```
import numpy as np
import pandas as pd
```

```
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.7/dist-packages (1.5.12)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.7/dist-packages (from kaggle) (5.0.2)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.15.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from kaggle) (4.62.3)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from kaggle) (2021.5.30)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from kaggle) (2.23.0)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.24.3)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.7/dist-packages (from kaggle) (2.8.2)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.7/dist-packages (from python-slugify->kaggle) (1.3
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->kaggle) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->kaggle) (3.0.4)
```

```
! cp kaggle.json ~/kaggle.json
! chmod 600 ~/.kaggle/kaggle.json
```

```
! kaggle datasets download terenceshin/covid19s-impact-on-airport-traffic
```

```
covid19s-impact-on-airport-traffic.zip: Skipping, found more recently modified local copy (use --force to force download)
```

```
! unzip covid19s-impact-on-airport-traffic.zip
```

```
Archive:  covid19s-impact-on-airport-traffic.zip
replace covid_impact_on_airport_traffic.csv? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
```

```
#Import related libaries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
```

```
#Read data
df = pd.read_csv("/content/covid_impact_on_airport_traffic.csv")
```

```
#Show first five row of data
df.head()
```

	AggregationMethod	Date	Version	AirportName	PercentOfBaseline	Centroid	City	S
0	Daily	2020-04-03	1.0	Kingsford Smith	64	POINT(151.180087713813 -33.9459774986125)	Sydney	S V
1	Daily	2020-04-13	1.0	Kingsford Smith	29	POINT(151.180087713813 -33.9459774986125)	Sydney	S V
2	Daily	2020-07-10	1.0	Kingsford Smith	54	POINT(151.180087713813 -33.9459774986125)	Sydney	S V
3	Daily	2020-09-02	1.0	Kingsford Smith	18	POINT(151.180087713813 -33.9459774986125)	Sydney	S V

```
# Information about data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7247 entries, 0 to 7246
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   AggregationMethod  7247 non-null   object 
 1   Date               7247 non-null   object 
 2   Version            7247 non-null   float64
 3   AirportName        7247 non-null   object
 4   PercentOfBaseline  7247 non-null   int64
 5   Centroid           7247 non-null   object
 6   City               7247 non-null   object
 7   State              7247 non-null   object
 8   ISO_3166_2         7247 non-null   object
 9   Country            7247 non-null   object
 10  Geography          7247 non-null   object
dtypes: float64(1), int64(1), object(9)
memory usage: 622.9+ KB
```

```
# Check the null values
df.isnull().sum()
```

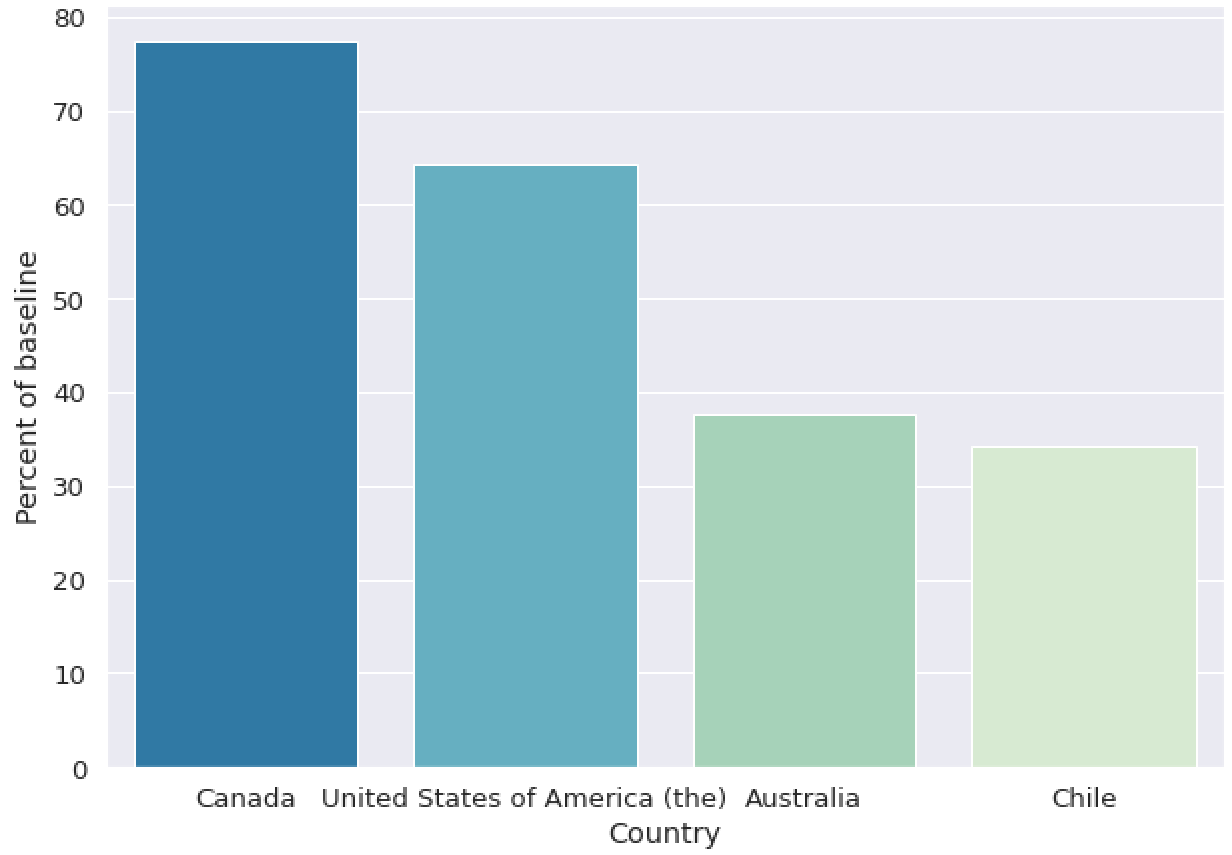
```
AggregationMethod    0
Date                 0
Version              0
AirportName          0
PercentOfBaseline    0
Centroid             0
City                 0
State                0
ISO_3166_2           0
Country              0
Geography            0
dtype: int64
```

```
df['Country'].unique()
```

```
array(['Australia', 'Chile', 'Canada', 'United States of America (the)'],
      dtype=object)
```

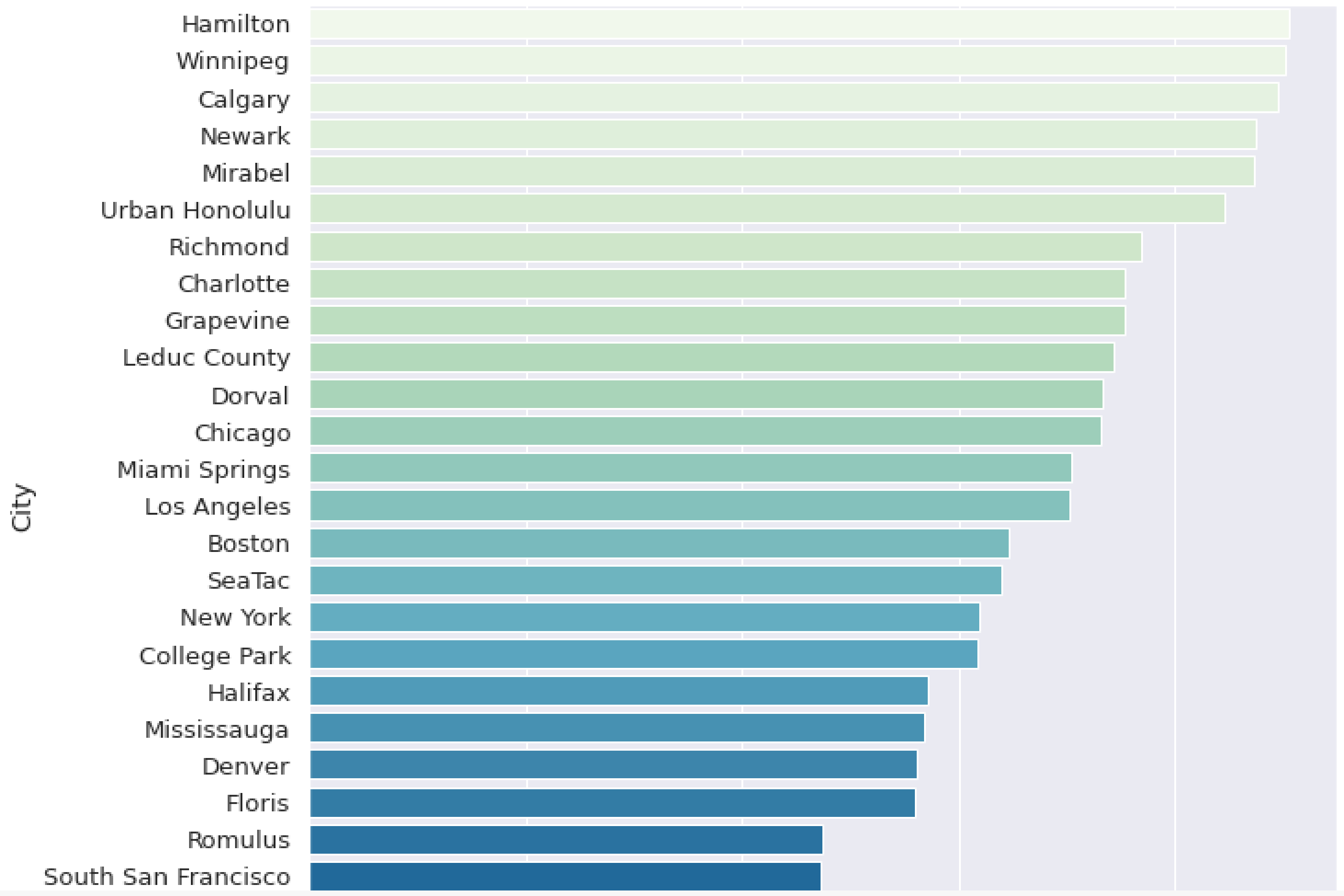
```
df1 = df.groupby("Country")['PercentOfBaseline'].mean().sort_values(ascending = False).reset_index()
sns.set(font_scale = 1.2)
plt.figure(figsize = [10,7])
sns.barplot(data = df1, x=  'Country', y = 'PercentOfBaseline',palette = 'GnBu_r')
plt.ylabel('Percent of baseline')
```

```
Text(0, 0.5, 'Percent of baseline')
```



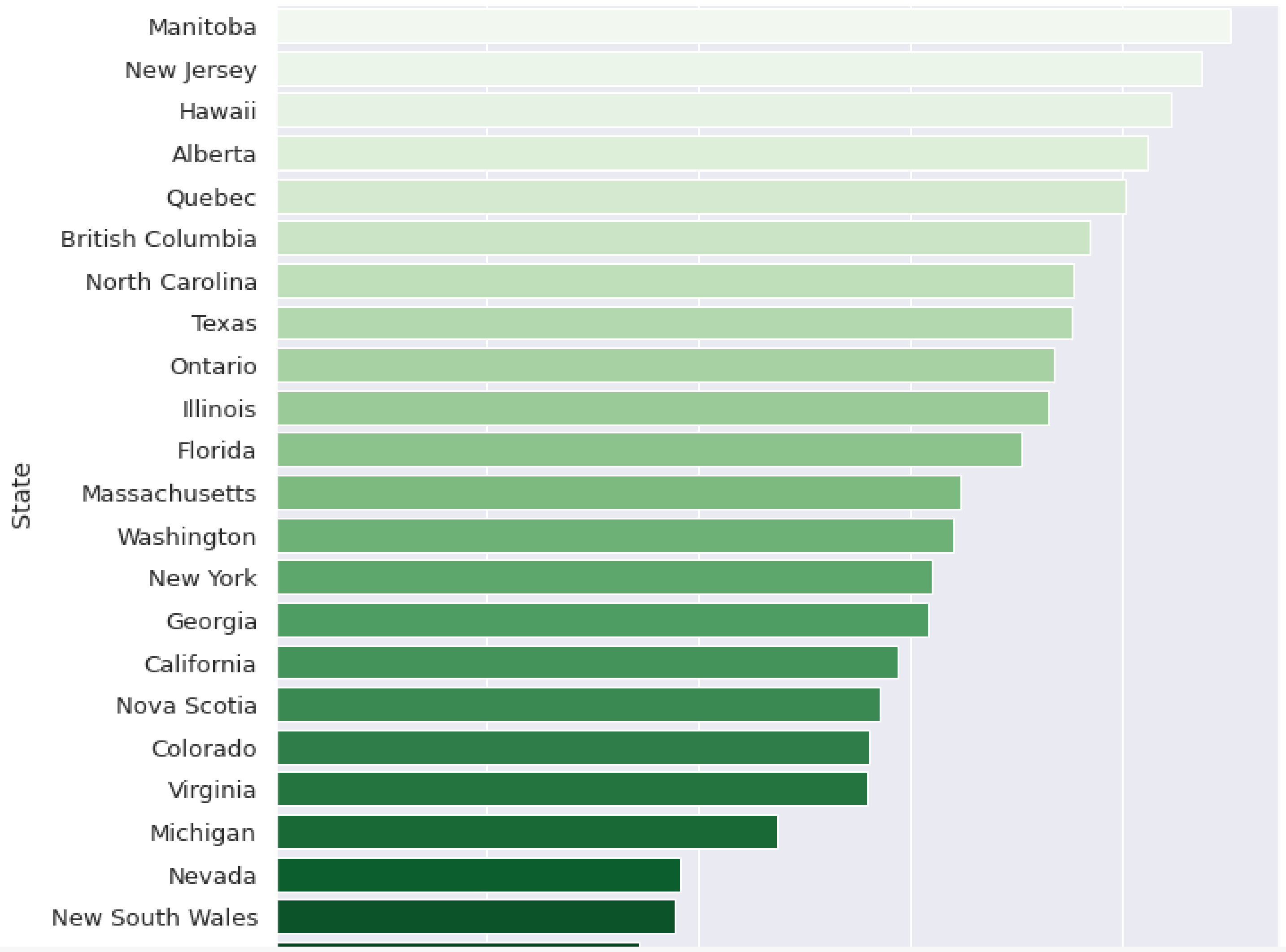
```
df1 = df.groupby("City")['PercentOfBaseline'].mean().sort_values(ascending = False).reset_index()
sns.set(font_scale = 1.2)
plt.figure(figsize = [10,10])
sns.barplot(data = df1, x = 'PercentOfBaseline', y ='City',palette = 'GnBu')
plt.xlabel("Percent of baseline")
```

```
Text(0.5, 0, 'Percent of baseline')
```



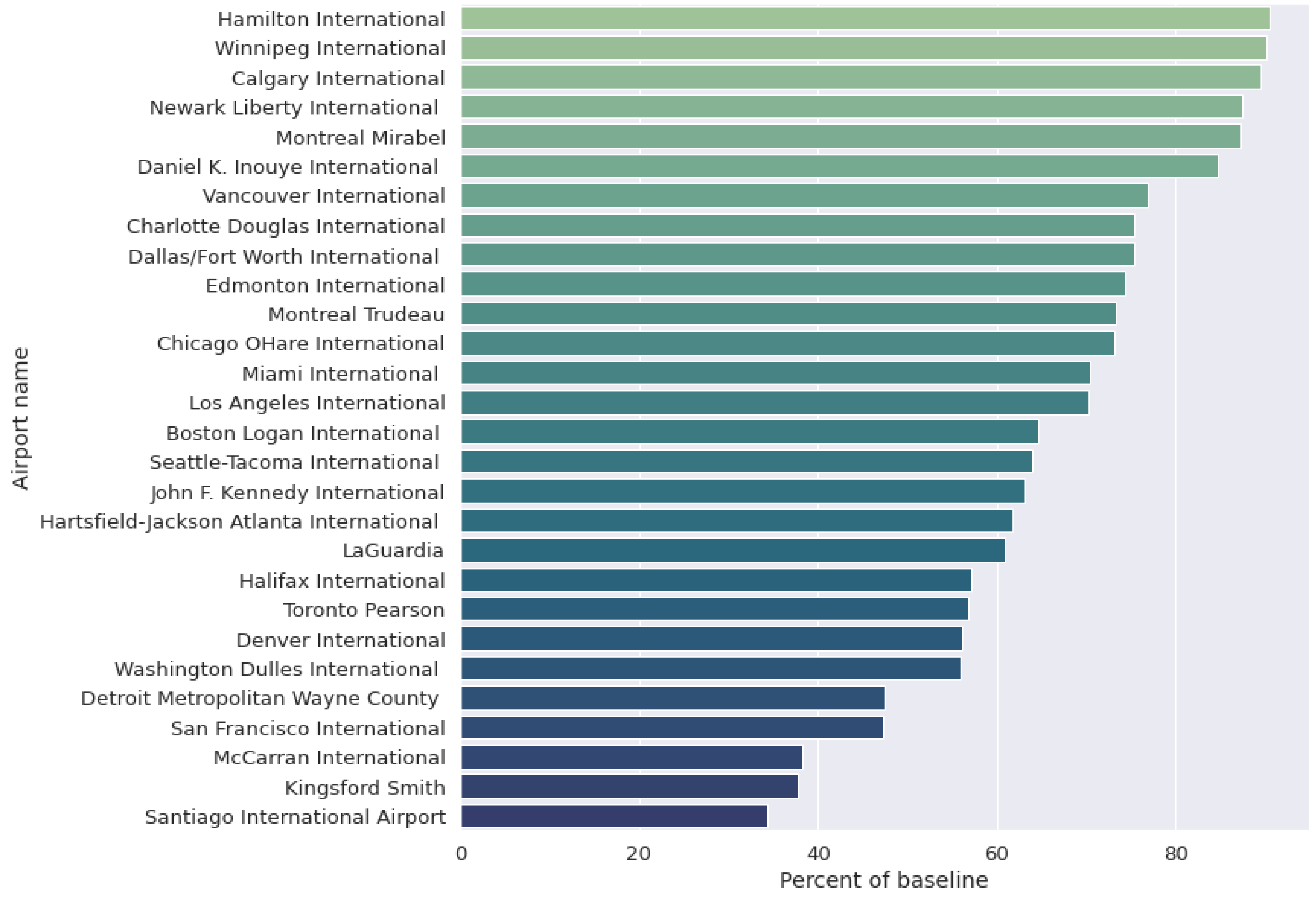
```
df1 = df.groupby("State")['PercentOfBaseline'].mean().sort_values(ascending = False).reset_index()
sns.set(font_scale = 1.2)
plt.figure(figsize = [10,10])
sns.barplot(data = df1, x = 'PercentOfBaseline', y = 'State',palette = 'Greens')
plt.xlabel("Percent of baseline")
```

```
Text(0.5, 0, 'Percent of baseline')
```



```
df1 = df.groupby("AirportName")['PercentOfBaseline'].mean().sort_values(ascending = False).reset_index()
sns.set(font_scale =1.2)
plt.figure(figsize = [10,10])
sns.barplot(data = df1, x = 'PercentOfBaseline', y = 'AirportName',palette = 'crest')
plt.xlabel('Percent of baseline')
plt.ylabel("Airport name")
```

```
Text(0, 0.5, 'Airport name')
```



```
df["lon"] = df.Centroid.apply(lambda x: x.split(" ")[0].replace("POINT("," "))
df["lat"] = df.Centroid.apply(lambda x: x.split(" ")[1].replace(")"," "))
```

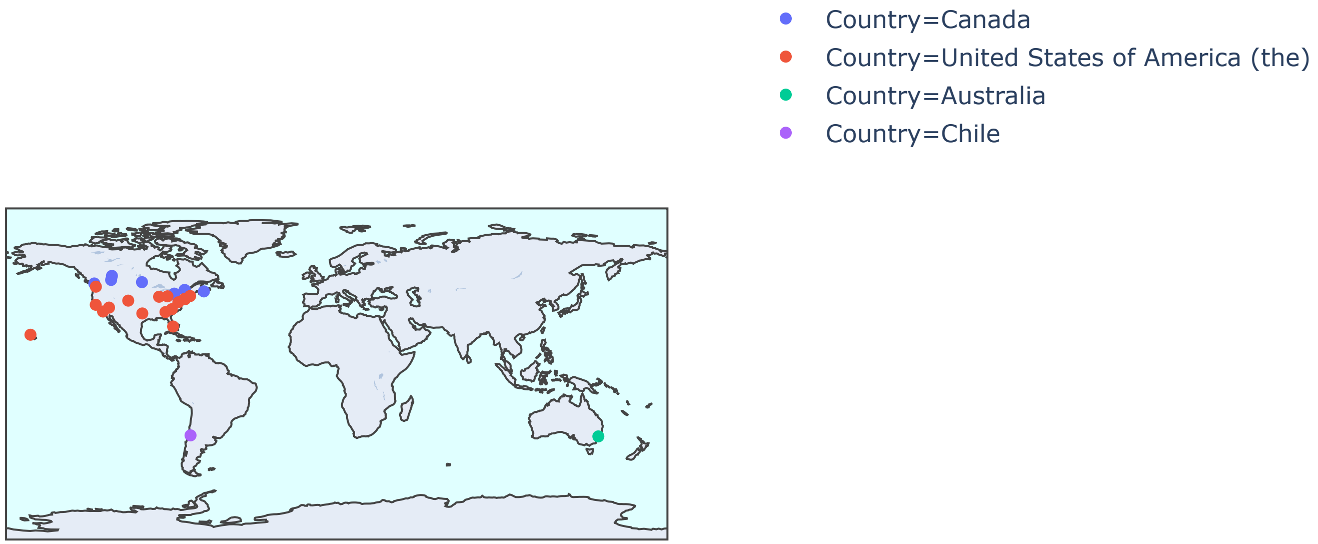
```
# Map
```

```
df1 = df.groupby(["Country","City",'lat','lon'])['PercentOfBaseline'].mean().sort_values(ascending = False).reset_index()
fig = px.scatter_geo(df1,
                     lat='lat',
                     lon='lon',
                     hover_name="Country",
                     color = 'Country',
                     hover_data = ['PercentOfBaseline',"City"],
                     labels = {"PercentOfBaseline":"Percent of Baseline"}

                    )

fig.update_geos(showocean = True,
                oceancolor = 'LightCyan',
                lakecolor = 'LightSteelBlue',
                showlakes = True,

               )
fig.show()
```



✓ 0s completed at 12:29 PM