bool   
 41  Traffic_Signal         bool   
 42  Turning_Loop           bool   
 43  Sunrise_Sunset         object 
 44  Civil_Twilight         object 
 45  Nautical_Twilight      object 
 46  Astronomical_Twilight  object 
dtypes: bool(13), float64(13), int64(1), object(20)
memory usage: 773.4+ MB
```

In [19]:
```
# finding only numeric columns
numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
numerics_df = df.select_dtypes(include=numerics)
```

In [20]:
```
numerics_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2845342 entries, 0 to 2845341
Data columns (total 14 columns):
 #   Column             Dtype  
---  ------             -----  
 0   Severity           int64  
 1   Start_Lat          float64
 2   Start_Lng          float64
 3   End_Lat            float64
 4   End_Lng            float64
 5   Distance(mi)       float64
 6   Number             float64
 7   Temperature(F)     float64
 8   Wind_Chill(F)      float64
 9   Humidity(%)        float64
 10  Pressure(in)       float64
 11  Visibility(mi)     float64
 12  Wind_Speed(mph)    float64
 13  Precipitation(in)  float64
dtypes: float64(13), int64(1)
memory usage: 303.9 MB
```

In [21]:
```
# number of columns with numeric value
len(numerics_df.columns)
```

Out[21]:
```
14
```

In [22]:
```
# finding missing values, if values is missing then True else False
df.isna()
```

Out[22]:

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Distan
0	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	
2845337	False	False	False	False	False	False	False	False	
2845338	False	False	False	False	False	False	False	False	
2845339	False	False	False	False	False	False	False	False	
2845340	False	False	False	False	False	False	False	False	
2845341	False	False	False	False	False	False	False	False	

2845342 rows × 47 columns

In [23]:
```
df.isna().sum().sort_values(ascending = False)
```

Out[23]:
```
Number                   1743911
Precipitation(in)         549458
Wind_Chill(F)             469643
Wind_Speed(mph)           157944
Wind_Direction             73775
Humidity(%)                73092
Weather_Condition          70636
Visibility(mi)             70546
Temperature(F)             69274
Pressure(in)               59200
Weather_Timestamp          50736
Airport_Code                9549
Timezone                    3659
Nautical_Twilight           2867
Civil_Twilight              2867
Sunrise_Sunset              2867
Astronomical_Twilight       2867
Zipcode                     1319
City                         137
Street                         2
Country                        0
Junction                       0
Start_Time                     0
End_Time                       0
Start_Lat                      0
Turning_Loop                   0
Traffic_Signal                 0
Traffic_Calming                0
Stop                           0
Station                        0
Roundabout                     0
Railway                        0
No_Exit                        0
Crossing                       0
Give_Way                       0
Bump                           0
Amenity                        0
Start_Lng                      0
End_Lat                        0
End_Lng                        0
Distance(mi)                   0
Description                    0
Severity                       0
Side                           0
County                         0
State                          0
ID                             0
dtype: int64
```

In [24]:
```
# Percentage of missing values
len(df)
```

Out[24]:
```
2845342
```

In [25]:
```
missing_percentage = df.isna().sum().sort_values(ascending = False) / len(df
missing_percentage
```

Out[25]:
```
Number                   6.129003e-01
Precipitation(in)        1.931079e-01
Wind_Chill(F)            1.650568e-01
Wind_Speed(mph)          5.550967e-02
Wind_Direction           2.592834e-02
Humidity(%)              2.568830e-02
Weather_Condition        2.482514e-02
Visibility(mi)           2.479350e-02
Temperature(F)           2.434646e-02
Pressure(in)             2.080593e-02
Weather_Timestamp        1.783125e-02
Airport_Code             3.356011e-03
Timezone                 1.285961e-03
Nautical_Twilight        1.007612e-03
Civil_Twilight           1.007612e-03
Sunrise_Sunset           1.007612e-03
Astronomical_Twilight    1.007612e-03
Zipcode                  4.635647e-04
City                     4.814887e-05
Street                   7.029032e-07
Country                  0.000000e+00
Junction                 0.000000e+00
Start_Time               0.000000e+00
End_Time                 0.000000e+00
Start_Lat                0.000000e+00
Turning_Loop             0.000000e+00
Traffic_Signal           0.000000e+00
Traffic_Calming          0.000000e+00
Stop                     0.000000e+00
Station                  0.000000e+00
Roundabout               0.000000e+00
Railway                  0.000000e+00
No_Exit                  0.000000e+00
Crossing                 0.000000e+00
Give_Way                 0.000000e+00
Bump                     0.000000e+00
Amenity                  0.000000e+00
Start_Lng                0.000000e+00
End_Lat                  0.000000e+00
End_Lng                  0.000000e+00
Distance(mi)             0.000000e+00
Description              0.000000e+00
Severity                 0.000000e+00
Side                     0.000000e+00
County                   0.000000e+00
State                    0.000000e+00
ID                       0.000000e+00
dtype: float64
```

In [26]:
```
type(missing_percentage)
```

Out[26]: pandas.core.series.Series

Exploratory Analysis and Visualization

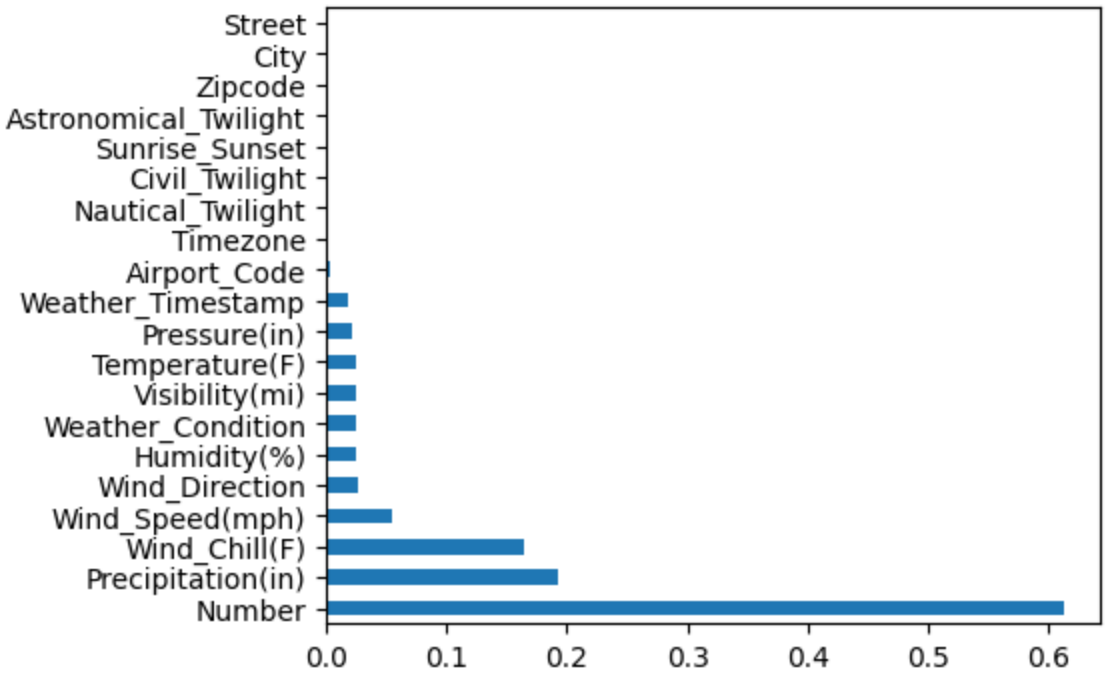
In [27]:
```
# inner part gives boolean results. whenever we have results in boolean and
msg_pt = missing_percentage[missing_percentage != 0]
msg_pt
```

Out[27]:
```
Number                  6.129003e-01
Precipitation(in)       1.931079e-01
Wind_Chill(F)           1.650568e-01
Wind_Speed(mph)         5.550967e-02
Wind_Direction          2.592834e-02
Humidity(%)             2.568830e-02
Weather_Condition       2.482514e-02
Visibility(mi)          2.479350e-02
Temperature(F)          2.434646e-02
Pressure(in)            2.080593e-02
Weather_Timestamp       1.783125e-02
Airport_Code            3.356011e-03
Timezone                1.285961e-03
Nautical_Twilight       1.007612e-03
Civil_Twilight          1.007612e-03
Sunrise_Sunset          1.007612e-03
Astronomical_Twilight   1.007612e-03
Zipcode                 4.635647e-04
City                    4.814887e-05
Street                  7.029032e-07
dtype: float64
```

In [28]:
```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

In [162…
```
msg_pt.plot(kind = 'barh', figsize = (5,4))
```

Out[162]:
```
<AxesSubplot: >
```



__Factors on which we will analysis the dataset__

__1. City__

__2. Start Time__

__3. Start Latitude & Longitude__

__4. Temperature__

__5. Weather Conditions__

__6. Traffic Signal__

__7. Bump__

1. City

In [30]:
```
df.City.count()
```

Out[30]:
```
2845205
```

In [156…
```
df.City.unique()
```

Out[156]:
```
array(['Dublin', 'Dayton', 'Cincinnati', ..., 'Clarksdale', 'Bridgeboro',
       'American Fork-Pleasant Grove'], dtype=object)
```

In [32]:
```
cities = df.City.nunique()
cities
```

Out[32]:
```
11681
```

In [33]:
```
cities_count = df.City.value_counts()
cities_count
```

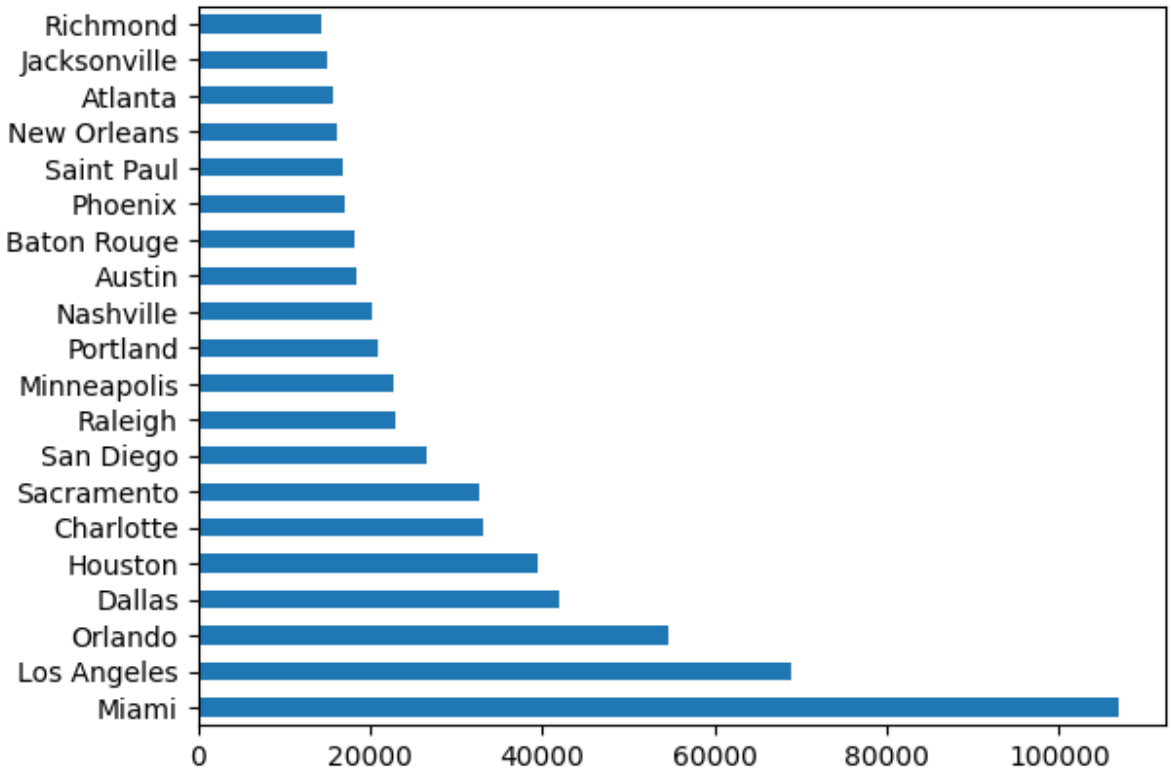
Out[33]:
```
Miami                            106966
Los Angeles                       68956
Orlando                           54691
Dallas                            41979
Houston                           39448
                                    ...
Ridgedale                             1
Sekiu                                 1
Wooldridge                            1
Bullock                               1
American Fork-Pleasant Grove          1
Name: City, Length: 11681, dtype: int64
```

In [34]:
```
cities_count[:20]
```

Out[34]:
```
Miami           106966
Los Angeles      68956
Orlando          54691
Dallas           41979
Houston          39448
Charlotte        33152
Sacramento       32559
San Diego        26627
Raleigh          22840
Minneapolis      22768
Portland         20944
Nashville        20267
Austin           18301
Baton Rouge      18182
Phoenix          17143
Saint Paul       16869
New Orleans      16251
Atlanta          15622
Jacksonville     14967
Richmond         14349
Name: City, dtype: int64
```

In [35]:
```
cities_count[:20].plot(kind = "barh")
```

Out[35]:
```
<AxesSubplot: >
```



The above graph is showing top-20 cities in US by accidents

In [37]:
```
import seaborn as sns
```

In [180…
```
## High and low accidents cities
high_accident_city = cities_count[cities_count >= 1000]
low_accident_city = cities_count[cities_count < 1000]
```

