In [3]:
```python
#USA weather events 2016-2020(kaggle dataset)
#import necessaary libraries

import pandas as pd                                   #for data analysis
import numpy as np                                    #mathematical operations
import matplotlib.pyplot as plt                       #data visualization
import seaborn as sns                                 #a data visualization library
import matplotlib.cm as cm
import matplotlib.colors as colors
```

In [4]:
```python
pip install folium
```

```
Requirement already satisfied: folium in ./opt/anaconda3/lib/python3.8/site-packages (0.12.1)
Requirement already satisfied: numpy in ./opt/anaconda3/lib/python3.8/site-packages (from folium) (1.20.1)
Requirement already satisfied: branca>=0.3.0 in ./opt/anaconda3/lib/python3.8/site-packages (from folium) (0.4.2)
Requirement already satisfied: requests in ./opt/anaconda3/lib/python3.8/site-packages (from folium) (2.25.1)
Requirement already satisfied: jinja2>=2.9 in ./opt/anaconda3/lib/python3.8/site-packages (from folium) (2.11.3)
Requirement already satisfied: MarkupSafe>=0.23 in ./opt/anaconda3/lib/python3.8/site-packages (from jinja2>=2.9->folium) (1.1.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in ./opt/anaconda3/lib/python3.8/site-packages (from requests->folium) (1.26.4)
Requirement already satisfied: idna<3,>=2.5 in ./opt/anaconda3/lib/python3.8/site-packages (from requests->folium) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in ./opt/anaconda3/lib/python3.8/site-packages (from requests->folium) (2020.12.5)
Requirement already satisfied: chardet<5,>=3.0.2 in ./opt/anaconda3/lib/python3.8/site-packages (from requests->folium) (4.0.0)
Note: you may need to restart the kernel to use updated packages.
```

In [5]:
```python
import folium
from folium import plugins                            # A Python library used for visualizing geospatial data
```

In [6]:
```python
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [10]:
```python
df = pd.read_csv("/Users/mrinmayeedhumane/Desktop/My_Projects/USA WeatherEvents/WeatherEvents_Jan2016-Dec2020.csv")
```

In [11]:
```python
df.head()
```

Out[11]:

	EventId	Type	Severity	StartTime(UTC)	EndTime(UTC)	TimeZone	AirportCode	LocationLat	LocationLng	City	County	State	ZipCode
0	W-1	Snow	Light	2016-01-06 23:14:00	2016-01-07 00:34:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
1	W-2	Snow	Light	2016-01-07 04:14:00	2016-01-07 04:54:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
2	W-3	Snow	Light	2016-01-07 05:54:00	2016-01-07 15:34:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
3	W-4	Snow	Light	2016-01-08 05:34:00	2016-01-08 05:54:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
4	W-5	Snow	Light	2016-01-08 13:54:00	2016-01-08 15:54:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0

In [12]:
```python
df['Type'].value_counts()
```

Out[12]:
```
Rain             3752341
Fog              1385264
Snow              827555
Cold              169182
Precipitation      96684
Storm              40553
Hail                2627
Name: Type, dtype: int64
```

In [13]:
```python
df["Severity"].value_counts()
```

Out[13]:
```
Light       3820970
Severe      1167203
Moderate    1020399
Heavy        166323
UNK           96684
Other          2627
Name: Severity, dtype: int64
```

In [14]:
```python
df["AirportCode"].value_counts()
```

Out[14]:
```
K3TH    11142
KMLP    10883
KHYW     9268
K0CO     8242
KSMP     7996
        ...
KCZZ       43
K4HV       27
KCQV       25
KSPL        8
KI39        1
Name: AirportCode, Length: 2071, dtype: int64
```

In [15]:
```python
#Here we nor not interested in the records which consist "UNK(i.e. Unknown)" and "Other" in the severity column,
#hence we are going to discard those rows


dataset = df[(df['Severity'] != 'UNK') & (df['Severity'] != 'Other')]
```

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

Out[16]:

	EventId	Type	Severity	StartTime(UTC)	EndTime(UTC)	TimeZone	AirportCode	LocationLat	LocationLng	City	County	State	ZipCode
0	W-1	Snow	Light	2016-01-06 23:14:00	2016-01-07 00:34:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
1	W-2	Snow	Light	2016-01-07 04:14:00	2016-01-07 04:54:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
2	W-3	Snow	Light	2016-01-07 05:54:00	2016-01-07 15:34:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
3	W-4	Snow	Light	2016-01-08 05:34:00	2016-01-08 05:54:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0
4	W-5	Snow	Light	2016-01-08 13:54:00	2016-01-08 15:54:00	US/Mountain	K04V	38.0972	-106.1689	Saguache	Saguache	CO	81149.0

In [17]:
```python
dataset_types = dataset[['AirportCode','Type']]     #create table which consists of given two column from main dataset
dataset_types.head(7)
```

Out[17]:

	AirportCode	Type
0	K04V	Snow
1	K04V	Snow
2	K04V	Snow
3	K04V	Snow
4	K04V	Snow
5	K04V	Snow
6	K04V	Fog

In [18]:
```python
#group the airport_code as per occurrence occurrence of type of weather events


