In [29]:

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
```

In [7]:

```python
#Reading the CSV file
data = pd.read_csv('/users/youcefdjeddar/downloads/meteorite-landings.csv')
```

In [8]:

```python
data.head(15)
```

Out[8]:

	name	id	nametype	recclass	mass	fall	year	reclat	reclong	GeoLocation
0	Aachen	1	Valid	L5	21.0	Fell	1880.0	50.77500	6.08333	(50.775000, 6.083330)
1	Aarhus	2	Valid	H6	720.0	Fell	1951.0	56.18333	10.23333	(56.183330, 10.233330)
2	Abee	6	Valid	EH4	107000.0	Fell	1952.0	54.21667	-113.00000	(54.216670, -113.000000)
3	Acapulco	10	Valid	Acapulcoite	1914.0	Fell	1976.0	16.88333	-99.90000	(16.883330, -99.900000)
4	Achiras	370	Valid	L6	780.0	Fell	1902.0	-33.16667	-64.95000	(-33.166670, -64.950000)
5	Adhi Kot	379	Valid	EH4	4239.0	Fell	1919.0	32.10000	71.80000	(32.100000, 71.800000)
6	Adzhi-Bogdo (stone)	390	Valid	LL3-6	910.0	Fell	1949.0	44.83333	95.16667	(44.833330, 95.166670)
7	Agen	392	Valid	H5	30000.0	Fell	1814.0	44.21667	0.61667	(44.216670, 0.616670)
8	Aguada	398	Valid	L6	1620.0	Fell	1930.0	-31.60000	-65.23333	(-31.600000, -65.233330)
9	Aguila Blanca	417	Valid	L	1440.0	Fell	1920.0	-30.86667	-64.55000	(-30.866670, -64.550000)
10	Aioun el Atrouss	423	Valid	Diogenite-pm	1000.0	Fell	1974.0	16.39806	-9.57028	(16.398060, -9.570280)
11	Aïr	424	Valid	L6	24000.0	Fell	1925.0	19.08333	8.38333	(19.083330, 8.383330)
12	Aire-sur-la-Lys	425	Valid	Unknown	NaN	Fell	1769.0	50.66667	2.33333	(50.666670, 2.333330)
13	Akaba	426	Valid	L6	779.0	Fell	1949.0	29.51667	35.05000	(29.516670, 35.050000)
14	Akbarpur	427	Valid	H4	1800.0	Fell	1838.0	29.71667	77.95000	(29.716670, 77.950000)

In [9]:

```python
#See how many missing data points we have

missing_values_count = data.isnull().sum()
print(missing_values_count)
```

```
name              0
id                0
nametype          0
recclass          0
mass            131
fall              0
year            288
reclat         7315
reclong        7315
GeoLocation    7315
```

```
dtype: int64
```

In [10]:

```python
#Checking that the rows are unique
data['name'].is_unique
data['id'].is_unique
```

Out[10]:

```
True
```

In [11]:

```python
#Determining the percentage of missing values for each column

mass_missing_values = (missing_values_count['mass']/len(data.mass)) * 100
year_missing_values = (missing_values_count['year']/len(data.year)) * 100
location_missing_values = (missing_values_count['GeoLocation']/len(data.GeoLocation)) * 100

print('The percentage of missing values in the mass column is:', mass_missing_values,'%')
print ('The percentage of missing values in the year column is:', year_missing_values,'%')
print('The percentage of missing values in the location column is:', location_missing_values,'%')
```

```
The percentage of missing values in the mass column is: 0.28655175430921337 %
The percentage of missing values in the year column is: 0.6299763758859043 %
The percentage of missing values in the location column is: 16.0009624639076 %
```

In [12]:

```python
#Afer inspecting the data I realized that it was impossible to guess the value of the missing rows
```

In [13]:

```python
#Droping missing values
new_data = data.dropna()
```

In [14]:

```python
#Getting some information about our dataset
new_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 38116 entries, 0 to 45715
Data columns (total 10 columns):
name           38116 non-null object
id             38116 non-null int64
nametype       38116 non-null object
recclass       38116 non-null object
mass           38116 non-null float64
fall           38116 non-null object
year           38116 non-null float64
reclat         38116 non-null float64
reclong        38116 non-null float64
GeoLocation    38116 non-null object
dtypes: float64(4), int64(1), object(5)
memory usage: 3.2+ MB
```

In [15]:

```python
#Getting some statistical information as well
new_data.describe()
```

Out[15]:

	id	mass	year	reclat	reclong
count	38116.000000	3.811600e+04	38116.000000	38116.000000	38116.000000

	id	mass	year	reclat	reclong
mean	25343.110557	1.560031e+04	1989.957472	-39.594193	61.308320
std	17395.132894	6.286735e+05	26.444565	46.177476	80.776778
min	1.000000	0.000000e+00	601.000000	-87.366670	-165.433330
25%	10831.750000	6.630000e+00	1986.000000	-76.716670	0.000000
50%	21732.500000	2.909000e+01	1996.000000	-71.500000	35.666670
75%	39887.250000	1.874100e+02	2002.000000	0.000000	157.166670
max	57458.000000	6.000000e+07	2101.000000	81.166670	178.200000

In [16]:

```
new_missing_values_count = new_data.isnull().sum()
print(new_missing_values_count)
print(len(new_data))
```

```
name           0
id             0
nametype       0
recclass       0
mass           0
fall           0
year           0
reclat         0
reclong        0
GeoLocation    0
dtype: int64
38116
```

In [17]:

```
print("Columns in original dataset: %d \n" % data.shape[1])
print("Columns with na's dropped: %d" % new_data.shape[1])

#Here we can see that the columns have not been affected
```

```
Columns in original dataset: 10 

Columns with na's dropped: 10
```

In [18]:

```
percentage_of_rows_removed = 100 - ((len(new_data)/len(data)) * 100)
```

In [19]:

```
print('Percentage of missing data removed:', percentage_of_rows_removed,'%')
```

```
Percentage of missing data removed: 16.624376585878025 %
```

In [20]:

```
#Let's see how many meteorite classes the dataset has
meteorite_class = new_data['recclass'].value_counts()
print(meteorite_class)
```

```
L6                   7519
H5                   6243
H6                   3898
H4                   3880
L5                   3264
LL5                  2199
LL6                  1660
L4                    939
H4/5                  395
CM2                   330
H3                    313
```

```
CO3                  308
Iron, IIIAB          270
L3                   268
LL                   223
Ureilite             214
E3                   205
LL4                  198
CV3                  184
Howardite            179
Diogenite            178
Eucrite-pmict        169
H5/6                 166
CR2                  116
Eucrite              115
Iron, IIAB           111
Mesosiderite         107
H~5                  106
Iron, ungrouped      105
LL3                   88
                    ...
L/LL4-6                1
H/L3.6                 1
L3.0-3.7               1
LL3.1-3.5              1
L3.10                  1
H(L)3-an               1
EL6/7                  1
LL6                    1
H3                     1
Mesosiderite-B         1
Stone-ung              1
H3.2-6                 1
H/L3                   1
H/L3.9                 1
CH/CBb                 1
E5-an                  1
Pallasite?             1
L3.9/4                 1
EH                     1
EH3/4-an               1
LL7(?)                 1
H3.4-5                 1
Lodranite-an           1
L/LL3.10               1
L3.7-3.9               1
LL3.00                 1
EL4/5                  1
L(LL)5                 1
L3.3-3.7               1
L3-melt breccia        1
Name: recclass, Length: 422, dtype: int64
```

In [21]:

```
#It looks like we have 422 classes for approximately 40000 meteorites. An average of about 100 met
eorites per class
```

In [22]:

```
#Let's see if we can do better
```

In [23]:

```
print('Top 10 meteorite classes:', meteorite_class[:10].sum())
```

```
Top 10 meteorite classes: 30327
```

In [24]:

```
#Interesting: only 10 meteorite classes encompass around 80% of all the 38116 meteorites
```

In [25]:

In [25]:

```python
#Searching for outliers

new_data.loc[new_data['mass'] == max(new_data.mass)]
```

Out[25]:

	name	id	nametype	recclass	mass	fall	year	reclat	reclong	GeoLocation
16383	Hoba	11890	Valid	Iron, IVB	60000000.0	Found	1920.0	-19.58333	17.91667	(-19.583330, 17.916670)

In [32]:

```python
#---------------------------------------------------------------------------------------------------
---------#
```