In [181…
```
high_accident_city
```

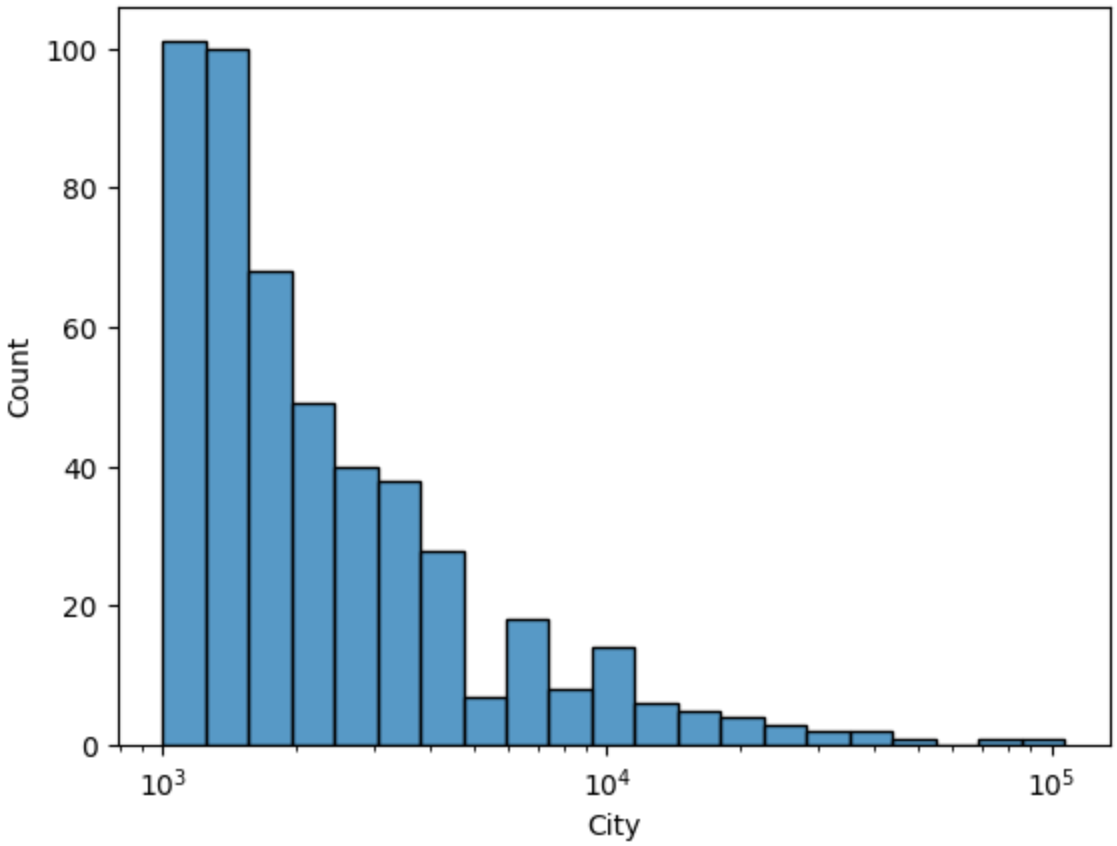
Out[181]:
```
Miami          106966
Los Angeles     68956
Orlando         54691
Dallas          41979
Houston         39448
                ...
Tualatin         1001
Utica            1001
Los Banos        1001
Mankato          1000
Chiloquin        1000
Name: City, Length: 496, dtype: int64
```

In [182…
```
len(high_accident_city) / cities
```

Out[182]:
```
0.04246211796935194
```

In [191…
```
sns.histplot(high_accident_city, log_scale = True)
```

```
: xlabel='City', ylabel='Count'>
```



__Percentage of high accidents cities is 4.2%__

In [184…
```
low_accident_city
```

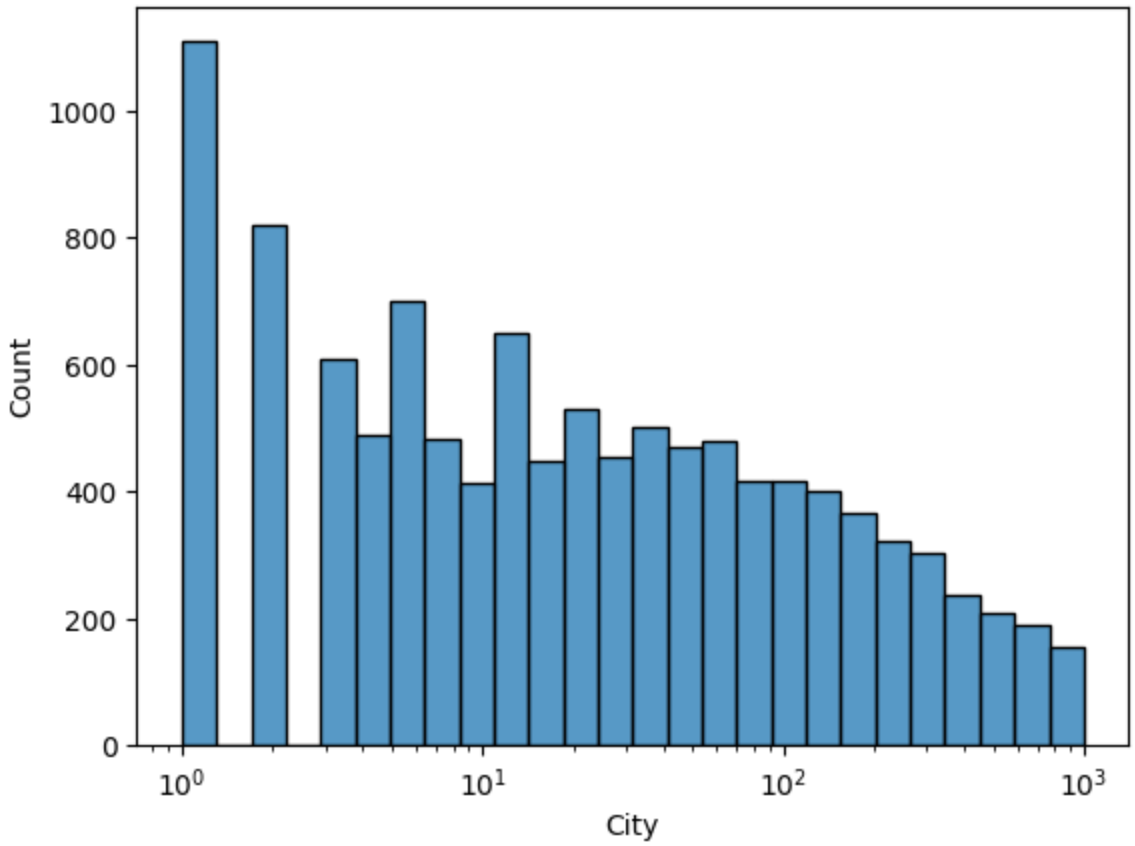
Out[184]:
```
Osseo                              997
Madras                             997
Manor                              992
Portsmouth                         988
Schenectady                        985
                                  ...
Ridgedale                            1
Sekiu                                1
Wooldridge                           1
Bullock                              1
American Fork-Pleasant Grove         1
Name: City, Length: 11185, dtype: int64
```

In [185…
```
len(low_accident_city) / cities
```

Out[185]:
```
0.957537882030648
```

In [192…
```
sns.histplot(low_accident_city, log_scale= True)
```

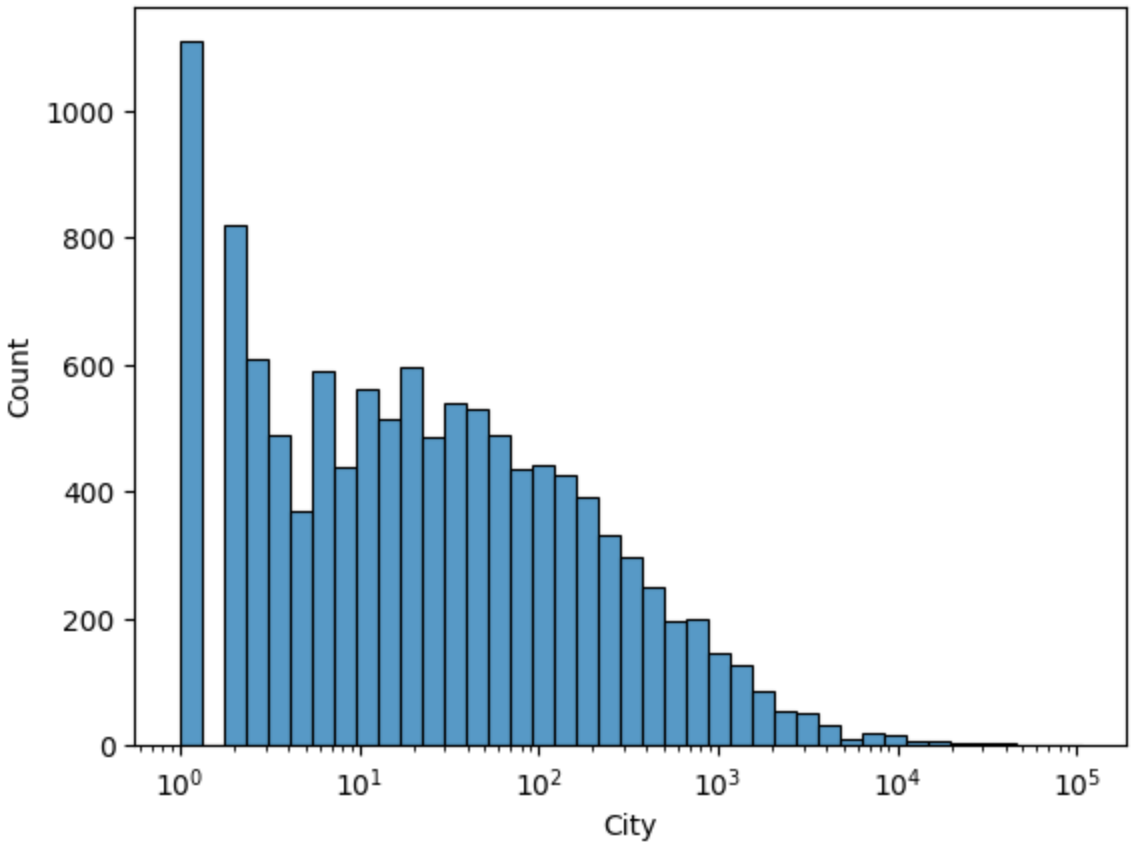
Out[192]:
```
<AxesSubplot: xlabel='City', ylabel='Count'>
```



__Percentage of low accidents cities is 9.5%__

In [186…
```
sns.histplot(cities_count, log_scale = True)
```

Out[186]:
```
<AxesSubplot: xlabel='City', ylabel='Count'>
```



Graph of count of cities by accidents.

In [196…
```
cities_count[cities_count <= 2]
```

Out[196]:
```
Sullivans Island               2
Brilliant                      2
Duson                          2
Binford                        2
Parrott                        2
                              ..
Ridgedale                      1
Sekiu                          1
Wooldridge                     1
Bullock                        1
American Fork-Pleasant Grove   1
Name: City, Length: 1929, dtype: int64
```

Miami, Los Angeles, Orlando, Dallas and Houston are the Top-5 cities where accidents occured more frequently.

Accidents by Cities are exponentially decreasing.

2. Start-Time

In [47]:
```
df.Start_Time
```

Out[47]:
```
0          2016-02-08 00:37:08
1          2016-02-08 05:56:20
2          2016-02-08 06:15:39
3          2016-02-08 06:51:45
4          2016-02-08 07:53:43
                   ...
2845337    2019-08-23 18:03:25
2845338    2019-08-23 19:11:30
2845339    2019-08-23 19:00:21
2845340    2019-08-23 19:00:21
2845341    2019-08-23 18:52:06
Name: Start_Time, Length: 2845342, dtype: object
```

In [48]:
```
# date and time in a string
df.Start_Time[0]
```

Out[48]:
```
'2016-02-08 00:37:08'
```

In [49]:
```
df.Start_Time = pd.to_datetime(df.Start_Time)
```

In [50]:
```
# overwrited Start_Time column and converted to standard date-time column
df.Start_Time[0]
```

Out[50]:
```
Timestamp('2016-02-08 00:37:08')
```

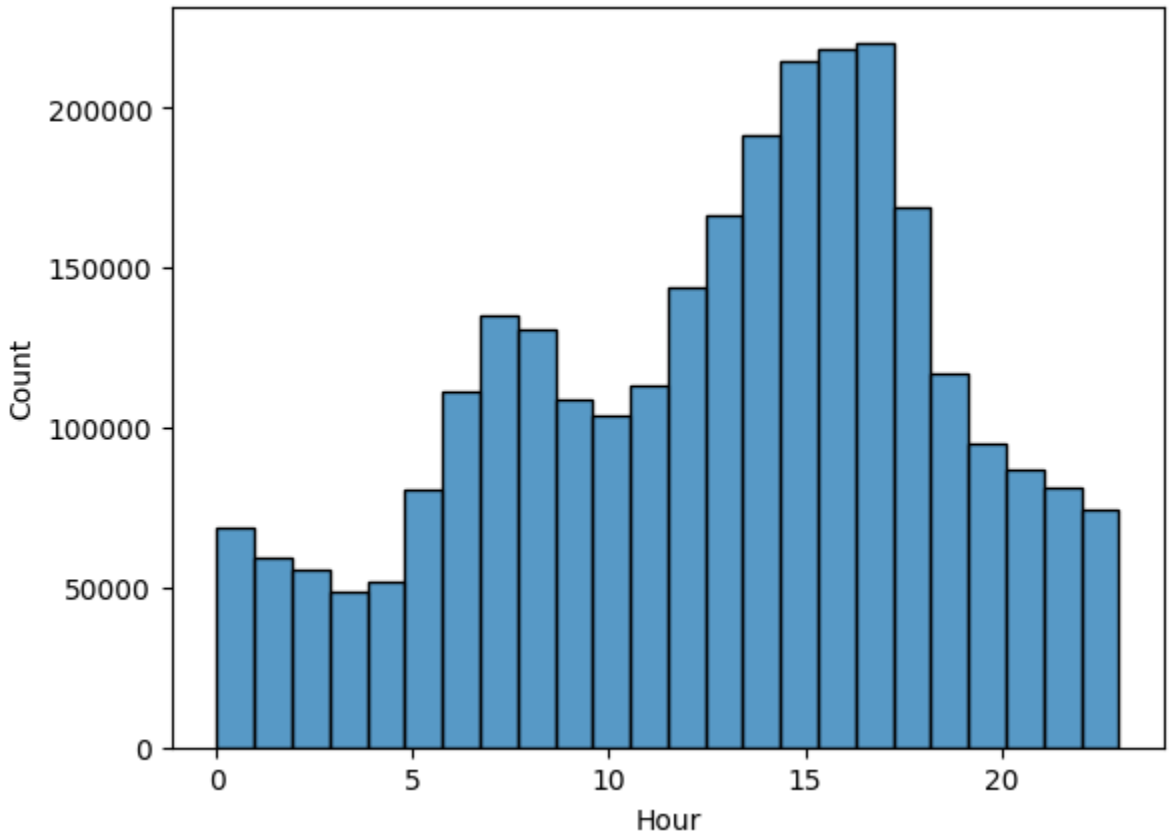
In [51]:
```
df['Hour'] = df['Start_Time'].dt.hour
```

In [52]:
```
df.Hour
```

Out[52]:
```
0           0
1           5
2           6
3           6
4           7
           ..
2845337    18
2845338    19
2845339    19
2845340    19
2845341    18
Name: Hour, Length: 2845342, dtype: int64
```

In [53]:
```
sns.histplot(df.Hour, bins = 24)
```

Out[53]:
```
<AxesSubplot: xlabel='Hour', ylabel='Count'>
```



Most vulnerable hours for accidents are in 6 a.m. to 9 a.m. and 3 p.m. to 6 p.m. i.e. office/working hours.

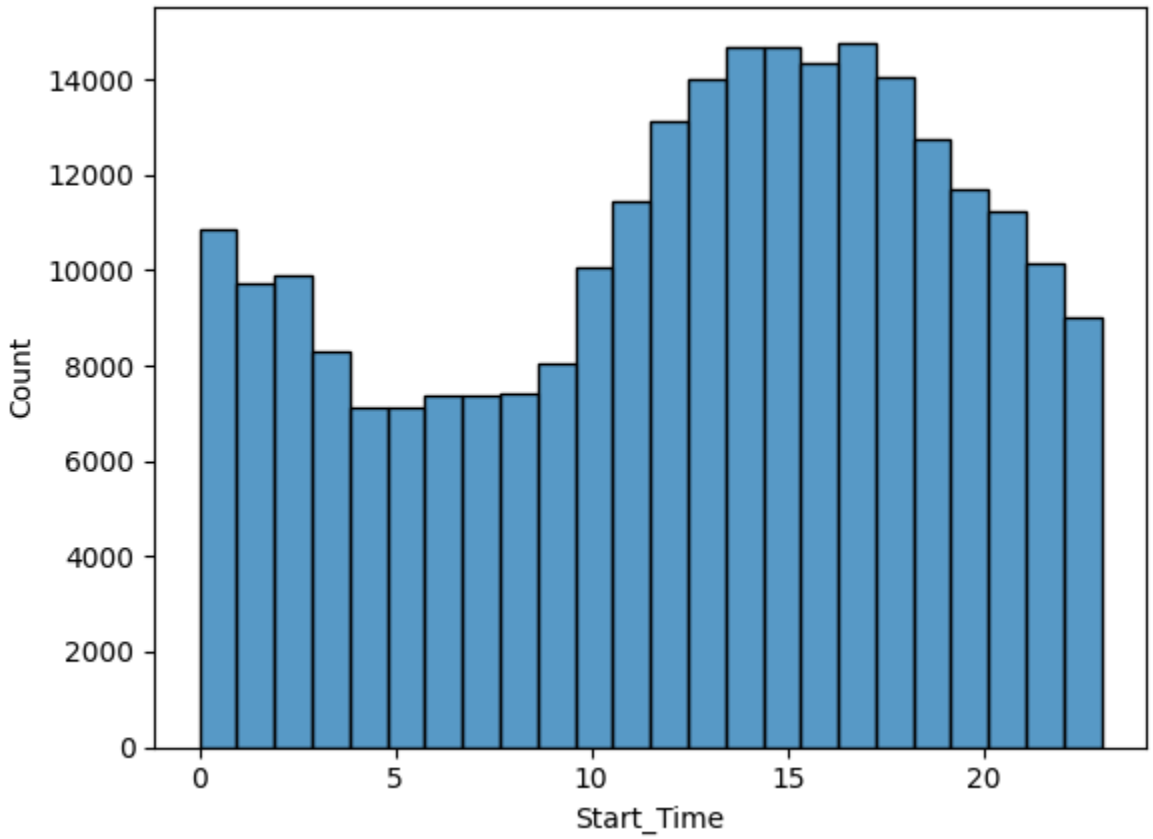
In [54]:
```
sunday = df.Start_Time[df.Start_Time.dt.dayofweek == 6]
```

In [55]:
```
sunday
```

Out[55]:
```
154       2016-02-14 03:58:33
155       2016-02-14 05:26:58
156       2016-02-14 16:30:40
157       2016-02-14 16:38:40
158       2016-02-14 17:40:17
                   ...
2843129   2019-08-18 22:48:14
2843130   2019-08-18 23:24:10
2843243   2019-08-18 22:56:56
2843244   2019-08-18 22:56:56
2843282   2019-08-18 22:54:41
Name: Start_Time, Length: 259274, dtype: datetime64[ns]
```

In [57]:
```
sns.histplot(sunday.dt.hour, bins = 24)
```

Out[57]:
```
<AxesSubplot: xlabel='Start_Time', ylabel='Count'>
```



On Sunday, the peak hours of accidents is 10 a.m. to 4 p.m. It's a time in which people prefers to do leisure activities.

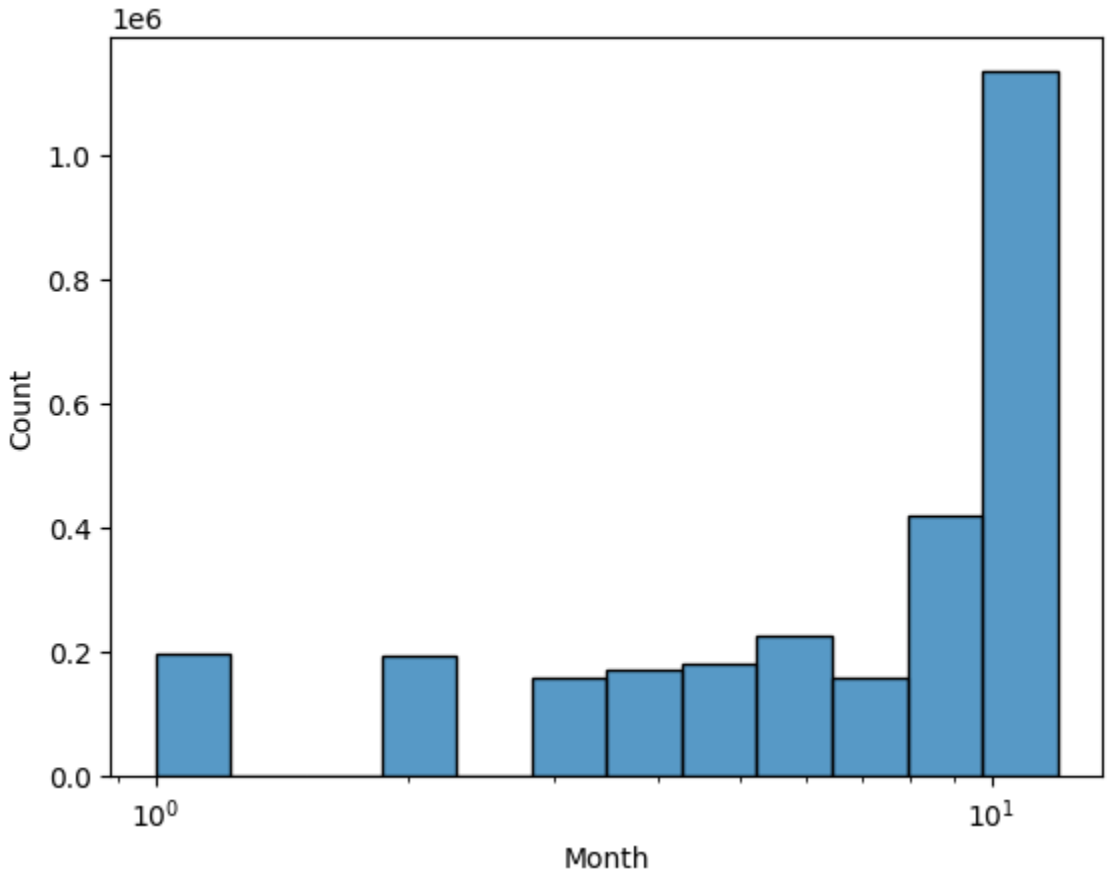
In [58]:
```
df['Month'] = df['Start_Time'].dt.month
```

In [59]:
```
df['Month']
```

Out[59]:
```
0          2
1          2
2          2
3          2
4          2
          ..
2845337    8
2845338    8
2845339    8
2845340    8
2845341    8
Name: Month, Length: 2845342, dtype: int64
```

In [60]:
```
sns.histplot(df.Month, log_scale = True, bins = 12)
```

Out[60]:
```
<AxesSubplot: xlabel='Month', ylabel='Count'>
```



The above graph shows that most of the accidents occured in December (in winters).

why more accidents in winters?

Ans : Due to ice and snow on roadways, which decreases the friction between tyres and roads.

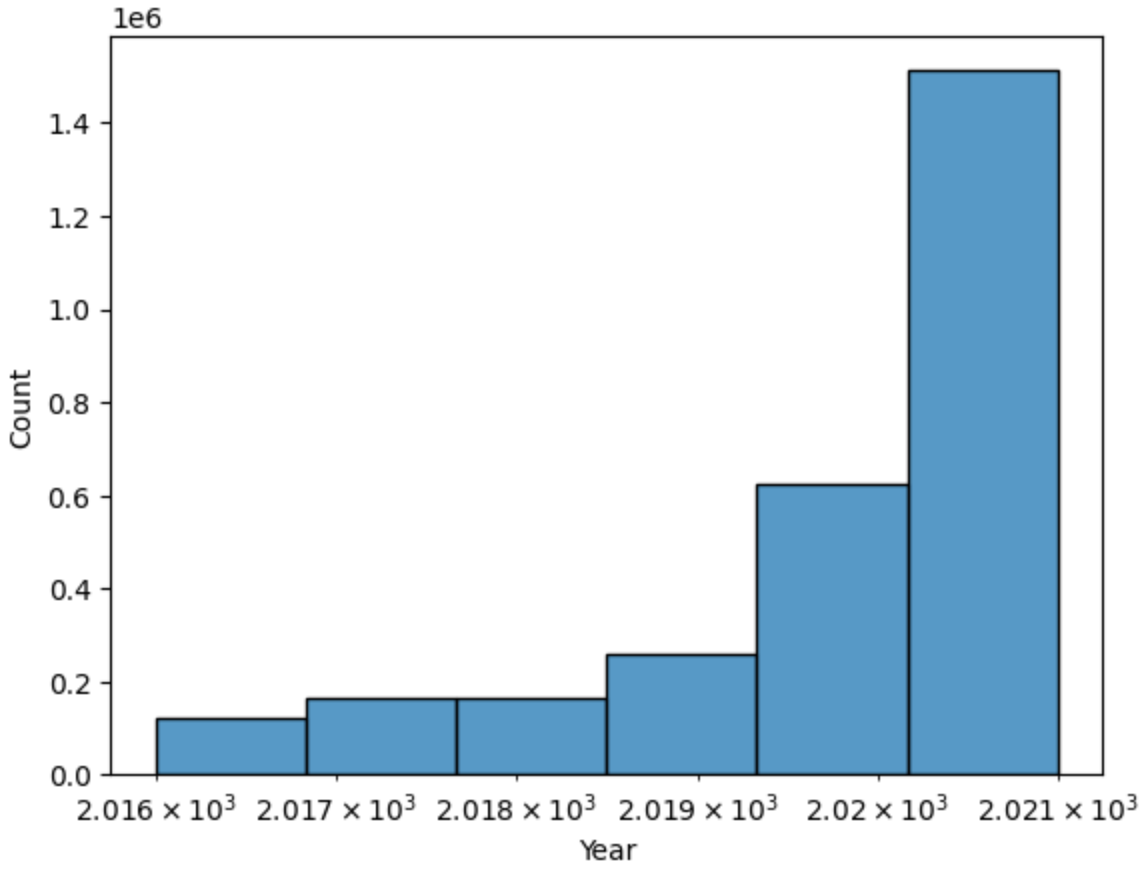
In [62]:
```
df["Year"] = df['Start_Time'].dt.year
```

In [63]:
```
df.Year
```

Out[63]:
```
0          2016
1          2016
2          2016
3          2016
4          2016
           ...
2845337    2019
2845338    2019
2845339    2019
2845340    2019
2845341    2019
Name: Year, Length: 2845342, dtype: int64
```

In [64]:
```
sns.histplot(df.Year, bins = 6, log_scale = True)
```

Out[64]: <AxesSubplot: xlabel='Year', ylabel='Count'>



With every successive year, the number of accidents increased in US.

3. Start Latitude and Longitude

In [66]:
```
df.Start_Lat
```

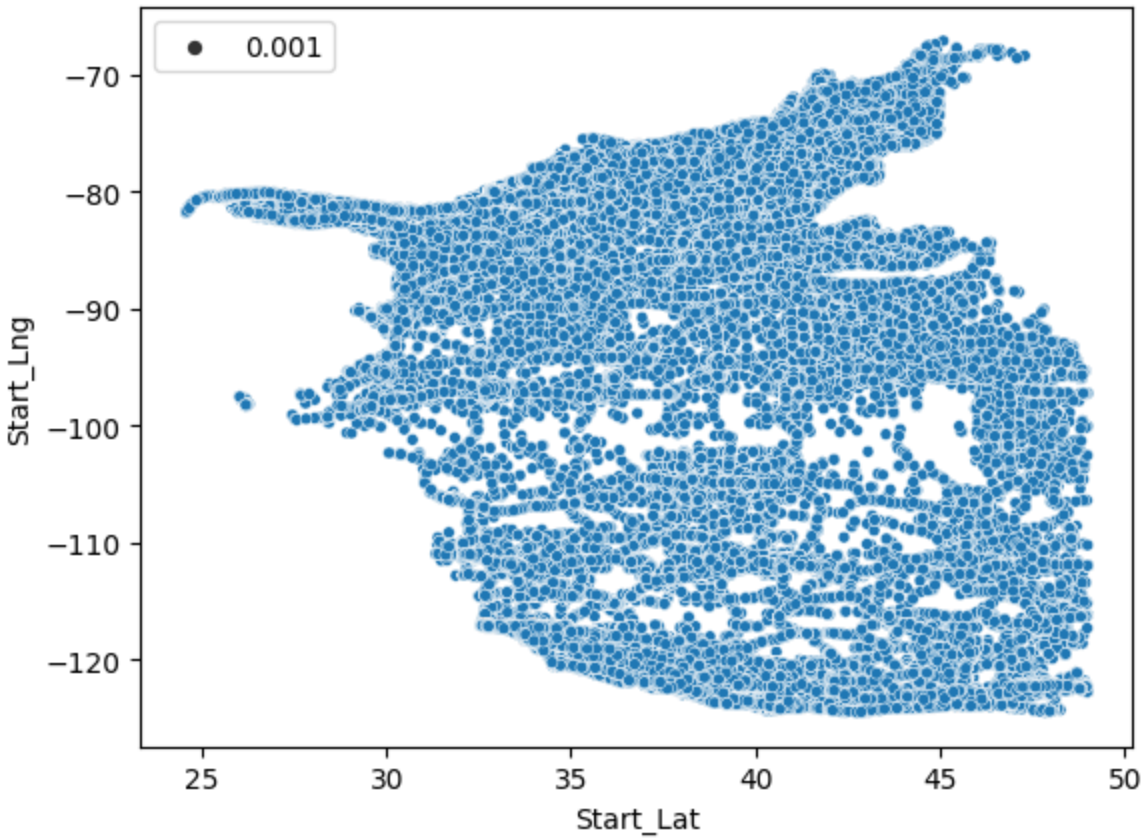
Out[66]:
```
0          40.108910
1          39.865420
2          39.102660
3          41.062130
4          39.172393
              ...
2845337    34.002480
2845338    32.766960
2845339    33.775450
2845340    33.992460
2845341    34.133930
Name: Start_Lat, Length: 2845342, dtype: float64
```

In [67]:
```
df.Start_Lng
```

Out[67]:
```
0          -83.092860
1          -84.062800
2          -84.524680
3          -81.537840
4          -84.492792
               ...
2845337   -117.379360
2845338   -117.148060
2845339   -117.847790
2845340   -118.403020
2845341   -117.230920
Name: Start_Lng, Length: 2845342, dtype: float64
```

In [68]:
```
sns.scatterplot(x = df.Start_Lat, y = df.Start_Lng, size = 0.001)
```

Out[68]:
```
<AxesSubplot: xlabel='Start_Lat', ylabel='Start_Lng'>
```



In [69]:
```
!pip install folium
```

```
Requirement already satisfied: folium in c:\users\shivam\appdata\local\prog
rams\python\python311\lib\site-packages (0.14.0)
Requirement already satisfied: branca>=0.6.0 in c:\users\shivam\appdata\loc
al\programs\python\python311\lib\site-packages (from folium) (0.6.0)
Requirement already satisfied: jinja2>=2.9 in c:\users\shivam\appdata\local
\programs\python\python311\lib\site-packages (from folium) (3.1.2)
Requirement already satisfied: numpy in c:\users\shivam\appdata\local\progr
ams\python\python311\lib\site-packages (from folium) (1.23.5)
Requirement already satisfied: requests in c:\users\shivam\appdata\local\pr
ograms\python\python311\lib\site-packages (from folium) (2.28.2)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\shivam\appdata\l
ocal\programs\python\python311\lib\site-packages (from jinja2>=2.9->folium)
(2.1.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\shivam
\appdata\local\programs\python\python311\lib\site-packages (from requests->
folium) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\shivam\appdata\loca
l\programs\python\python311\lib\site-packages (from requests->folium) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\shivam\app
data\local\programs\python\python311\lib\site-packages (from requests->foli
um) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\shivam\appdat
a\local\programs\python\python311\lib\site-packages (from requests->folium)
(2022.12.7)
[notice] A new release of pip available: 22.3.1 -> 23.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

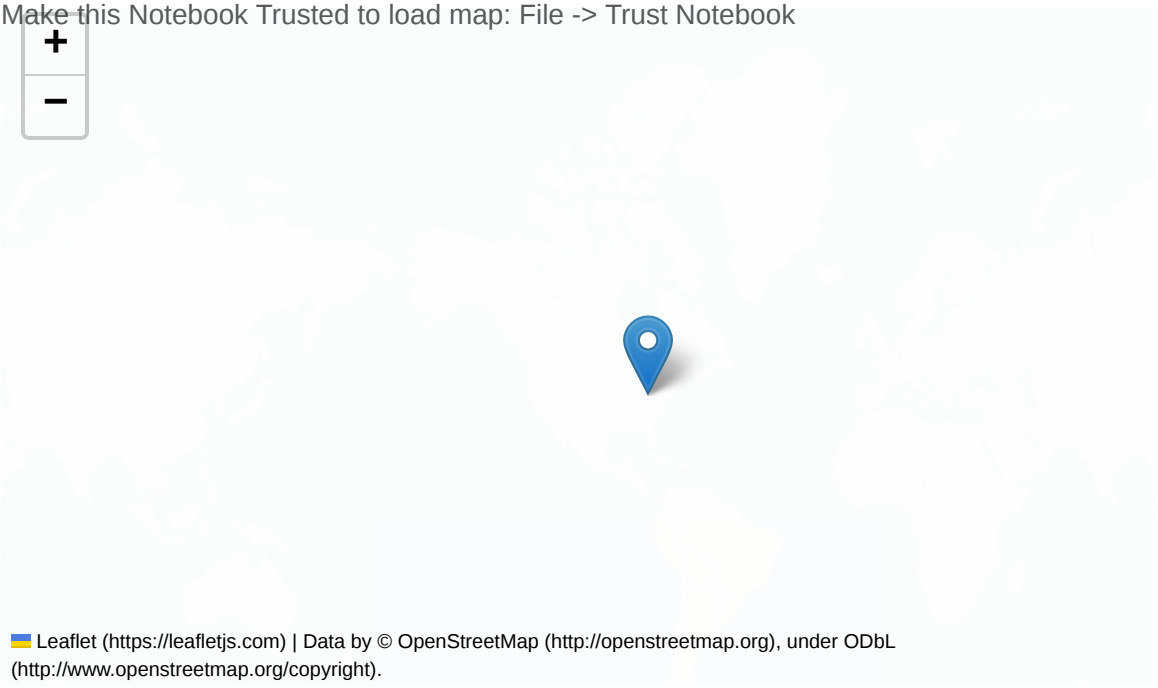
In [70]:
```
import folium
```

In [71]:
```
lat, lng = df.Start_Lat[0], df.Start_Lng[0]
lat, lng
```

Out[71]:
```
(40.10891, -83.09286)
```

In [72]:
```
map = folium.Map()
marker = folium.Marker((lat, lng))
marker.add_to(map)
map
```

Out[72]: Make this Notebook Trusted to load map: File -> Trust Notebook



In [73]:
```
from folium import plugins
from folium.plugins import HeatMap
```

In [74]:
```
list(zip(list(df.Start_Lat), list(df.Start_Lng)))
```

Out[74]:
```
[(40.10891, -83.09286),
 (39.86542, -84.0628),
 (39.10266, -84.52468),
 (41.06213, -81.53784),
 (39.172393, -84.49279200000002),
 (39.06324, -84.03243),
 (39.77565, -84.18603),
 (41.37531, -81.82016999999998),
 (40.702247, -84.075887),
 (40.10931, -82.96849),
 (39.19288, -84.47723),
 (39.13877, -84.53394),
 (41.4739, -81.704233),
 (39.582242, -83.677814),
 (40.151785, -81.312635),
 (40.151747, -81.312682),
 (39.97241, -82.84695),
 (39.9838, -82.85656999999998),
 (40.02664, -82.9944),
 (41.679361, -83.573037),
 (40.99613, -85.26613),
 (39.0381, -84.59243000000002),
 (40.03386, -82.99601),
 (39.85798, -84.28181),
 (39.15267, -84.5395),
 (39.30732, -85.95982),
 (39.77572, -84.04815),
 (39.97643, -83.12306),
 (39.58595, -85.82518),
 (39.3638, -85.51659000000002),
 (39.45132, -85.62516),
 (39.75067, -84.14148),
 (39.2675, -81.49929),
 (39.86025, -85.96623000000002),
 (41.95677, -83.67214),
 (38.27401, -85.74844),
 (41.95677, -83.67214),
 (40.52225, -80.06666),
 (40.487814, -80.009439),
 (40.156298, -83.018431),
 (41.47461, -81.71181999999997),
 (41.0618, -81.54608),
 (40.45112, -85.15048),
 (40.35429, -85.14993),
 (39.75855, -85.13715),
 (39.74408, -85.13749),
 (39.965148, -83.020499),
 (40.72813, -84.78965),
 (39.85223, -85.25846999999997),
 (41.46747, -81.75909),
 (41.83193, -80.10143000000002),
 (39.744287, -84.204939),
 (39.71548, -84.22033),
 (39.7504, -84.20561),
 (41.72316, -84.96444),
 (41.0, -82.99708000000003),
```

```
(38.1781, -85.71946),
(38.18577, -85.80678),
(38.27191, -85.80838),
(39.923905, -82.87008),
(41.035566, -81.569917),
(39.9239, -83.68767),
(41.851914, -80.175232),
(39.93849, -82.84849),
(38.96943, -80.1096),
(41.66805, -83.57063000000002),
(41.48339, -81.66297),
(41.50127, -81.4804),
(39.789093, -82.989106),
(41.67073, -81.24561),
(41.70846, -81.17636),
(41.44246, -81.78485),
(39.77128, -84.1923),
(41.50499, -81.47417),
(40.109653, -80.2029),
(38.38852, -81.7687),
(41.03572, -81.57809),
(41.47487, -81.72095),
(41.62845, -84.80559000000002),
(41.62894, -84.80373),
(41.16102, -81.78573),
(41.628232, -84.808858),
(41.62986, -84.76619000000002),
(39.74729, -84.21426),
(41.42099, -81.69051999999998),
(41.42318, -81.84674),
(38.79691, -84.48273),
(38.80878, -84.49638),
(38.30155, -85.85499),
(38.33667, -81.65623000000002),
(38.33614, -81.65623000000002),
(41.0961, -81.58593),
(41.12155, -85.18715999999998),
(38.829993, -80.667067),
(40.05642, -83.03097),
(41.37717, -81.5139),
(39.17397, -84.49031),
(39.75513, -84.16614),
(39.17397, -84.49031),
(41.12624, -81.65299),
(38.33667, -81.65623000000002),
(41.10389, -81.5),
(39.16417, -84.45275),
(40.11184, -83.00883),
(38.31922, -85.75301999999998),
(39.09881, -84.50801),
(41.473859, -81.708409),
(41.039988, -81.565332),
(41.73565, -84.99082),
(39.26566, -84.43965),
(41.46981, -81.81104),
                  81.63157),
```

