Getting the Data

Data Source:

https://github.com/CSSEGISandData/COVID-19 (https://github.com/CSSEGISandData/COVID-19)
https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series (https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series)

Naming conventions

`group`

`group` refers to the two separate "groups" of data.

- `"world"` - represents data from each country.
- `"usa"` - represents data from each state in the United States.

`kind`

`kind` will refer to the two different kinds of COVID-19 data.

- `"deaths"`
- `"cases"`

`area`

- `"area"` will refer to specific countries or states.

Downloading the data

In [1]:
```
import pandas as pd

DOWNLOAD_URL = (
    "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/"
    "master/csse_covid_19_data/csse_covid_19_time_series/"
    "time_series_covid19_{kind}_{group}.csv"
)

GROUPS = "world", "usa"
KINDS = "deaths", "cases"


# Function 1
def download_data(group, kind):
    """
    Fetches and returns COVID-19 data from the John Hopkins GitHub repository.
    Selects data type ('deaths' or 'cases') and scope ('world' or 'usa').

    Parameters
    ----------
    group : str
        'world' for global data or 'usa' for US data.
    kind : str
        'deaths' for death data or 'cases' for case data.

    Returns
    -------
    DataFrame
        Pandas DataFrame with the requested data.
    """
    group_change_dict = {"world": "global", "usa": "US"}
    kind_change_dict = {"deaths": "deaths", "cases": "confirmed"}
    group = group_change_dict[group]
    kind = kind_change_dict[kind]
    return pd.read_csv(DOWNLOAD_URL.format(kind=kind, group=group))
```

In [2]:
```
df_world_deaths = download_data('world', 'deaths')
df_world_deaths.head()
```

Out[2]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	..
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	..
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	..
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	..
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	..
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	..

5 rows × 1147 columns

In [3]:
```
GROUPS = "world", "usa"
KINDS = "deaths", "cases"

def read_all_data():
    """
    Downloads all data combinations (world/usa and deaths/cases) from the reposito

    Returns
    -------
    dict
        Dictionary of DataFrames, keyed by "{group}_{kind}".
    """
    data = {}
    for group in GROUPS:
        for kind in KINDS:
            df = download_data(group, kind)
            data[f"{group}_{kind}"] = df
    return data
```

In [4]:
```
data = read_all_data()
data['world_cases'].head(5)
```

Out[4]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	..
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	..
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	..
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	..
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	..
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	..

5 rows × 1147 columns

In [5]:
```
data['usa_cases'].head(5)
```

Out[5]:

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long_	...	2/
0	84001001	US	USA	840	1001.0	Autauga	Alabama	US	32.539527	-86.644082	...	
1	84001003	US	USA	840	1003.0	Baldwin	Alabama	US	30.727750	-87.722071	...	
2	84001005	US	USA	840	1005.0	Barbour	Alabama	US	31.868263	-85.387129	...	
3	84001007	US	USA	840	1007.0	Bibb	Alabama	US	32.996421	-87.125115	...	
4	84001009	US	USA	840	1009.0	Blount	Alabama	US	33.982109	-86.567906	...	

5 rows × 1154 columns

Save the data locally

In [6]:
```
def write_data(data, directory, **kwargs):
    """
    Saves each DataFrame in 'data' to CSV files in the specified directory.

    Parameters
    ----------
    data : dict
        Dictionary of DataFrames to save.
    directory : str
        Target directory for CSV files.

    Returns
    -------
    None
    """
    for name, df in data.items():
        df.to_csv(f"{directory}/{name}.csv", **kwargs)
```

In [7]:
```
#run write_data function
write_data(data, "data/raw", index=False)
```

In [8]:
```
def read_local_data(group, kind, directory):
    """
    Reads a specific CSV file as a DataFrame from a given directory.

    Parameters
    ----------
    group : str
        'world' or 'usa'.
    kind : str
        'deaths' or 'cases'.
    directory : str
        Directory path to read the file from.

    Returns
    -------
    DataFrame
    """
    return pd.read_csv(f"{directory}/{group}_{kind}.csv")
```

In [9]:
```
read_local_data('world', 'deaths', 'data/raw').head(3)
```

Out[9]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	...

3 rows × 1147 columns

In [10]:
```
def run():
    """
    Executes data loading and transformation steps for all data combinations.

    Returns
    -------
    dict
        Dictionary of transformed DataFrames.
    """
    data = {}
    for group in GROUPS:
        for kind in KINDS:
            df = read_local_data(group, kind, "data/raw")
            data[f"{group}_{kind}"] = df
    return data
```

In [11]:
```
# run run() function
data = run()
data['usa_deaths'].tail(3)
```

Out[11]:

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long
3339	84090056	US	USA	840	90056.0	Unassigned	Wyoming	US	0.000000	0.00000
3340	84056043	US	USA	840	56043.0	Washakie	Wyoming	US	43.904516	-107.68018
3341	84056045	US	USA	840	56045.0	Weston	Wyoming	US	43.839612	-104.56748

3 rows × 1155 columns