6.2.7 Create a DataFrame of City Weather Data

You have the data in a list of dictionaries, which is a format that you can use to create a Pandas DataFrame. You will also need to export the DataFrame as a CSV file for Jack.

Our next steps will entail converting the array of dictionaries to a DataFrame, ensuring the columns are in the correct order, and exporting the DataFrame to a comma-separated (CSV) file.

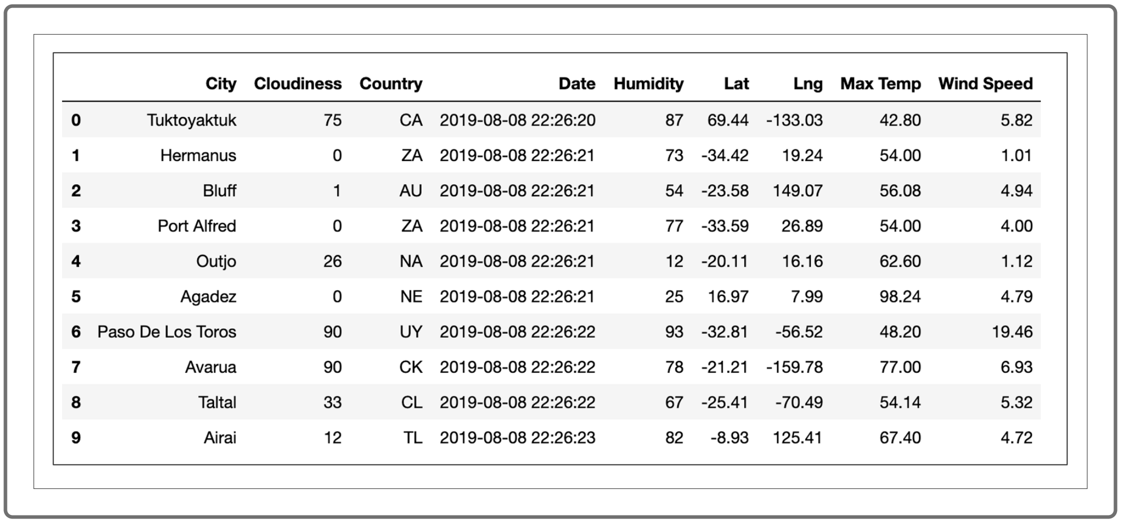
REWIND

Recall that we can convert a list of dictionaries to a Pandas DataFrame using `df = pd.DataFrame(list with dictionaries)`.

In a new cell, add the following code to convert the array of dictionaries to a Pandas DataFrame and run the cell.

```
# Convert the array of dictionaries to a Pandas DataFrame.
city_data_df = pd.DataFrame(city_data)
city_data_df.head(10)
```

The DataFrame should appear like the following.



	City	Cloudiness	Country	Date	Humidity	Lat	Lng	Max Temp	Wind Speed
0	Tuktoyaktuk	75	CA	2019-08-08 22:26:20	87	69.44	-133.03	42.80	5.82
1	Hermanus	0	ZA	2019-08-08 22:26:21	73	-34.42	19.24	54.00	1.01
2	Bluff	1	AU	2019-08-08 22:26:21	54	-23.58	149.07	56.08	4.94
3	Port Alfred	0	ZA	2019-08-08 22:26:21	77	-33.59	26.89	54.00	4.00
4	Outjo	26	NA	2019-08-08 22:26:21	12	-20.11	16.16	62.60	1.12
5	Agadez	0	NE	2019-08-08 22:26:21	25	16.97	7.99	98.24	4.79
6	Paso De Los Toros	90	UY	2019-08-08 22:26:22	93	-32.81	-56.52	48.20	19.46
7	Avarua	90	CK	2019-08-08 22:26:22	78	-21.21	-159.78	77.00	6.93
8	Taltal	33	CL	2019-08-08 22:26:22	67	-25.41	-70.49	54.14	5.32
9	Airai	12	TL	2019-08-08 22:26:23	82	-8.93	125.41	67.40	4.72

Next, we'll reorder the columns as City, Country, Date, Lat, Lng, Max Temp, Humidity, Cloudiness, and Wind Speed, so they are easy to read.

REWIND

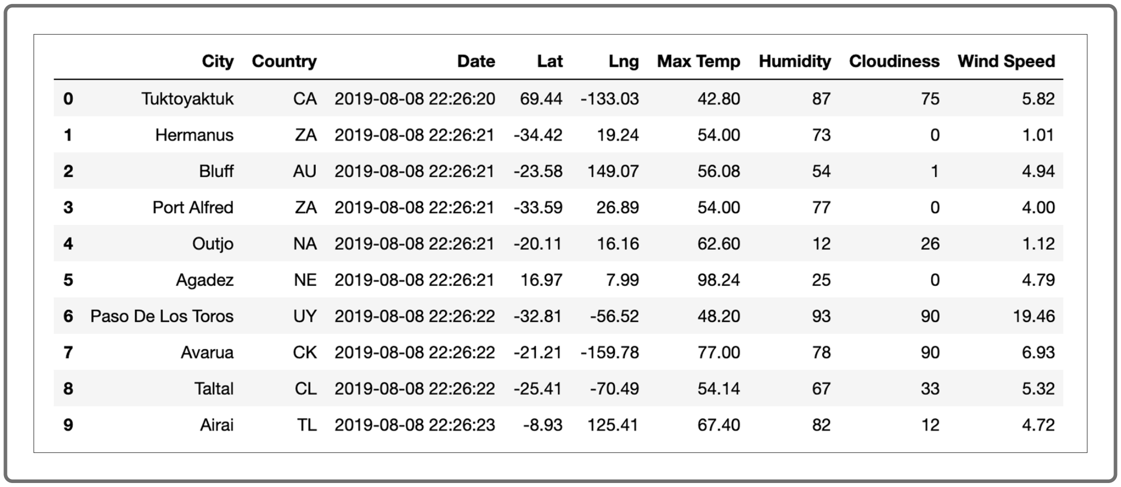
Recall that to reorder the columns, we assign a variable to an array of the columns in the order we want them to appear:

```
new_column_order = ["column2", "column4", "column1"]
```

Then, we assign a new or the same DataFrame with new column order:

```
df = df[new_column_order]
```

Once you reorder the columns and run the cell, the DataFrame should look like the following.



	City	Country	Date	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed
0	Tuktoyaktuk	CA	2019-08-08 22:26:20	69.44	-133.03	42.80	87	75	5.82
1	Hermanus	ZA	2019-08-08 22:26:21	-34.42	19.24	54.00	73	0	1.01
2	Bluff	AU	2019-08-08 22:26:21	-23.58	149.07	56.08	54	1	4.94
3	Port Alfred	ZA	2019-08-08 22:26:21	-33.59	26.89	54.00	77	0	4.00
4	Outjo	NA	2019-08-08 22:26:21	-20.11	16.16	62.60	12	26	1.12
5	Agadez	NE	2019-08-08 22:26:21	16.97	7.99	98.24	25	0	4.79
6	Paso De Los Toros	UY	2019-08-08 22:26:22	-32.81	-56.52	48.20	93	90	19.46
7	Avarua	CK	2019-08-08 22:26:22	-21.21	-159.78	77.00	78	90	6.93
8	Taltal	CL	2019-08-08 22:26:22	-25.41	-70.49	54.14	67	33	5.32
9	Airai	TL	2019-08-08 22:26:23	-8.93	125.41	67.40	82	12	4.72

Lastly, following the instructions below, we'll create an output file to save the DataFrame as a CSV in a new folder for that file.

In our World_Weather_Analysis folder, create a new folder called "weather_data." Add the following code to a new cell, run the cell, then confirm your CSV file is in the folder.

```
# Create the output file (CSV).
output_data_file = "weather_data/cities.csv"
# Export the City_Data into a CSV.
city_data_df.to_csv(output_data_file, index_label="City_ID")
```

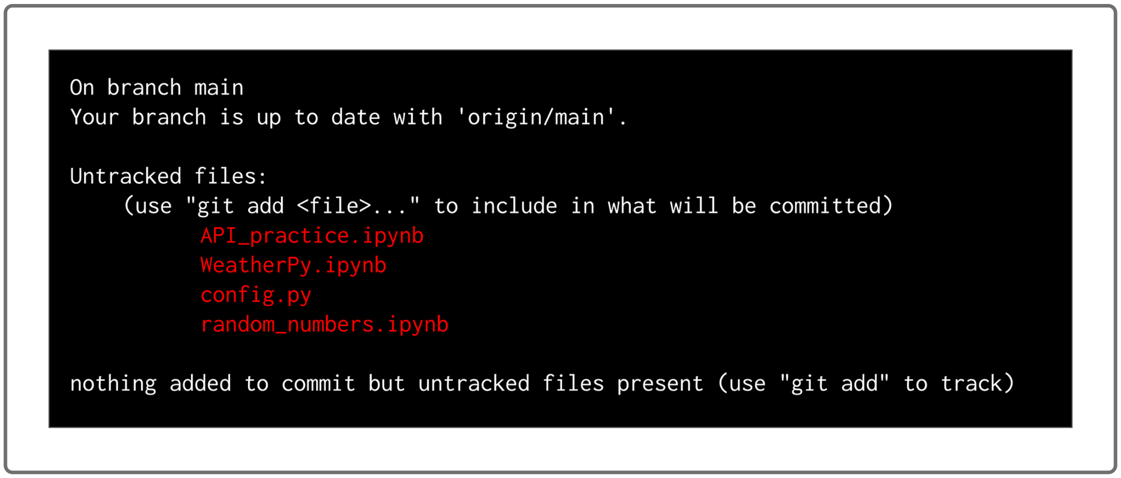
The last line in the code block will export the DataFrame to a CSV file, with the index label (or column A) header as "City_ID." If we ever need to export the CSV file to a DataFrame, that header will be present in the DataFrame.

We've completed our tasks for making API calls, parsing the response, and collecting the data for our project. Before we move on to graphing and statistical analysis, let's update our GitHub repository.

Modify the .gitignore File

We don't want the `config.py` file containing the API key to be exposed to the public on GitHub, as this would mean anyone could copy and use our API key, possibly causing us to incur charges.

When we type `git status` in the command line, we can see all the files we have created so far that are untracked.



```
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
    (use "git add <file>..." to include in what will be committed)
          API_practice.ipynb
          WeatherPy.ipynb
          config.py
          random_numbers.ipynb

nothing added to commit but untracked files present (use "git add" to track)
```

If we only wanted to add the `WeatherPy.ipynb` file to GitHub we could type

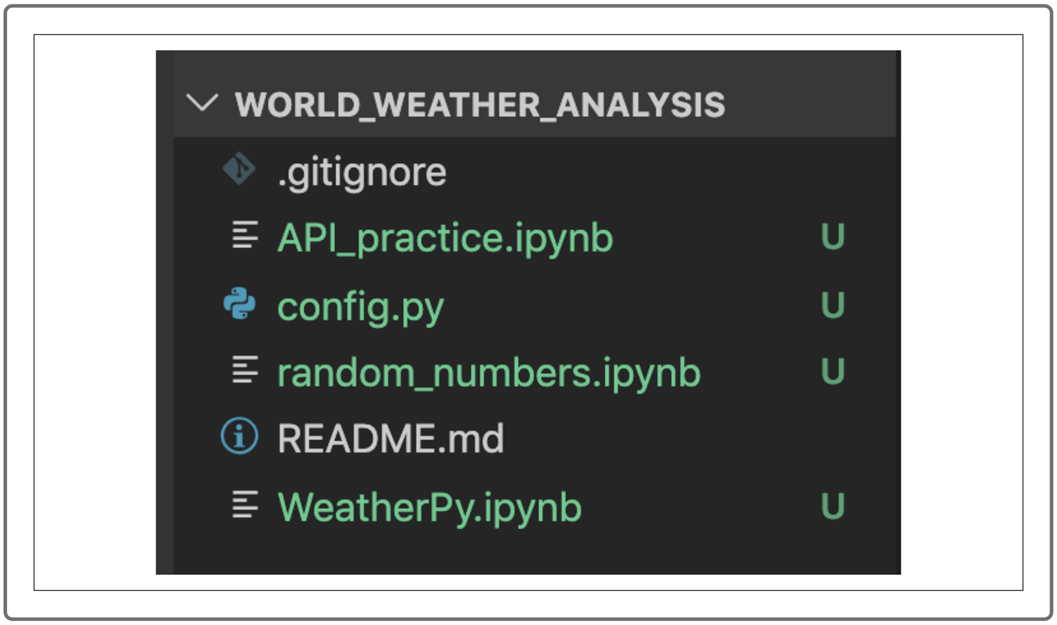
`git add WeatherPy.ipynb`. However, every time we want to add a new file or update current files to the repository, we have to add each file individually, which is time-consuming and cumbersome. Instead, we can add the files we don't want to track to the `.gitignore` file.

REWIND

GitHub does not track files and extensions that are added to the `.gitignore` file.

Before we add our files to GitHub, let's add `config.py` to the `.gitignore` file. Follow these steps:

1. Open your World_Weather_Analysis GitHub folder in VS Code.



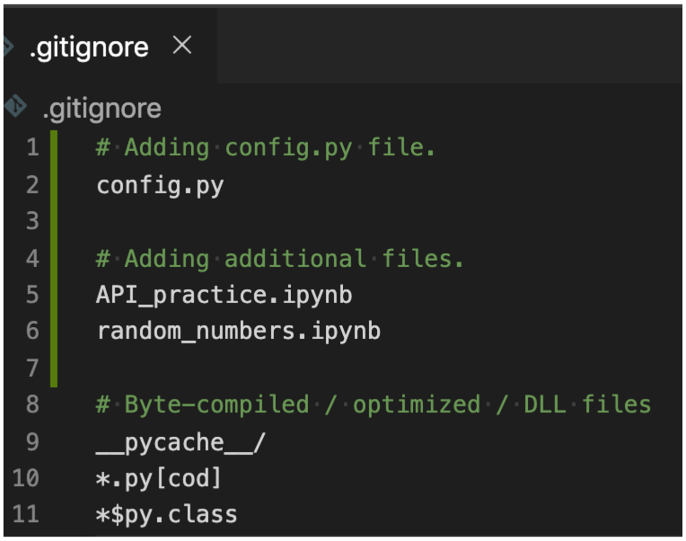
```
WORLD_WEATHER_ANALYSIS
  .gitignore
  API_practice.ipynb        U
  config.py                 U
  random_numbers.ipynb      U
  README.md
  WeatherPy.ipynb           U
```

2. Open the `.gitignore` file, and on the first line type the following:

```
# Adding config.py file.
config.py
```

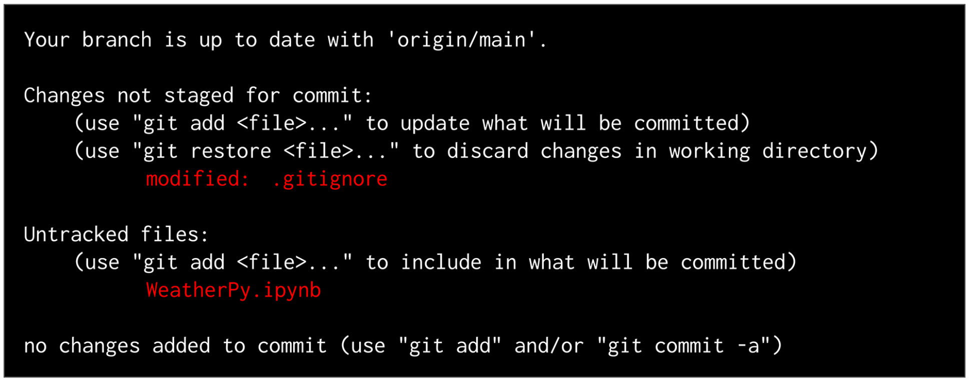
3. While the `.gitignore` file is open, add the `API_practice.ipynb` and `random_numbers.ipynb` files and save the file.

   Your `.gitignore` file should look similar to the following.



```
# Adding config.py file.
config.py

# Adding additional files.
API_practice.ipynb
random_numbers.ipynb

# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class
```

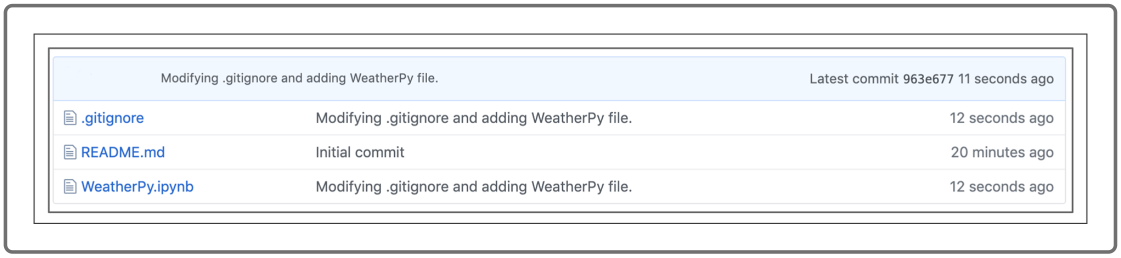
4. In the command line, type `git status` and press Enter. The output should say the `.gitignore` file has been modified and the `WeatherPy.ipynb` file is untracked.



```
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
    (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
          modified:  .gitignore

Untracked files:
    (use "git add <file>..." to include in what will be committed)
          WeatherPy.ipynb

no changes added to commit (use "git add" and/or "git commit -a")
```

5. Use `git add`, `git commit`, and `git push` to commit the modifications to `.gitignore` and the `WeatherPy.ipynb` file to GitHub.

On GitHub, the only new file you should see is the `WeatherPy.ipynb` file.



Modifying .gitignore and adding WeatherPy file. Latest commit 963e677 11 seconds ago

.gitignore	Modifying .gitignore and adding WeatherPy file.	12 seconds ago
README.md	Initial commit	20 minutes ago
WeatherPy.ipynb	Modifying .gitignore and adding WeatherPy file.	12 seconds ago

Congratulations on modifying your `.gitignore` file!