```
(40.58919, -80.09885),
(41.06356, -81.49078),
(41.06218, -81.50595),
(39.17736, -84.4873),
(38.33956, -81.60991999999997),
(41.47889, -81.6934),
(40.58919, -80.09885),
(41.41119000000001, -81.64414000000002),
(39.24907, -84.44561999999998),
(39.28775, -84.40073000000002),
(41.47379, -81.89631),
(41.54941, -81.60676),
(39.03347, -84.603),
(39.74903, -84.20106),
(39.89473, -83.044),
(39.19641, -84.47331),
(39.22461, -84.45364000000002),
(39.15306, -84.44283),
(39.97262, -82.98368),
(40.535447, -80.03596),
(40.529455, -80.03506999999998),
(41.49647, -81.67733),
(41.0634, -81.49079),
(39.74284, -84.20479),
(41.46822, -81.9476),
(41.92479, -83.36026),
(41.184738, -80.432713),
(40.27317, -85.5577),
(39.98264, -85.14301999999998),
(40.27317, -85.5577),
(40.42667, -85.54968000000002),
(39.97527, -85.14018),
(39.09889, -84.52118),
(38.88012, -84.62527),
(38.79256, -84.60423),
(39.74284, -84.20479),
(40.484222, -80.13755400000002),
(41.20104, -80.06859),
(39.05528, -84.39072),
(41.623325, -83.692226),
(40.53024, -80.00957),
(40.06527, -82.99591),
(40.43954, -80.00971),
(41.422004, -81.730092),
(39.3638, -85.51659000000002),
(39.82083, -85.7749),
(40.0366, -85.72561999999998),
(39.855691, -84.29079200000002),
(40.049152, -85.682047),
(39.83939, -85.55752),
(40.13386, -80.00666),
(40.43954, -80.00971),
(40.13386, -80.00666),
(40.13386, -80.00666),
(39.97859, -82.9763),
                  83.04255),
```

```
(39.68291, -85.97372),
(38.49304, -85.77156),
(38.33428, -85.82177),
(39.99921, -83.02875),
(38.53844, -85.77702),
(38.1781, -85.71946),
(39.87433, -82.99882),
(41.39546, -85.32855),
(41.39469, -85.28893000000002),
(41.90473, -83.36986),
(40.77137000000001, -84.62203000000002),
(40.80059, -84.61374),
(39.87433, -82.99882),
(40.78593, -84.61787),
(40.80059, -84.61374),
(41.67487, -83.57266),
(41.47376, -81.86113),
(38.33465, -85.75389),
(39.14014, -84.53495),
(41.80143, -85.34888000000002),
(41.35761, -81.81988),
(39.34392, -84.39043000000002),
(39.816154, -83.18010500000003),
(38.824929, -85.47449499999998),
(38.82415, -85.63794),
(40.593244, -85.150017),
(40.600487, -85.150502),
(41.19362, -80.2374),
(40.55267, -85.14974000000002),
(40.56744000000001, -85.14993),
(40.14212, -81.55153),
(39.94397, -82.41127),
(39.96893, -83.11922),
(40.6126, -80.09494000000002),
(40.11184, -83.00883),
(41.06347, -81.50372),
(39.972931, -85.965783),
(39.95459, -83.04255),
(40.11184, -83.00883),
(39.97241, -82.84695),
(39.9912, -85.9188),
(41.34965, -81.51146),
(41.11987, -80.69004),
(41.11851, -80.69242),
(40.05642, -83.03097),
(38.117487, -85.760397),
(41.44134, -81.65055),
(38.61531, -80.76034),
(41.124052, -80.677589),
(39.95085, -82.94428),
(41.06257, -81.52228000000002),
(40.11212, -83.03905999999998),
(39.887033, -83.043909),
(40.04916, -83.03336),
(41.12906, -85.1612),
                    85.10323000000002),
```

```
(38.08783, -83.90263),
(41.63316, -83.54203000000003),
(39.149639, -84.539381),
(40.9605, -85.3528),
(40.96051, -85.34522),
(41.08204, -80.64846999999997),
(41.61791, -83.54106999999998),
(38.41592, -82.3465),
(41.67571, -83.69385),
(39.08272, -84.52264),
(39.280947, -80.278075),
(38.36387, -81.73867),
(40.0187, -82.90465999999998),
(41.53757, -81.64083000000002),
(41.16522, -81.47454),
(39.52352, -82.02360999999998),
(39.517216, -82.025802),
(40.322818, -83.071034),
(41.73565, -84.99082),
(39.14648, -84.45455),
(39.82122, -85.76482),
(41.11889, -81.65406999999998),
(40.121557, -83.151457),
(40.11017, -83.15876999999998),
(41.57777, -81.54987),
(40.121557, -83.151457),
(40.11017, -83.15876999999998),
(41.04592, -81.69463),
(41.42099, -81.69051999999998),
(39.27486, -84.346),
(39.28266, -84.56742),
(39.9467, -82.22232),
(39.17397, -84.49031),
(39.17736, -84.4873),
(39.9467, -82.22232),
(39.9467, -82.22232),
(41.68518, -83.5678),
(41.66572, -83.5652),
(39.27125, -84.35176),
(41.47395, -81.69931),
(39.98505, -80.74006),
(39.95813, -80.76219),
(41.56544, -81.5835),
(38.32587, -83.11389),
(38.25869, -85.7622),
(38.29322, -85.654625),
(39.78609, -83.24606999999997),
(41.53968, -81.62795),
(39.08927, -84.5227),
(38.25922, -85.76632),
(41.55264, -81.60032),
(39.02187, -84.50492),
(41.06755, -81.57421),
(40.49223, -80.01029),
(41.48836, -81.66471),
                  82.94428),
```

```
(41.42063, -81.80859),
(41.06755, -81.57421),
(40.82949, -81.39732),
(41.40091, -81.81724),
(41.13355, -85.13584),
(38.870386, -84.624429),
(41.47974, -81.6667),
(40.30095, -80.16651999999998),
(39.11543, -84.50003000000002),
(41.1419, -81.47694),
(39.14917, -84.44711),
(39.81979000000001, -84.18908),
(39.84509, -84.1899),
(41.47461, -81.71181999999997),
(39.17304, -84.47589),
(39.1021, -84.49754),
(41.49873, -81.67260999999998),
(39.12627, -84.53509),
(41.47394600000001, -81.699011),
(39.9745, -83.09654),
(39.85911, -84.2808),
(39.24943, -84.36541),
(38.31179, -85.58773000000002),
(39.27486, -84.346),
(39.29287, -84.31981),
(39.15267, -84.5395),
(41.4675, -81.49014),
(39.26661, -84.35605),
(39.27125, -84.35176),
(38.20674, -85.74846),
(40.13804, -82.97112),
(41.47965, -81.66758),
(39.08887, -84.52271),
(41.6882, -83.64064),
(40.06527, -82.99591),
(39.86536, -84.05238),
(38.420223, -82.293065),
(41.47414000000001, -81.69893),
(41.47542, -81.68721),
(40.07297, -83.1347),
(40.05609000000001, -80.679211),
(38.19105, -85.71902),
(38.25454000000001, -85.71969),
(39.171694, -84.494399),
(41.06239, -81.52229),
(39.259254, -84.440978),
(39.905178, -82.895471),
(40.0988, -83.13331),
(41.33991, -81.81739),
(40.0999, -82.92089),
(39.4728, -80.11963),
(41.46631, -81.77065999999998),
(38.34109, -81.70017),
(41.160072, -85.232647),
(41.163626, -85.24482900000002),
                84.53495),
```

```
(40.94537, -81.15473),
(41.68591, -83.911307),
(41.704523, -83.9136),
(40.945772, -81.17219399999998),
(38.82445, -82.98299),
(38.84111, -82.98432),
(39.00079, -84.42043000000002),
(39.96071, -85.35383),
(39.2675, -81.49929),
(40.27317, -85.5577),
(40.42667, -85.54968000000002),
(38.130782, -81.388878),
(38.545195, -85.778613),
(38.362427, -81.714434),
(38.39115, -81.58664),
(40.08978, -82.98725999999998),
(40.0122, -82.99188000000002),
(41.68918, -83.53681999999998),
(39.3781, -84.365),
(39.11919, -84.53539),
(39.05029, -83.77673),
(39.04827, -83.77199),
(40.0717, -82.90821),
(39.09655, -84.48312),
(40.43077, -80.02638),
(41.443708, -85.19285),
(41.438119, -85.15969399999999),
(41.46747, -81.75909),
(39.044435, -84.465864),
(39.22461, -84.45364000000002),
(41.104217, -81.499987),
(41.1003, -81.4999),
(39.1021, -84.49754),
(41.46375, -81.69353000000002),
(41.52134, -83.45642),
(41.52549000000001, -83.46344),
(41.43166, -81.68160999999998),
(38.22363, -85.50532),
(40.85466, -81.41646999999998),
(38.22272, -85.50111),
(39.12424, -84.53501),
(39.09858, -84.51974),
(39.63551, -84.19519),
(40.39715, -80.10368000000003),
(41.56709, -81.57816),
(41.18225, -84.94275999999998),
(39.10148, -84.52341),
(41.55245, -83.60061999999998),
(38.52006, -81.35221),
(41.3513, -83.62276),
(39.25015, -81.30566),
(41.345031, -82.285379),
(41.351335, -82.26144599999998),
(38.38457, -81.804429),
(40.41528, -80.01234000000002),
                83.12479),
```

```
(41.03669, -81.50492),
(41.68971, -83.53045999999998),
(39.587784, -83.723939),
(39.626237, -83.73105),
(41.06243, -81.49904000000002),
(39.97262, -82.98368),
(39.93658, -83.00941999999998),
(41.208579, -82.807829),
(41.4051, -81.8184),
(39.29035, -84.39078),
(39.90524, -82.89541),
(40.04376, -82.99708000000003),
(39.98793, -82.98568),
(41.874915, -85.409527),
(41.890234, -85.409642),
(39.9335, -82.78939),
(40.04916, -83.03336),
(39.99493, -82.93332),
(39.83727, -83.09326999999998),
(40.98954000000001, -81.49372),
(38.53844, -85.77702),
(41.10389, -81.5),
(38.577894, -85.77896),
(41.42099, -81.69051999999998),
(38.58054600000001, -85.77866),
(39.833935, -84.061832),
(38.30344, -85.88893),
(39.93654, -83.40048),
(39.95612, -83.37499),
(39.17766, -84.39780999999998),
(40.02155, -83.0354),
(39.97518, -85.63374),
(39.94576, -85.63454),
(40.27744000000001, -85.72518000000002),
(39.17304, -84.47589),
(41.68648, -83.5669),
(41.66267, -85.03312),
(41.68976, -83.666),
(39.94594, -82.9432),
(41.4758, -81.66036),
(41.628312, -84.812135),
(39.96724, -83.02459),
(41.96429000000001, -83.35018000000002),
(41.282993, -83.638931),
(41.63126, -83.48336),
(41.427584, -85.8495),
(41.471378, -85.839527),
(41.46747, -81.75909),
(39.18364, -84.48373000000002),
(41.94495, -80.40234),
(39.53078, -84.310264),
(41.438774, -81.80341999999997),
(39.9838, -82.85656999999998),
(40.44413400000001, -80.028261),
(39.97727, -83.15445),
(39.9745, -83.09654),
```

```
(39.97865, -83.11698),
(39.97327, -83.12002),
(39.22571, -84.36832),
(39.986342, -83.118881),
(39.73629, -84.20489),
(41.4051, -81.8184),
(41.41101, -81.66634),
(39.24464, -84.44835),
(40.11212, -83.03905999999998),
(40.30095, -80.16651999999998),
(41.49243, -81.67554),
(40.1827, -80.26148),
(39.95459, -83.04255),
(39.97527, -85.14018),
(39.98511, -85.14406),
(38.29514, -85.75655),
(39.93654, -83.40048),
(39.11543, -84.50003000000002),
(41.030684, -81.885739),
(41.43166, -81.68160999999998),
(40.6664, -80.227514),
(40.67137, -80.22581),
(41.305199, -81.438626),
(40.354252, -85.843032),
(40.402251, -85.842969),
(41.26752000000001, -85.85673),
(39.97307, -82.98405),
(39.9672, -81.28699999999998),
(39.96392, -81.27197),
(41.52631, -83.62123000000003),
(39.93292, -82.83025),
(39.08272, -84.52264),
(39.8552, -84.32502),
(41.65404, -85.70526),
(41.39465, -81.65326),
(41.06211, -81.53053),
(41.09598, -81.49976),
(41.85222, -85.67843),
(38.148808, -80.929665),
(41.52447, -85.57791),
(41.5105, -85.57781),
(39.90524, -82.89541),
(41.23598, -81.49325999999998),
(38.70101, -80.66483000000002),
(39.22498, -84.38094),
(40.776023, -80.12718100000002),
(38.86027, -80.65729),
(39.88203, -83.04748000000002),
(39.89253400000001, -83.038786),
(39.89698, -83.03428000000002),
(41.67439, -83.693727),
(41.12999, -85.15503000000002),
(39.09889, -84.52118),
(41.52631, -83.62123000000003),
(39.088552, -84.522724),
                  82.98879000000002),
```

```
(39.21018, -84.46345),
(39.95369, -82.96403000000002),
(40.40934, -80.19247),
(40.44126, -80.19314),
(39.16161, -84.51375),
(39.16161, -84.51375),
(39.15141, -84.54061),
(39.75855, -85.13715),
(41.67571, -83.69385),
(41.67571, -83.69385),
(39.28805, -84.37462),
(41.53618, -83.6176),
(39.07202, -84.51907),
(38.259133, -85.752672),
(40.44425, -80.02821),
(40.44278, -80.02468),
(38.30155, -85.85499),
(40.35405, -80.13756),
(40.35449000000001, -80.13835999999998),
(38.30344, -85.88893),
(41.08789, -83.66011),
(41.02454, -80.92163000000002),
(41.02413, -80.90579),
(39.08168, -84.7069),
(38.18539000000001, -85.81793),
(39.27734, -84.86438000000003),
(39.16437, -84.43731),
(39.748585, -84.226084),
(39.4728, -80.11963),
(41.187807, -80.149019),
(41.19242, -80.15860500000002),
(38.233483, -85.62671800000003),
(38.33954, -81.609895),
(40.07297, -83.1347),
(40.08402, -83.13046),
(38.34631, -83.40449),
(38.35382, -83.40553),
(39.04416, -84.46656),
(38.41807, -81.54525),
(38.309401, -85.847643),
(41.64679, -83.54913),
(38.198748, -85.733953),
(39.97401, -82.99355),
(39.17397, -84.49031),
(41.68843, -83.56258000000003),
(41.67904, -83.5732),
(39.25249, -84.36493),
(38.310736, -85.590395),
(41.46941, -81.80535),
(38.20674, -85.74846),
(39.226317, -84.37107900000002),
(41.68721, -83.55579),
(40.09775, -84.22926),
(41.42147, -81.70682),
(38.29414600000001, -85.756847),
                  84.47589),
```

```
(38.15982, -84.67308),
(39.11675, -84.53355),
(39.24943, -84.36541),
(41.06146, -81.5049),
(39.18364, -84.48373000000002),
(39.98793, -82.98568),
(39.95971, -83.04811),
(39.1479, -84.53929000000002),
(39.89555, -83.84997),
(39.99226, -82.98517),
(39.40983, -82.97548),
(40.6126, -80.09494000000002),
(39.22479000000001, -84.27783000000002),
(39.11675, -84.53355),
(41.43869, -81.80554000000002),
(40.016819, -82.90411),
(39.09757000000001, -84.51666),
(41.06247000000001, -81.52405),
(39.13219, -84.49464),
(39.097654, -84.516019),
(41.34965, -81.51146),
(41.37717, -81.5139),
(40.12589000000001, -80.13661),
(40.13189000000001, -80.132407),
(40.00507, -83.1186),
(39.43794000000001, -82.53893000000002),
(39.12627, -84.53509),
(38.07759, -84.45711999999997),
(40.09079000000001, -82.90783),
(39.64731, -85.91696999999998),
(39.08927, -84.5227),
(38.26109, -85.73695),
(38.19785, -85.70445),
(39.05654000000001, -84.54283000000002),
(41.42413, -81.64519),
(39.28638, -84.42706),
(41.37499, -83.61635),
(41.06228, -81.50986999999998),
(41.53907, -81.63354),
(40.04916, -83.03336),
(41.13613, -81.72196),
(39.09891, -84.49181999999998),
(39.96268, -82.84305),
(39.17397, -84.49031),
(39.16911, -84.42104),
(39.003821, -80.391285),
(39.17736, -84.4873),
(39.14316, -84.47069),
(39.2696, -84.34894),
(40.521732, -81.64755),
(40.513361, -81.651599),
(41.52631, -83.62123000000003),
(38.992607, -80.195782),
(38.839151, -82.152877),
(40.27675, -85.49916),
                80.13163),
```

```
(39.83711, -84.54392),
(39.02296, -84.51784),
(38.455787, -81.88205),
(39.02397, -84.48039),
(38.22433, -85.751),
(39.05973, -84.42438),
(39.27734, -84.86438000000003),
(41.47965, -81.66758),
(39.02257, -84.48631),
(40.44846, -80.00755),
(39.09891, -84.28556999999998),
(41.06026, -81.55485999999998),
(41.802758, -83.684687),
(38.0791, -83.94591),
(38.01846, -84.15159),
(38.30344, -85.88893),
(38.84109, -85.81618),
(41.14181, -80.72015999999998),
(39.14648, -84.45455),
(41.04263, -81.50543),
(38.21089, -85.82606),
(39.75067, -84.14148),
(38.28581, -85.50792),
(39.15273300000001, -84.541403),
(38.23585, -85.658849),
(38.20249, -85.68697),
(40.53491, -83.10678),
(40.52035, -83.10043),
(39.12965, -84.49555),
(38.22811, -85.7514),
(40.41233, -80.07815),
(39.7504, -84.20561),
(39.08272, -84.52264),
(39.781457, -85.839651),
(39.781457, -85.839651),
(41.06076, -81.45923),
(39.688148, -83.88431),
(39.693164, -83.882745),
(39.9488, -83.02845),
(41.11584000000001, -81.61428000000002),
(39.797697, -84.189677),
(39.20326, -85.95763000000002),
(39.24464, -84.44835),
(39.73733, -84.25733000000002),
(39.969853, -85.972976),
(41.11889, -81.65406999999998),
(39.13219, -84.49464),
(41.458, -83.62201999999998),
(40.09268, -83.03726),
(40.066965, -83.289888),
(40.045608, -83.29660799999998),
(39.97307, -82.98405),
(39.98248, -85.123922),
(39.981095, -85.11027299999998),
(40.43021, -80.02554),
(40.4231000000001, -80.02781999999998),
```

```
(39.976567, -83.128639),
(41.256658, -85.363439),
(41.37499, -83.61635),
(41.252653, -85.347328),
(39.93658, -83.00941999999998),
(39.14014, -84.53495),
(40.153014, -82.970316),
(39.90524, -82.89541),
(39.13219, -84.49464),
(40.42581, -80.4294),
(41.08789, -83.65977),
(39.9488, -83.02845),
(41.47585, -81.87980999999998),
(39.1079, -84.50284),
(38.340803, -81.680728),
(40.96473, -85.28862),
(38.441, -82.12679),
(38.4413, -82.01529000000002),
(38.441, -82.12679),
(38.18977, -83.48379),
(41.194131, -80.201982),
(40.09601, -83.13626),
(41.23632, -83.65401999999997),
(38.1737, -84.81473000000003),
(38.19016, -85.7616),
(39.25249, -84.36493),
(39.137217, -84.04718299999998),
(39.141454, -84.016543),
(38.344424, -81.613662),
(41.93113, -83.35857),
(40.89268, -84.61808),
(40.46013, -80.19126),
(41.119342, -80.698025),
(41.117872, -80.693898),
(39.15267, -84.5395),
(38.27362, -85.80906),
(41.458, -83.62201999999998),
(39.35655, -84.26419),
(38.4062, -81.54054000000002),
(41.02525900000001, -81.5042),
(38.27362, -85.80906),
(38.11969000000001, -85.77235),
(38.275406, -85.81314300000003),
(38.27508, -85.81517099999998),
(41.39769, -81.939),
(39.46513, -80.13559000000002),
(38.217994, -81.426024),
(38.215401, -81.42437199999998),
(39.042851, -84.612615),
(38.26109, -85.73695),
(39.05654000000001, -84.54283000000002),
(41.840758, -84.361657),
(41.841123, -84.361652),
(41.63316, -83.54203000000003),
(40.02874, -81.04368000000002),
(      , -81.80258),
```

```
(41.0811, -81.50348000000002),
(38.35575, -81.65420999999998),
(39.94585, -82.94293),
(41.69017, -83.5181),
(40.11212, -83.03905999999998),
(41.50259000000001, -81.66974),
(39.989994, -83.02512),
(41.480458, -81.692932),
(39.22461, -84.45364000000002),
(40.04376, -82.99708000000003),
(39.93621, -82.88318000000002),
(39.167788, -82.524172),
(39.20655, -82.48212),
(41.36576, -81.51602),
(39.977322, -81.183327),
(39.961933, -81.175436),
(39.68292, -84.23004),
(40.02664, -82.9944),
(38.583372, -81.994824),
(38.58119600000001, -82.004356),
(40.07297, -83.1347),
(41.58689, -83.56234),
(38.31714, -85.57677),
(40.07297, -83.1347),
(39.07202, -84.51907),
(41.46747, -81.75909),
(40.82645, -81.39675),
(40.82645, -81.39675),
(39.05654000000001, -84.54283000000002),
(39.06272, -84.65728),
(39.03368, -84.59527),
(40.02121, -82.99503),
(39.866695, -84.9305),
(39.05654000000001, -84.54283000000002),
(39.03232, -84.59137),
(41.04058, -81.61256999999998),
(40.98884, -83.64875),
(39.95658, -83.37075),
(39.89857, -83.72139),
(39.896448, -83.72312099999998),
(41.47965, -81.66758),
(39.71548, -84.22033),
(40.0389, -85.71828000000002),
(41.17516, -81.23116999999998),
(41.15107, -81.21375),
(40.0389, -85.71828000000002),
(39.99336, -85.83861999999998),
(40.46013, -80.19126),
(41.68976, -83.666),
(39.97326, -83.00690999999998),
(39.48572, -83.94191),
(39.83355, -85.99635),
(39.79721, -85.97153),
(39.98056, -83.11924),
(41.458, -83.62243000000002),
                -82.995352),
```

```
(39.99921, -83.02875),
(40.869049, -81.95845600000001),
(41.419021, -81.519515),
(39.34392, -84.39043000000002),
(39.9454, -82.60271),
(38.353957, -81.73513100000002),
(38.354505, -81.731585),
(41.501623, -81.479519),
(39.86477, -83.99871),
(38.737813, -85.250959),
(40.00507, -83.1186),
(39.91978, -82.93149),
(39.09757000000001, -84.51666),
(41.07995, -81.50389),
(41.63316, -83.54203000000003),
(39.82529, -83.03175999999998),
(39.83209, -82.99884),
(39.83208, -83.000447),
(38.416631, -84.861968),
(40.55549, -80.11634000000002),
(39.17736, -84.4873),
(41.484589, -81.691778),
(41.22734000000001, -81.62705),
(39.04695, -84.57498000000002),
(39.25595, -84.44287),
(39.8888, -82.88181999999998),
(39.74729, -84.21426),
(39.97415, -83.09486),
(39.08272, -84.52264),
(39.949808, -83.02289300000002),
(39.95141, -83.0137),
(38.7056, -80.66107),
(39.1479, -84.53929000000002),
(41.7573, -83.48075),
(39.98946, -85.9299),
(39.98894, -83.025012),
(41.110996, -80.82384),
(41.68685900000001, -83.554005),
(41.53907, -81.63354),
(39.300562, -84.518869),
(41.202543, -85.7011),
(39.19288, -84.47723),
(40.07297, -83.1347),
(39.140991, -84.483618),
(41.121599, -80.789503),
(41.31438, -81.51286),
(39.85922, -84.27762),
(40.42005, -80.03531),
(39.9589, -82.98257),
(39.89087, -83.85382),
(39.89862, -83.85353),
(40.03938, -83.050567),
(41.41218, -81.61371),
(38.306159, -80.833765),
(40.104, -82.94154),
(   , -85.62212),
```

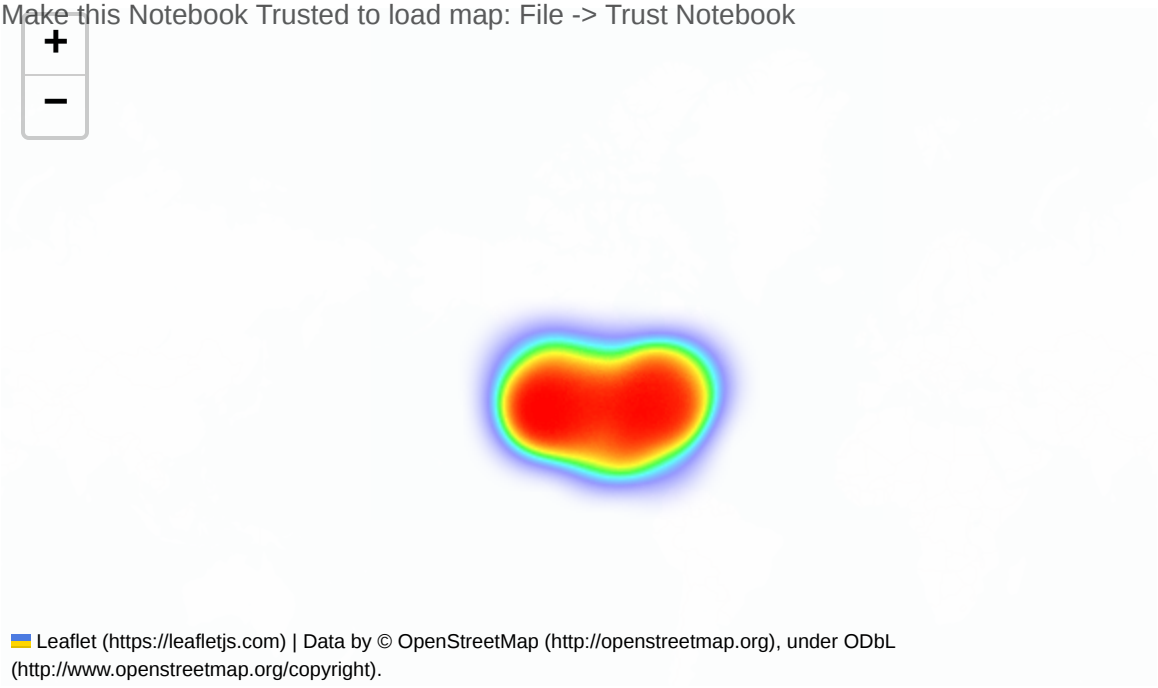
```
(41.55264, -81.60032),
(39.99226, -82.98517),
(38.31553, -85.58313000000003),
(38.400331, -85.371049),
(38.31251, -85.57741999999998),
(38.400331, -85.371049),
(39.88938, -82.88316),
(38.15873, -80.90504),
(39.9745, -83.09654),
(41.47487, -81.72095),
(39.26931, -84.4307),
(38.2201, -85.82223),
(41.48339, -81.66297),
(39.16161, -84.51375),
(41.42734, -82.08322),
(41.52631, -83.62123000000003),
(39.91732, -83.012335),
(41.60867, -81.48891),
(41.20554, -85.17116),
(41.507189, -81.67124),
(39.97415, -83.09486),
(39.97043, -83.01961999999997),
(39.93849, -82.84849),
(39.89698, -83.03428000000002),
(39.71452, -84.22281),
(40.11468, -83.01644),
(38.34244, -85.80447),
(41.47036, -81.65984),
(38.30056, -85.84703),
(39.169389, -84.50389),
(39.936838, -83.042661),
(39.937567, -83.04209200000003),
(39.9425, -82.53536),
(39.27125, -84.35176),
(41.398877, -83.616108),
(39.24422, -84.30224),
(40.412997, -80.07973299999998),
(39.021521, -84.50784300000002),
(41.12367, -80.76913),
(41.073876, -83.66091999999998),
(39.01992, -83.65886),
(38.97914, -83.65465),
(40.09927, -83.15793000000002),
(39.93658, -83.00941999999998),
(39.11543, -84.50003000000002),
(41.474552, -81.711522),
(39.78517, -84.18457),
(41.47974, -81.6667),
(38.27719, -85.69383),
(38.495124, -85.771118),
(39.73277, -84.20522),
(41.458, -83.62243000000002),
(39.045381, -84.462312),
(41.526, -81.65974),
(40.78725, -81.38592),
                    5.75495),
```

```
(39.02575, -84.61272),
(40.90923, -81.79765),
(40.91649, -81.78869),
(39.10051, -80.39199),
(40.082643, -82.91367199999998),
(41.47162, -81.90626),
(40.08155, -82.91859000000002),
(40.07976, -82.92689),
(38.27511, -85.81238),
(39.02908, -84.53565),
(39.04455, -84.57798000000003),
(39.83026, -81.45434),
(39.81436, -81.46898),
(38.41336, -82.3483),
(38.20674, -85.74846),
(38.10937, -85.70123000000002),
(40.26758, -82.92745),
(38.22285, -85.57436),
(40.43827, -80.00527),
(39.08272, -84.52264),
(40.046599, -85.99359),
(41.67904, -83.5732),
(40.52225, -80.06666),
(41.041763, -81.3932),
(39.02541, -85.88784),
(39.887572, -84.047499),
(39.90803, -84.03969000000002),
(39.8181, -81.45925),
(39.8181, -81.45925),
(39.81447, -81.46769),
(41.47307, -80.36586),
(41.460561, -80.366376),
(39.75022, -84.19668),
(39.16161, -84.51375),
(39.82329, -84.166569),
(39.818751, -84.167637),
(39.15267, -84.5395),
(40.480209, -85.719799),
(39.02676, -84.61421),
(40.480209, -85.719799),
(40.48013, -85.72942900000002),
(38.24863, -85.70467),
(39.24907, -84.44561999999998),
(39.18364, -84.48373000000002),
(41.420898, -81.69322),
(39.18976, -84.26285),
(40.00921, -83.03149),
(41.12369, -80.75595),
(41.09705, -81.50012),
(41.03011, -81.40292099999998),
(39.9533, -83.0032),
(39.91404, -83.01729),
(39.11811, -84.49975),
(39.73277, -84.20522),
(41.47379, -81.69592),
                  81.69931),
```

```
 (39.62334, -81.83577),
 (39.98233, -82.98449000000002),
 (40.15106, -80.03243),
 (41.66332, -83.56385999999998),
 (39.35245, -84.37507),
 (41.67178, -83.6939),
 (38.38018, -82.60993),
 (39.03347, -84.603),
 (40.42116, -80.0438),
 (41.341147, -83.347543),
 (41.34131, -83.376561),
 (40.26758, -82.92745),
 (40.432459, -80.023538),
 (39.9912, -85.9188),
 (39.05654000000001, -84.54283000000002),
 (38.39328, -85.76223),
 (39.97165800000001, -80.017133),
 (39.98381, -80.01011),
 (39.84457, -85.51639399999998),
 (38.399528, -85.76430500000002),
 (38.38981, -81.76937),
 (41.47965, -81.66758),
 (39.97859, -82.9763),
 (40.68109000000001, -80.24616999999998),
 (41.47487, -81.72095),
 (40.68109000000001, -80.24616999999998),
 (40.1412, -82.97121),
 (39.85942, -84.27778),
 (39.10838, -84.50296999999998),
 (39.32662, -84.42002),
 (39.24907, -84.44561999999998),
 (41.11959, -81.64568),
 (39.04455, -84.57798000000003),
 (38.26109, -85.73695),
 (38.21911, -85.50582),
 (39.94976, -83.04032),
 (38.82584, -120.029214),
 (37.358209, -121.840017),
 (37.881943, -122.307987),
 (37.881038, -122.307788),
 (38.518811, -121.101664),
 (38.518811, -121.101664),
 (36.9903, -119.71146),
 (37.42592, -122.09879),
 (37.75745, -122.21131),
 (37.31648, -121.96746),
 (37.44415, -122.2688),
 (37.71981, -121.65943),
 ...]
```

In [75]:
```
map = folium.Map()
HeatMap(zip(list(df.Start_Lat), list(df.Start_Lng))).add_to(map)
map
```

Out[75]: Make this Notebook Trusted to load map: File -> Trust Notebook



HeatMap shows the distribution of accidnets in cities of US according to their frequencies.

4. Temperature

In [90]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2845342 entries, 0 to 2845341
Data columns (total 50 columns):
 #   Column                 Dtype         
---  ------                 -----         
 0   ID                     object        
 1   Severity               int64         
 2   Start_Time             datetime64[ns]
 3   End_Time               object        
 4   Start_Lat              float64       
 5   Start_Lng              float64       
 6   End_Lat                float64       
 7   End_Lng                float64       
 8   Distance(mi)           float64       
 9   Description            object        
 10  Number                 float64       
 11  Street                 object        
 12  Side                   object        
 13  City                   object        
 14  County                 object        
 15  State                  object        
 16  Zipcode                object        
 17  Country                object        
 18  Timezone               object        
 19  Airport_Code           object        
 20  Weather_Timestamp      object        
 21  Temperature(F)         float64       
 22  Wind_Chill(F)          float64       
 23  Humidity(%)            float64       
 24  Pressure(in)           float64       
 25  Visibility(mi)         float64       
 26  Wind_Direction         object        
 27  Wind_Speed(mph)        float64       
 28  Precipitation(in)      float64       
 29  Weather_Condition      object        
 30  Amenity                bool          
 31  Bump                   bool          
 32  Crossing               bool          
 33  Give_Way               bool          
 34  Junction               bool          
 35  No_Exit                bool          
 36  Railway                bool          
 37  Roundabout             bool          
 38  Station                bool          
 39  Stop                   bool          
 40  Traffic_Calming        bool          
 41  Traffic_Signal         bool          
 42  Turning_Loop           bool          
 43  Sunrise_Sunset         object        
 44  Civil_Twilight         object        
 45  Nautical_Twilight      object        
 46  Astronomical_Twilight  object        
 47  Hour                   int64         
 48  Month                  int64         
 49  Year                   int64         
```

```
dtypes: bool(13), datetime64[ns](1), float64(13), int64(4), object(19)
memory usage: 838.5+ MB
```

In [94]:
```
df['Temperature(F)']
```

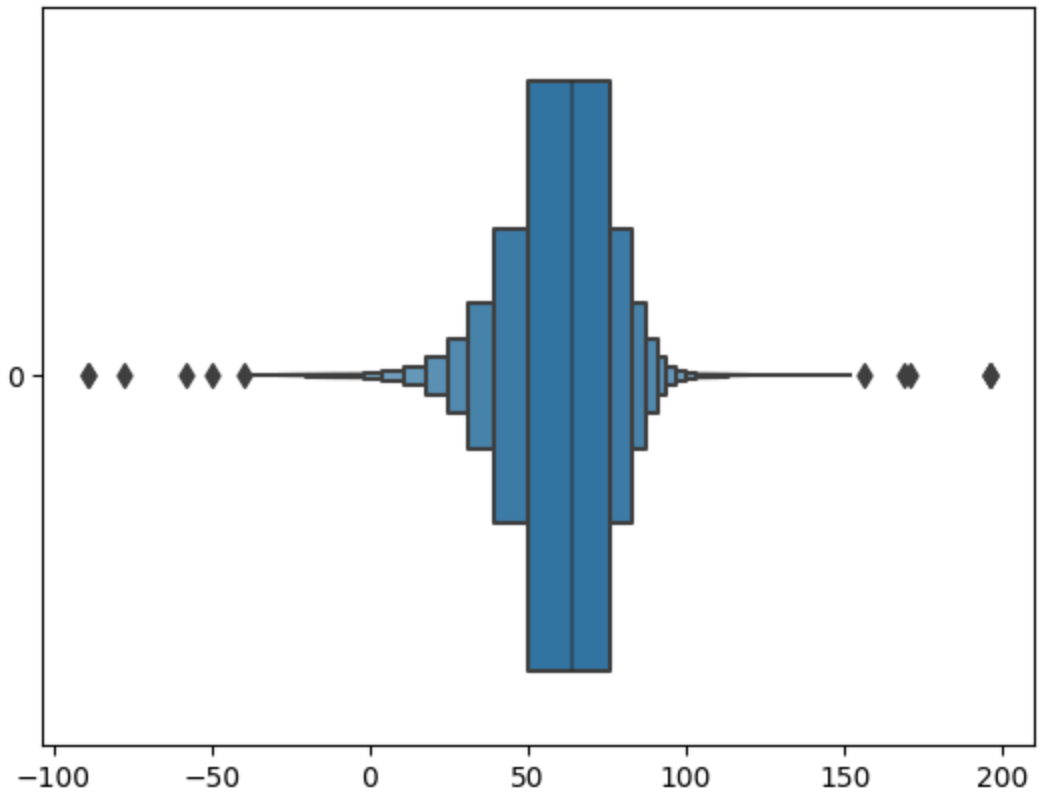
Out[94]:
```
0          42.1
1          36.9
2          36.0
3          39.0
4          37.0
           ...
2845337    86.0
2845338    70.0
2845339    73.0
2845340    71.0
2845341    79.0
Name: Temperature(F), Length: 2845342, dtype: float64
```

In [100…
```
temp_values = df['Temperature(F)'].value_counts()
temp_values
```

Out[100]:
```
 73.0     64505
 77.0     63575
 75.0     60534
 72.0     59681
 68.0     58557
          ...
 109.8        1
-9.8          1
 170.6        1
 107.2        1
 99.1         1
Name: Temperature(F), Length: 788, dtype: int64
```

In [105…
```
sns.boxenplot(df['Temperature(F)'], orient = 'h')
```

Out[105]:
```
<AxesSubplot: >
```



So, cities which experiencing the temperature between (50-60) Degree Fahrenheit, are more vulnerable to accidents.

5. Weather Condition

In [115…
```
len(df.Weather_Condition.unique())
```

Out[115]: 128

In [116…
```
df.Weather_Condition.unique()
```

Out[116]:
```
array(['Light Rain', 'Overcast', 'Mostly Cloudy', 'Snow', 'Light Snow',
       'Cloudy', nan, 'Scattered Clouds', 'Clear', 'Partly Cloudy',
       'Light Freezing Drizzle', 'Light Drizzle', 'Haze', 'Rain',
       'Heavy Rain', 'Fair', 'Drizzle', 'Fog', 'Thunderstorms and Rain',
       'Patches of Fog', 'Light Thunderstorms and Rain', 'Mist',
       'Rain Showers', 'Light Rain Showers', 'Heavy Drizzle', 'Smoke',
       'Light Freezing Fog', 'Light Freezing Rain', 'Blowing Snow',
       'Heavy Thunderstorms and Rain', 'Heavy Snow', 'Snow Grains',
       'Squalls', 'Light Fog', 'Shallow Fog', 'Thunderstorm',
       'Light Ice Pellets', 'Thunder', 'Thunder in the Vicinity',
       'Fair / Windy', 'Light Rain with Thunder',
       'Heavy Thunderstorms and Snow', 'Light Snow Showers',
       'Cloudy / Windy', 'Ice Pellets', 'N/A Precipitation',
       'Light Thunderstorms and Snow', 'T-Storm', 'Rain / Windy',
       'Wintry Mix', 'Partly Cloudy / Windy', 'Heavy T-Storm', 'Sand',
       'Light Rain / Windy', 'Widespread Dust', 'Mostly Cloudy / Windy',
       'Blowing Dust / Windy', 'Blowing Dust', 'Volcanic Ash',
       'Freezing Rain / Windy', 'Small Hail', 'Wintry Mix / Windy',
       'Light Snow / Windy', 'Heavy Ice Pellets', 'Heavy Snow / Windy',
       'Heavy Rain / Windy', 'Heavy T-Storm / Windy', 'Fog / Windy',
       'Dust Whirls', 'Showers in the Vicinity', 'Funnel Cloud',
       'Thunder / Windy', 'Snow / Windy', 'Haze / Windy',
       'Light Snow and Sleet', 'T-Storm / Windy',
       'Sand / Dust Whirlwinds', 'Light Snow with Thunder', 'Rain Shower',
       'Blowing Snow / Windy', 'Light Rain Shower', 'Snow and Sleet',
       'Drizzle and Fog', 'Light Sleet', 'Drizzle / Windy',
       'Light Snow Shower', 'Snow and Thunder / Windy',
       'Light Sleet / Windy', 'Smoke / Windy', 'Widespread Dust / Windy',
       'Light Drizzle / Windy', 'Tornado', 'Squalls / Windy', 'Hail',
       'Blowing Snow Nearby', 'Partial Fog', 'Sand / Windy',
       'Thunder / Wintry Mix', 'Light Freezing Rain / Windy', 'Duststorm',
       'Light Snow and Sleet / Windy', 'Heavy Rain Shower / Windy',
       'Sand / Dust Whirlwinds / Windy', 'Light Rain Shower / Windy',
       'Thunder and Hail', 'Freezing Rain', 'Heavy Sleet', 'Sleet',
       'Freezing Drizzle', 'Snow and Sleet / Windy',
       'Heavy Freezing Drizzle', 'Heavy Freezing Rain', 'Blowing Sand',
       'Thunder / Wintry Mix / Windy', 'Mist / Windy', 'Sleet / Windy',
       'Patches of Fog / Windy', 'Sand / Dust Whirls Nearby',
       'Heavy Rain Shower', 'Drifting Snow', 'Heavy Blowing Snow',
       'Low Drifting Snow', 'Light Blowing Snow', 'Heavy Rain Showers',
       'Light Haze', 'Heavy Thunderstorms with Small Hail',
       'Heavy Snow with Thunder', 'Thunder and Hail / Windy'],
      dtype=object)
```

In [110…
```
weather = df.Weather_Condition.value_counts()
weather
```

Out[110]:
```
Fair                          1107194
Mostly Cloudy                  363959
Cloudy                         348767
Partly Cloudy                  249939
Clear                          173823
                                ...
Sleet / Windy                       1
Mist / Windy                        1
Blowing Sand                        1
Heavy Freezing Rain                 1
Thunder and Hail / Windy            1
Name: Weather_Condition, Length: 127, dtype: int64
```

In [111…
```
weather.head(20)
```

Out[111]:
```
Fair                         1107194
Mostly Cloudy                 363959
Cloudy                        348767
Partly Cloudy                 249939
Clear                         173823
Light Rain                    128403
Overcast                       84882
Scattered Clouds               45132
Light Snow                     43752
Fog                            41226
Haze                           36354
Rain                           31044
Fair / Windy                   15195
Heavy Rain                     11824
Smoke                           7200
Light Drizzle                   7041
Thunder in the Vicinity         6944
Cloudy / Windy                  6839
T-Storm                         6546
Mostly Cloudy / Windy           6297
Name: Weather_Condition, dtype: int64
```

In [140…
```
weather[weather>80000]
```

Out[140]:
```
Fair             1107194
Mostly Cloudy     363959
Cloudy            348767
Partly Cloudy     249939
Clear             173823
Light Rain        128403
Overcast           84882
Name: Weather_Condition, dtype: int64
```

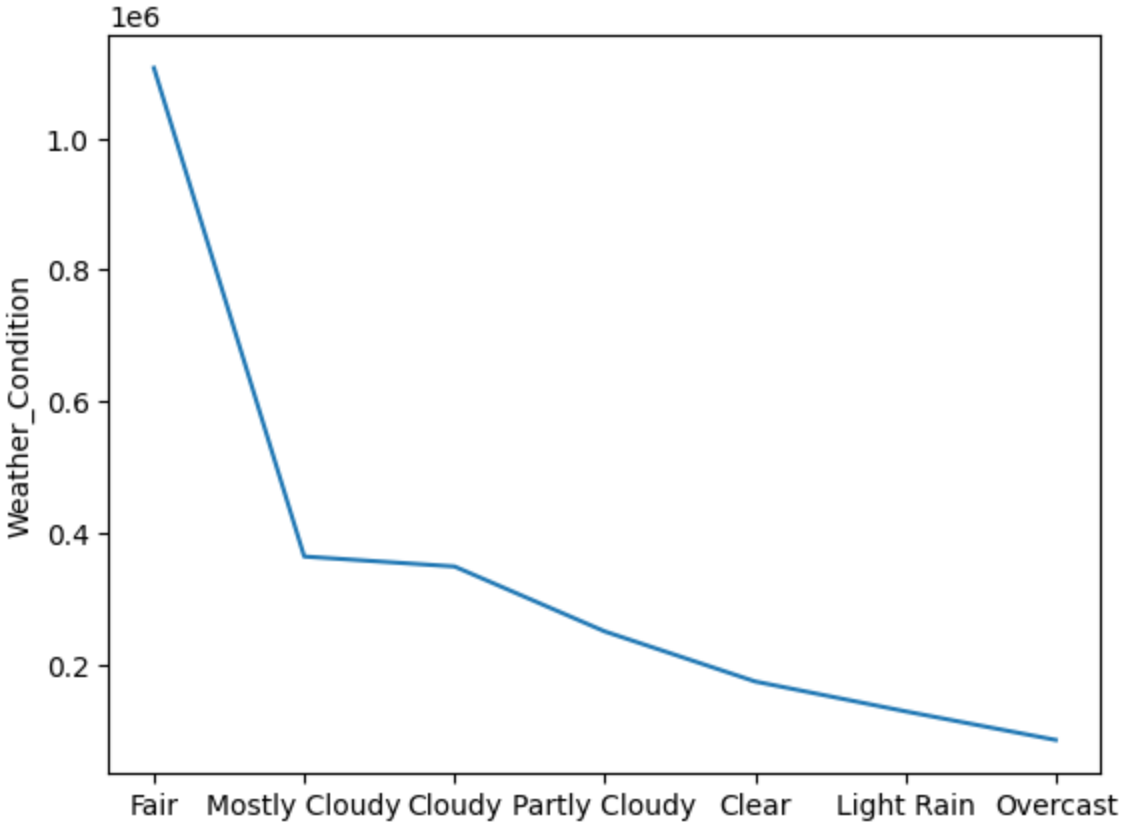
In [141…
```
Weather = pd.DataFrame(weather[weather>80000])
Weather
```

Out[141]:

	Weather_Condition
Fair	1107194
Mostly Cloudy	363959
Cloudy	348767
Partly Cloudy	249939
Clear	173823
Light Rain	128403
Overcast	84882

In [143…
```
sns.lineplot(x = Weather.index, y = Weather.Weather_Condition, data = Weathe
```

Out[143]:
```
<AxesSubplot: ylabel='Weather_Condition'>
```



__From above graph, it is clearly visisble that 'Fair' weather condition is responsible for huge number of accidents, followed by 'Mostly Cloudy' and 'Cloudy' condition of weather.__

In [ ]:
```
```

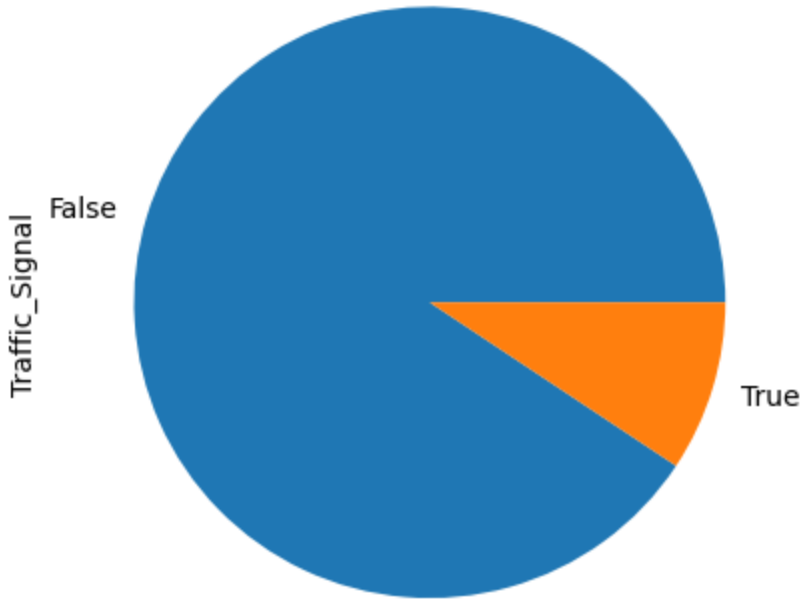
6. Traffic signal

In [147…
```
df.Traffic_Signal.unique()
```

Out[147]:
```
array([False,  True])
```

In [150…
```
df.Traffic_Signal.value_counts().plot(kind = "pie")
```

Out[150]:
```
<AxesSubplot: ylabel='Traffic_Signal'>
```



It is clear that, Poor traffic management is reponsible for huge number of accidets in US.

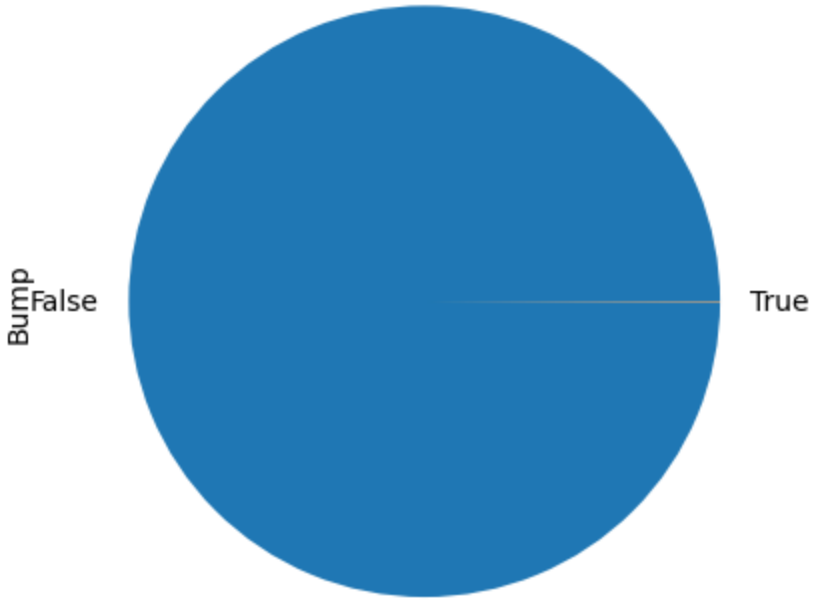
7. Bump

In [199…
```
df.Bump.value_counts()
```

Out[199]:
```
False    2844321
True        1021
Name: Bump, dtype: int64
```

In [200…
```
df.Bump.value_counts().plot(kind = 'pie')
```

Out[200]:
```
<AxesSubplot: ylabel='Bump'>
```



*The pie chart shows that there is obvious lack of speed breakers / bump in the neraby areas where accidents occured.

In [ ]:

Insights

1. Los Angeles, Miami, and Dallas are among Top-5 cities by accidents.

2. Among 11,000+ cities, Less than 5 % (actually, 4.2%) cities where accidents are above 1000 in the given time period.

3. Around 2000 cities, in which there was only 2 and less than 2 accidents in the interval of 4 years. - need more investigation

4. Accidents by cities follows exponentailly decreasing curve.

5. On working days, there are 2 peak hours. But on sundays (weekends), most of the accidenst are in afternoon period.

6. There is a missing data for some months, so we can't predict the month wise accidents accurately. But from the given data, it is visible that December has record high number of accidents.

7. In most of the accidents, weather was Fair.

8. Poor / Absence of accurate Traffic Signals and Bumps contribute highest accidents in the country.

Conclusion:

If we ignore other factors like temperature, weather conditions, start time, etc.- which are not in our hand or cannot be changed due to behaviour. The main cause of accidents are inadequate bumps and poor working of traffic signals.

The graph of Traffic signal and Bump is accuartely predicting that what is the major cause of accidents.

If we can improve the traffic signals and install signals of better technologies, with required number of bump in congested and accidents prone areas, then we can significantly reduce the number of accidents in US.

In [ ]: