VacationPy

Note

- Instructions have been included for each segment. You do not have to follow them exactly, but they are included to help you think through the steps.

In [1]:
```
# Dependencies and Setup
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import requests
import gmaps
import os
import json

# Import API key
from api_keys import g_key
```

Store Part I results into DataFrame

- Load the csv exported in Part I to a DataFrame

In [2]:
```
# Load the CSV file
cities_df = pd.read_csv("../output_data/cities.csv").drop(columns = "City_ID")

cities_df.head()
```

Out[2]:

	City	Cloudiness	Country	Date	Humidity	Lat	Lng	Max Temp	Wind Speed
0	hermanus	17	ZA	1595126828	90	-34.42	19.23	53.01	1.99
1	yar-sale	100	RU	1595126588	91	66.83	70.83	49.01	12.64
2	evansville	1	US	1595126613	70	37.97	-87.56	84.20	6.93
3	fort saint james	74	CA	1595127017	81	54.43	-124.25	60.55	4.72
4	ushuaia	75	AR	1595126619	94	-54.80	-68.30	32.00	9.17

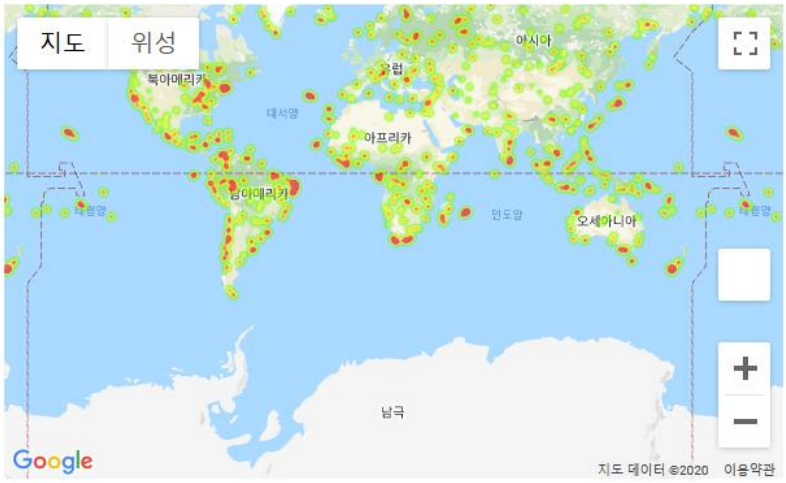
Humidity Heatmap

- Configure gmaps.
- Use the Lat and Lng as locations and Humidity as the weight.
- Add Heatmap layer to map.

In [3]:
```
# Configure gmaps
# width: 600px, height: 400px, zoom_level: 1, center is the location of the first city in the dataframe
layout = {"width" : "600px", "height": "400px"}
center = (cities_df.iloc[0, 5], cities_df.iloc[0, 6])
zoom_level = 1
gmaps.configure(api_key = g_key)

# Set locations and weights
locations = cities_df.iloc[:, [5, 6]]
weights = cities_df.iloc[:, 4]
```

In [4]:
```
# Make a map with a heatmap layer
fig = gmaps.figure(layout = layout, center = center, zoom_level = zoom_level)
heatmap = gmaps.heatmap_layer(locations, point_radius = 5, weights = weights, max_intensity = float(cities_df["Humidity"].max()))
fig.add_layer(heatmap)
fig
```



Create new DataFrame fitting weather criteria

- Narrow down the cities to fit weather conditions.
- Drop any rows will null values.

In [5]:
```
# Get cities that have ideal weather condition: 70 < Max temperature < 80°F, humidity < 40%, wind speed < 10 mph
ideal_cities_df = cities_df.loc[(cities_df.iloc[:, 7] < 80) &
                                (cities_df.iloc[:, 7] > 70) &
                                (cities_df.iloc[:, 4] < 40) &
                                (cities_df.iloc[:, 8] < 10)]

ideal_cities_df = ideal_cities_df.dropna()
ideal_cities_df
```

Out [5]:

	City	Cloudiness	Country	Date	Humidity	Lat	Lng	Max Temp	Wind Speed
12	zhangjiakou	0	CN	1595127018	29	40.81	114.88	76.71	8.55
281	airai	0	TL	1595127037	33	-8.93	125.41	74.26	3.51
287	dongsheng	0	CN	1595127037	26	39.82	109.98	74.01	5.28
320	mount isa	0	AU	1595126459	11	-20.73	139.50	78.80	6.93
324	yongan	0	CN	1595127041	34	39.70	113.69	72.48	8.79
325	komsomolskiy	0	UZ	1595127041	39	40.43	71.72	78.80	4.70
341	yeppoon	0	AU	1595127042	33	-23.13	150.73	75.20	4.70
379	rio verde de mato grosso	0	BR	1595127044	39	-18.92	-54.84	70.65	5.68
425	quchan	0	IR	1595127047	22	37.11	58.51	76.69	5.64

Hotel Map

- Store into variable named `hotel_df`.
- Add a "Hotel Name" column to the DataFrame.
- Set parameters to search for hotels with 5000 meters.
- Hit the Google Places API for each city's coordinates.
- Store the first Hotel result into the DataFrame.
- Plot markers on top of the heatmap.

In [6]:
```
# Copy necessary data from ideal_cities_df to hotel_df: City, Country
hotel_df = ideal_cities_df.iloc[:, [0, 2]].copy()

# Set keyword, radius, parameters, and url
keyword = "hotel"
radius = 5000
params = {"keyword" : keyword,
          "radius" : radius,
          "key" : g_key}

base_url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json"
```

In [7]:
```
# Iterate through the index of ideal_cities_df to get the locations of the ideal cities
for idx in ideal_cities_df.index:
    params["location"] = f"{ideal_cities_df.loc[idx, 'Lat']}, {ideal_cities_df.loc[idx, 'Lng']}"

    response = requests.get(base_url, params).json()

    # Store the hotel names and the locations of the hotels
    try:
        hotel_df.loc[idx, "Hotel Name"] = response["results"][0]["name"]
        hotel_df.loc[idx, "Lat"] = response["results"][0]["geometry"]["location"]["lat"]
        hotel_df.loc[idx, "Lng"] = response["results"][0]["geometry"]["location"]["lng"]

    except IndexError:
        pass
```

In [8]:
```
# Drop all rows that have null values
hotel_df = hotel_df.dropna()
hotel_df
```

Out[8]:

	City	Country	Hotel Name	Lat	Lng
12	zhangjiakou	CN	Pai Hotel Zhangjiakou North Mingde Road	40.836899	114.888733
287	dongsheng	CN	Crowne Plaza Ordos	39.828710	109.961820
320	mount isa	AU	ibis Styles Mt Isa Verona	-20.726140	139.492803
324	yongan	CN	Hengjili Hotel	39.694180	113.699635
325	komsomolskiy	UZ	Hotel Mehmon Saroy	40.384188	71.781349
341	yeppoon	AU	Yeppoon Beach House	-23.118958	150.748649
379	rio verde de mato grosso	BR	Hotel Serra Verde	-18.927593	-54.835754
425	quchan	IR	Khayam Hostel	37.106465	58.507865

In [9]:
```
# NOTE: Do not change any of the code in this cell

# Using the template add the hotel marks to the heatmap
info_box_template = """
<dl>
<dt>Name</dt><dd>{Hotel Name}</dd>
<dt>City</dt><dd>{City}</dd>
<dt>Country</dt><dd>{Country}</dd>
</dl>
"""
# Store the DataFrame Row
# NOTE: be sure to update with your DataFrame name
hotel_info = [info_box_template.format(**row) for index, row in hotel_df.iterrows()]
hotel_locations = hotel_df[["Lat", "Lng"]]
```

In [10]:
```
# Add marker layer on top of heat map
marker = gmaps.marker_layer(hotel_locations, info_box_content = hotel_info)
fig.add_layer(marker)

# Display figure
fig
```