#convert the categorical variables in column 'Type' into indicator/dummy variables

types = pd.get_dummies(dataset_types['Type'])
types['AirportCode'] = dataset_types['AirportCode']

types = types.groupby('AirportCode').sum().reset_index()

types.head()
```

Out[18]:

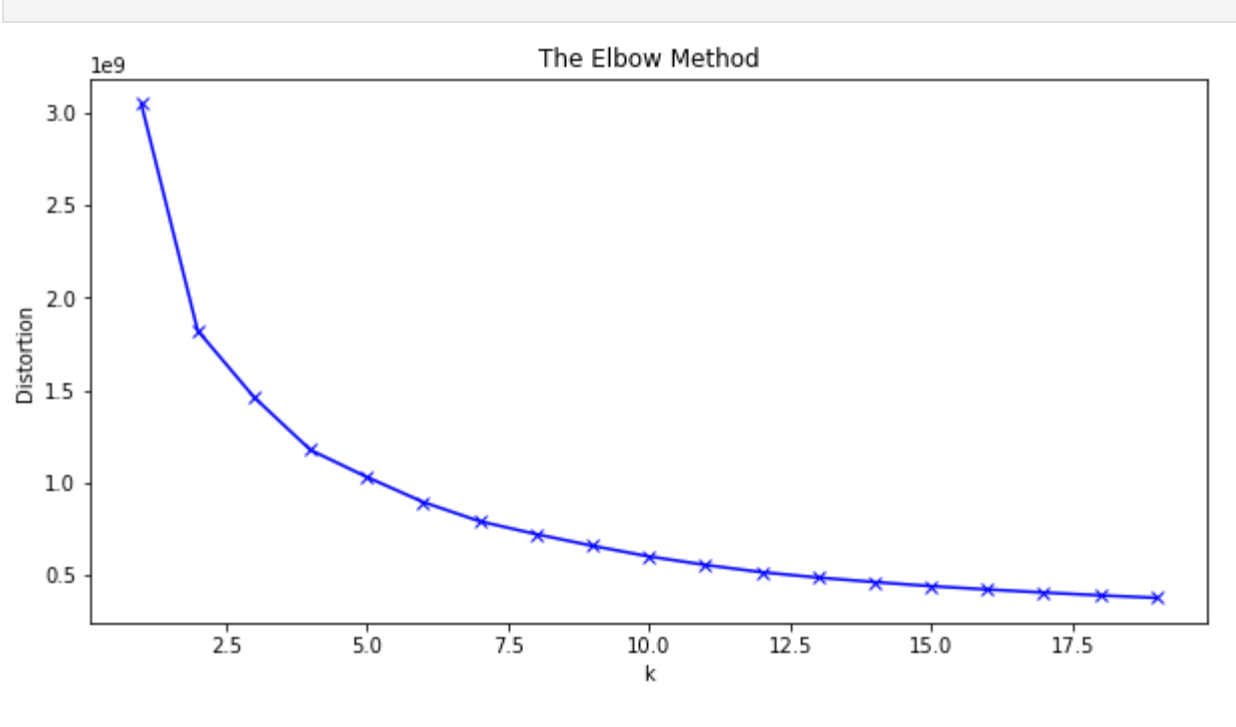
	AirportCode	Cold	Fog	Rain	Snow	Storm
0	K01M	22.0	964.0	2507.0	58.0	0.0
1	K04V	20.0	67.0	1152.0	670.0	57.0
2	K04W	103.0	503.0	1717.0	946.0	1.0
3	K06D	612.0	763.0	848.0	1984.0	85.0
4	K08D	749.0	1214.0	701.0	1090.0	48.0

In [19]:
```python
#k-means clustering


codes =types[['AirportCode']]
types.drop('AirportCode', axis=1, inplace=True)
```

In [20]:
```python
#import kmeans algorith from scikitlearn librsry
from sklearn.cluster import KMeans
from sklearn import datasets
from sklearn.decomposition import PCA
```

In [21]:
```python
distortions = []

K = range(1, 20)
for k in K:
    kmean = KMeans(n_clusters=k, random_state=0, n_init = 50, max_iter = 500)
    kmean.fit(types)
    distortions.append(kmean.inertia_)
```

In [22]:
```python
plt.figure(figsize=(10,5))
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method')
plt.show()
```



In [23]:
```python
kmeans = KMeans(n_clusters=5, random_state=0).fit(types)

codes['cluster'] = kmeans.labels_
codes.head()
```

```
<ipython-input-23-23da3e86526f>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  codes['cluster'] = kmeans.labels_
```

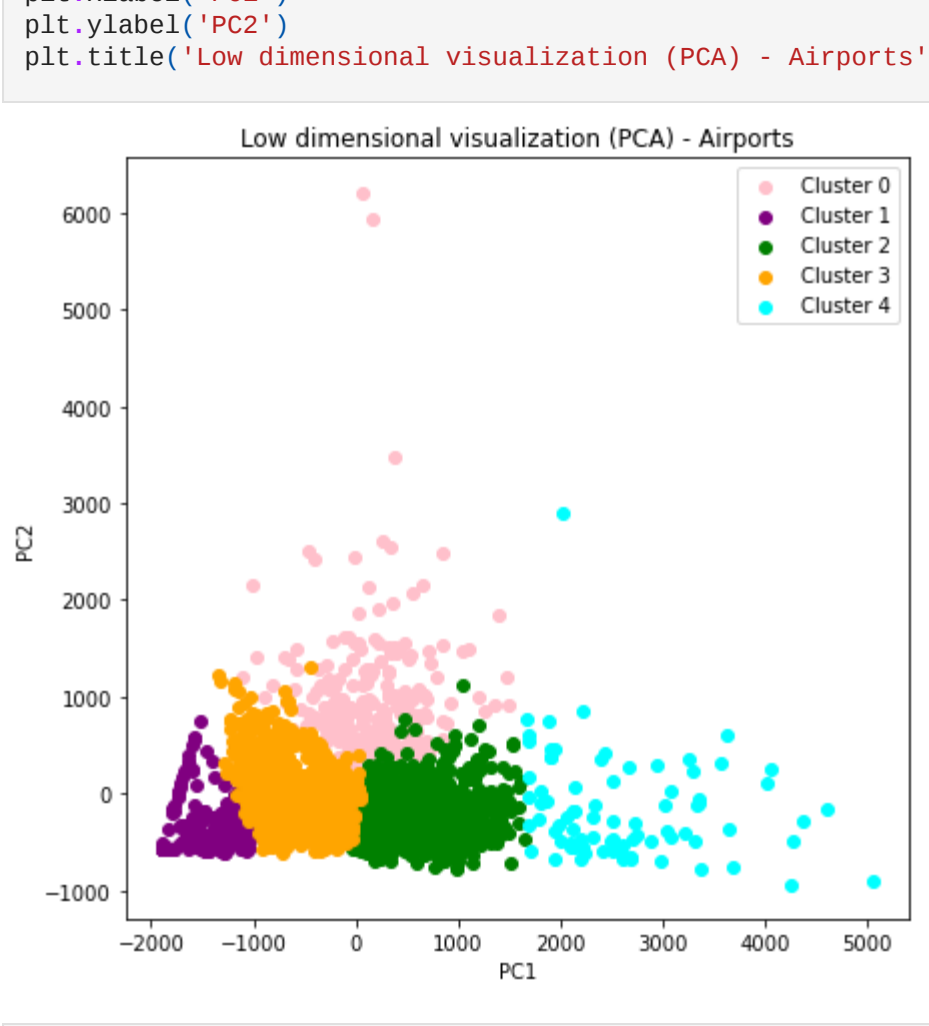
Out[23]:

	AirportCode	cluster
0	K01M	2
1	K04V	3
2	K04W	0
3	K06D	0
4	K08D	3

In [24]:
```python
pca = PCA().fit(types)
pca_types = pca.transform(types)
print("Variance explained by each component (%): ")
for i in range(len(pca.explained_variance_ratio_)):
      print("\n",i+1,":", pca.explained_variance_ratio_[i]*100)
print("Total sum (%): ",sum(pca.explained_variance_ratio_)*100)
print("Explained variance of the first two components (%): ",sum(pca.explained_variance_ratio_[0:1])*100)
```

```
Variance explained by each component (%):

 1 : 65.44042945575798

 2 : 20.029216969696844

 3 : 12.850764249575597

 4 : 1.176583866549171

 5 : 0.5030054584204038
Total sum (%):  100.0
Explained variance of the first two components (%):  65.44042945575798
```

In [25]:
```python
c0 = []
c1 = []
c2 = []
c3 = []
c4 = []

for i in range(len(pca_types)):
    if kmeans.labels_[i] == 0:
        c0.append(pca_types[i])
    if kmeans.labels_[i] == 1:
        c1.append(pca_types[i])
    if kmeans.labels_[i] == 2:
        c2.append(pca_types[i])
    if kmeans.labels_[i] == 3:
        c3.append(pca_types[i])
    if kmeans.labels_[i] == 4:
        c4.append(pca_types[i])


c0 = np.array(c0)
c1 = np.array(c1)
c2 = np.array(c2)
c3 = np.array(c3)
c4 = np.array(c4)

plt.figure(figsize=(7,7))
plt.scatter(c0[:,0], c0[:,1], c='pink', label='Cluster 0')
plt.scatter(c1[:,0], c1[:,1], c='purple', label='Cluster 1')
plt.scatter(c2[:,0], c2[:,1], c='green', label='Cluster 2')
plt.scatter(c3[:,0], c3[:,1], c='orange', label='Cluster 3')
plt.scatter(c4[:,0], c4[:,1], c='cyan', label='Cluster 4')
plt.legend()
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('Low dimensional visualization (PCA) - Airports');
```



In [26]:
```python
types['cluster']  = kmeans.labels_

types.head()
```

Out[26]:

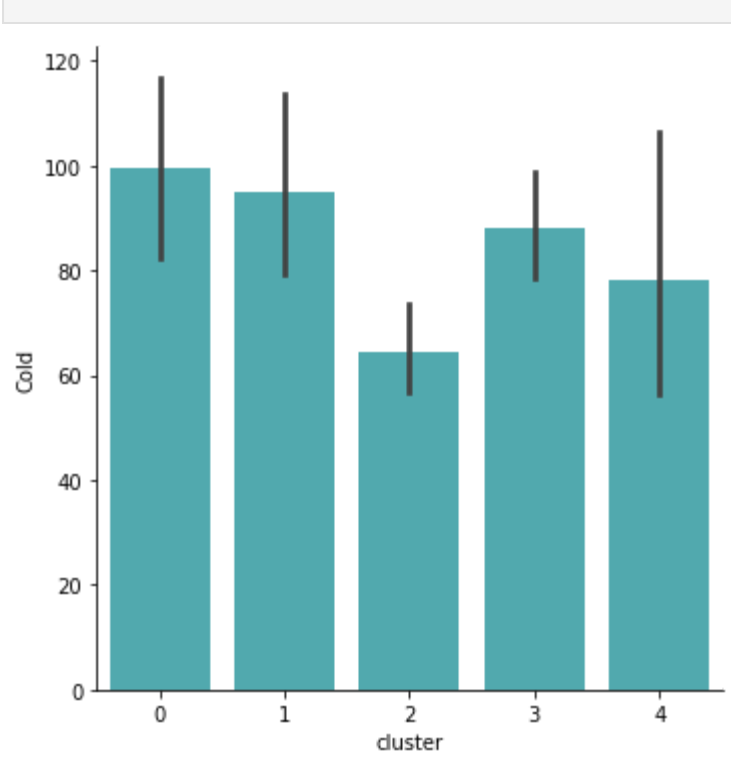
	Cold	Fog	Rain	Snow	Storm	cluster
0	22.0	964.0	2507.0	58.0	0.0	2
1	20.0	67.0	1152.0	670.0	57.0	3
2	103.0	503.0	1717.0	946.0	1.0	0
3	612.0	763.0	848.0	1984.0	85.0	0
4	749.0	1214.0	701.0	1090.0	48.0	3

In [27]:
```python
types.groupby('cluster').mean()
```

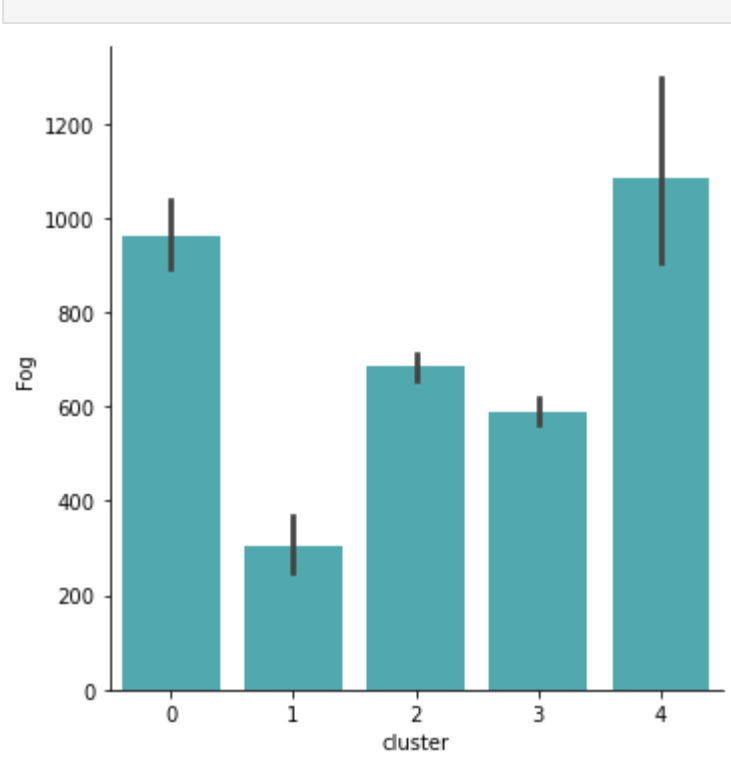
Out[27]:

	Cold	Fog	Rain	Snow	Storm
cluster					
0	99.634675	963.328173	1946.795666	1206.074303	19.040248
1	95.140496	306.239669	311.516529	53.082645	25.789256
2	64.562254	684.904325	2475.938401	180.039318	5.678899
3	88.291541	590.051360	1218.672205	413.315710	34.069486
4	78.325000	1085.037500	4402.962500	177.025000	15.937500

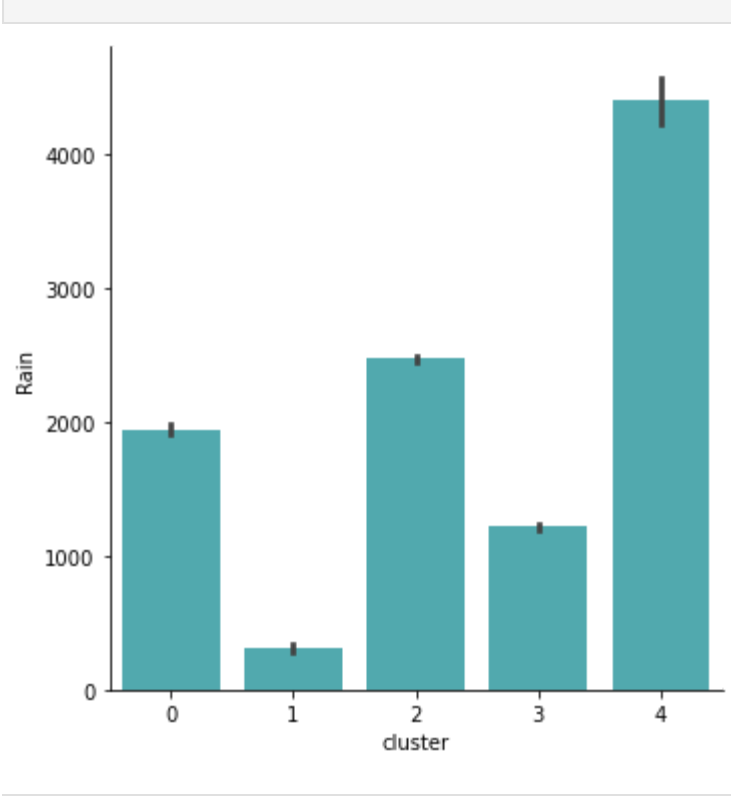
In [28]:
```python
sns.catplot(x='cluster', y='Cold', data=types, kind='bar',  color= '#42b7bd');
```



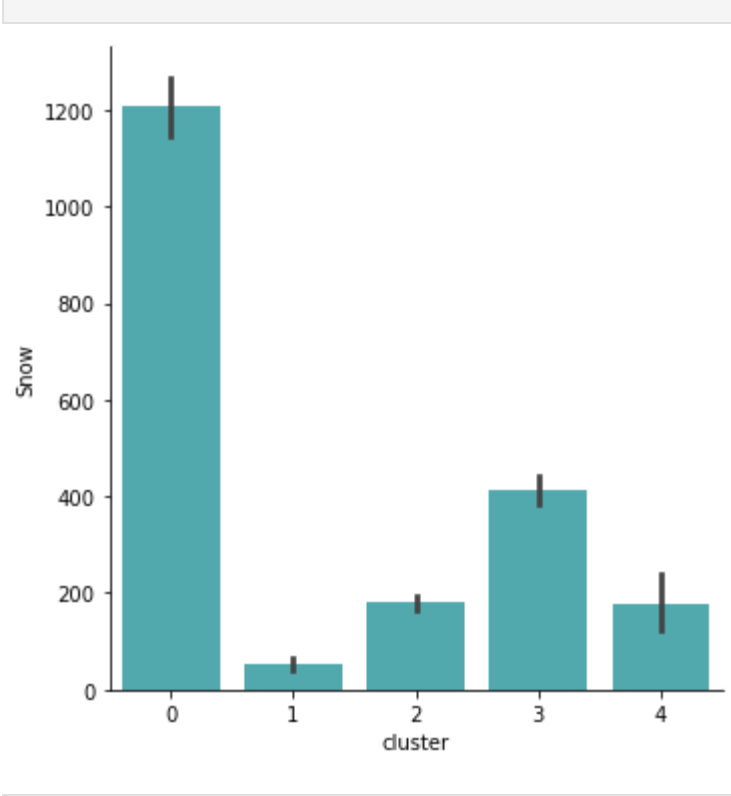
In [29]:
```python
sns.catplot(x='cluster', y='Fog', data=types, kind='bar', color= '#42b7bd');
```



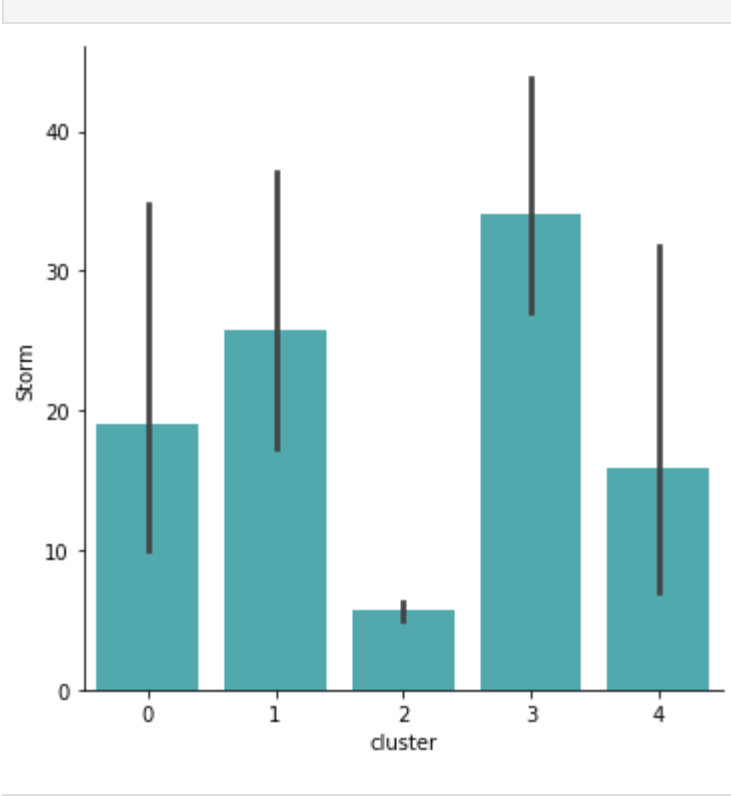
In [30]:
```python
sns.catplot(x='cluster', y='Rain', data=types, kind='bar', color= '#42b7bd');
```



In [31]:
```python
sns.catplot(x='cluster', y='Snow', data=types, kind='bar', color= '#42b7bd');
```

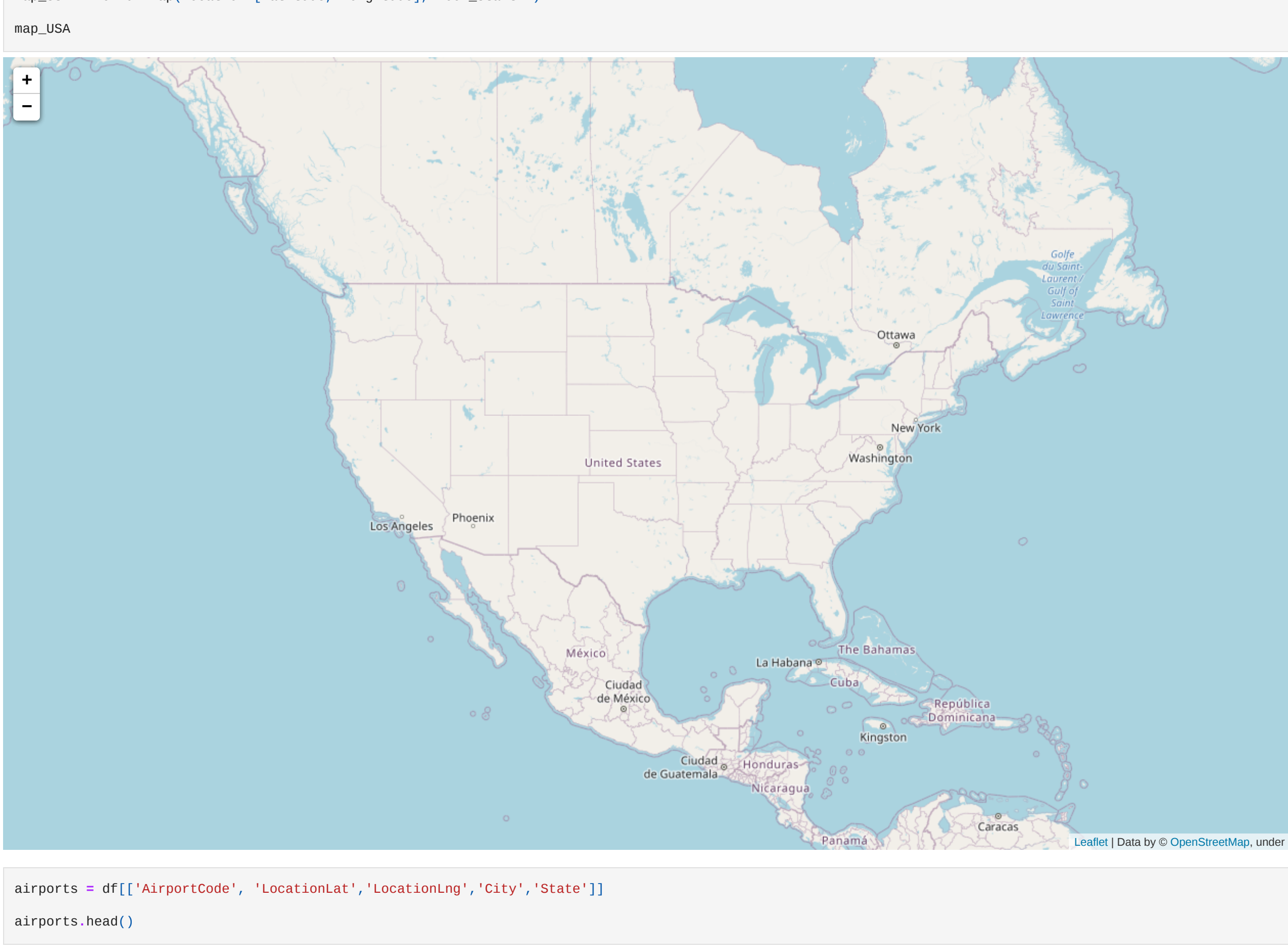


In [32]:
```python
sns.catplot(x='cluster', y='Storm', data=types, kind='bar', color= '#42b7bd');
```



In [33]:
```python
latitude = 38.500000
longitude = -95.665

map_USA = folium.Map(location=[latitude, longitude], zoom_start=4)

map_USA
```

Out[33]:



In [34]:
```python
airports = df[['AirportCode', 'LocationLat','LocationLng','City','State']]

airports.head()
```

Out[34]:

	AirportCode	LocationLat	LocationLng	City	State
0	K04V	38.0972	-106.1689	Saguache	CO
1	K04V	38.0972	-106.1689	Saguache	CO
2	K04V	38.0972	-106.1689	Saguache	CO
3	K04V	38.0972	-106.1689	Saguache	CO
4	K04V	38.0972	-106.1689	Saguache	CO

In [35]:
```python
number_of_occurences = pd.DataFrame(airports['AirportCode'].value_counts())
number_of_occurences.reset_index(inplace=True)
number_of_occurences.columns = ['AirportCode', 'Count']
number_of_occurences.head()
```

Out[35]:

	AirportCode	Count
0	K3TH	11142
1	KMLP	10883
2	KHYW	9268
3	K0CO	8242
4	KSMP	7996

In [36]:
```python
number_of_occurences = number_of_occurences.merge(airports.drop_duplicates())

number_of_occurences = number_of_occurences.merge(codes)

number_of_occurences.head()
```

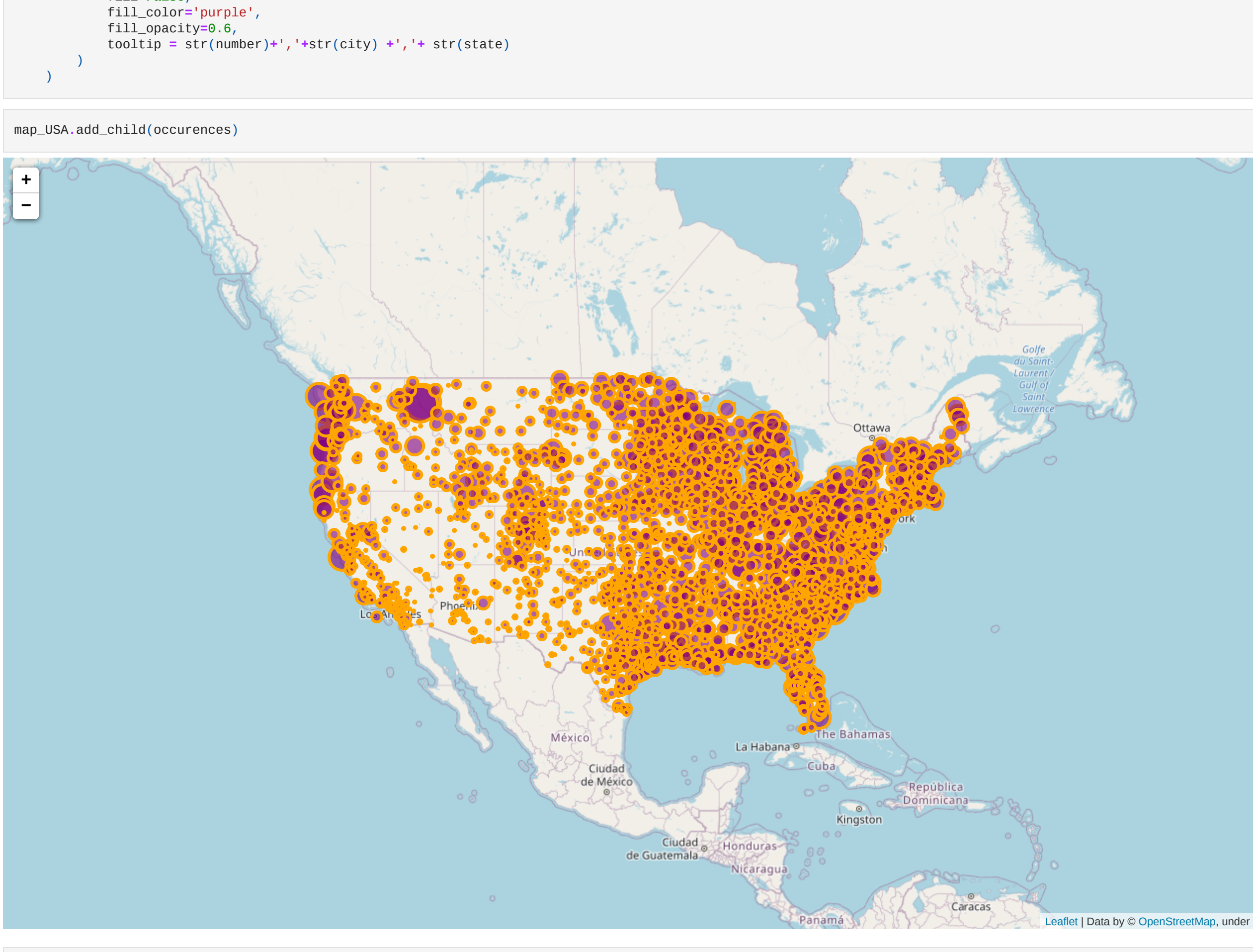
Out[36]:

	AirportCode	Count	LocationLat	LocationLng	City	State	cluster
0	K3TH	11142	47.6000	-115.3667	Thompson Falls-West End	MT	0
1	KMLP	10883	47.4542	-115.6697	Mullan	ID	0
2	KHYW	9268	33.8285	-79.1222	Conway	SC	4
3	K0CO	8242	39.7945	-105.7639	Granby	CO	0
4	KSMP	7996	47.2765	-121.3382	Enumclaw	WA	0

In [37]:
```python
occurences = folium.map.FeatureGroup()
n_mean = number_of_occurences['Count'].mean()

for lat, lng, number, city, state in zip(number_of_occurences['LocationLat'],
                                         number_of_occurences['LocationLng'],
                                         number_of_occurences['Count'],
                                         number_of_occurences['City'],
                                         number_of_occurences['State'],):
    occurences.add_child(
        folium.vector_layers.CircleMarker(
            [lat, lng],
            radius=number/n_mean*5, # define how big you want the circle markers to be
            color='orange',
            fill=False,
            fill_color='purple',
            fill_opacity=0.6,
            tooltip = str(number)+','+str(city) +','+ str(state)
        )
    )
```

In [38]:
```python
map_USA.add_child(occurences)
```

Out[38]:



In [ ]: