I. Motivation

在這個疫情肆虐的時代，除了防疫工作之外，數據分析也占了很重要的一部份，防疫人員從圖表中能夠清楚得知疫情的分布與人民染疫狀況，因此我想在本次作業中，嘗試使用多項資料集匯聚成圖表進行分析。

II. Application Description

在程式中，我使用的語言是 python，IDE 是 Google Colab。在程式中，總共分為四大部分，分別是 CSSE, Global Seven Days Moving Average, Taiwan Map, World Map，個別利用不同 dataset 進行分析並以圖表方式呈現；前兩個部分利用 pymysql 這個 package 連上 AWS 上的 MySQL，以 query 方式將資料匯入並整理，接著再對 table 進行 query 取出目標資料；而因為 python 的用途廣泛，藉由 pandas, geopandas 這兩個 package 也能在不使用 sql 的情況下對資料進行分析與整理，因此後兩個部分嘗試使用這兩個 package，使用 python 語法將資料匯入、整理與取出目標資料。藉由這兩種相異的資料處理方法，能夠看到他們的語法與運作方式差異。

III. Data sources and how you collect and import the data (manually or automatically)

A. CSSE

在這部分中，我使用的是 Our World in Data Github 上的 owid-covid-data.csv，裡面匯集了多項其他的 covid19 資料，十分完整，而在程式中則是藉由 wget package 與 URL 自動下載 csv 檔案。

B. Global Seven Days Moving Average

在這部分中，我使用的是 COVID-19 全球疫情地圖中的 COVID-19 七日移動平均新增確診數資料下載 API，在程式中一樣藉由 wget package 與 URL 自動下載 csv 檔案。

C. Taiwan Map

在這部分中，我使用的是 COVID-19 全球疫情地圖中的時間軸-縣市鄉鎮疫情表單，不同的是，在程式中藉由 pandas package 與 URL 自動下載 csv 檔案。不過由於這部分中要畫出地圖，因此還需要各市區的界線，而我使用的是台灣鄉鎮市區界線中的.shp 檔，需要使用者手動下載檔案後，放入本 Google Colab .ipynb 檔案的相同 Google Drive 資料夾中。

D. World Map

在這部分中，我使用的是 COVID-19 全球疫情地圖中的 COVID-19 七日移動平均新增確診數資料下載 API，在程式中一樣藉由 pandas package 與 URL 自動下載 csv 檔案。不過由於這部分中同樣要畫出地圖，因此還需要全球各國的界線，而我使用的是 wri-bounds Github 中"All" countries U.S.版本的.shp 檔，需要使用者手動下載檔案後，放入本 Google Colab .ipynb 檔案的相同 Google Drive 資料夾中。

IV. Database schema

A. CSSE

`owid-covid-data`的 schema 可以從 create table query 中得知，query 如下：

```
CREATE TABLE IF NOT EXISTS `owid-covid-data`(
    `iso_code` TEXT NOT NULL,
    `continent` TEXT NOT NULL,
    `location` TEXT NOT NULL,
    `date` Date NOT NULL,
    `total_cases` integer,
    `new_cases` integer,
    `new_cases_smoothed` double precision,
    `total_deaths` integer,
    `new_deaths` integer,
    `new_deaths_smoothed` double precision,
    `total_cases_per_million` double precision,
    `new_cases_per_million` double precision,
    `new_cases_smoothed_per_million` double precision,
    `total_deaths_per_million` double precision,
    `new_deaths_per_million` double precision,
    `new_deaths_smoothed_per_million` double precision,
    `reproduction_rate` double precision,
    `icu_patients` double precision,
    `icu_patients_per_million` double precision,
    `hosp_patients` double precision,
    `hosp_patients_per_million` double precision,
    `weekly_icu_admissions` double precision,
    `weekly_icu_admissions_per_million` double precision,
    `weekly_hosp_admissions` double precision,
    `weekly_hosp_admissions_per_million` double precision,
    `total_tests` double precision,
    `new_tests` double precision,
    `total_tests_per_thousand` double precision,
    `new_tests_per_thousand` double precision,
    `new_tests_smoothed` double precision,
    `new_tests_smoothed_per_thousand` double precision,
    `positive_rate` double precision,
```

```
    `tests_per_case` double precision,
    `tests_units` double precision,
    `total_vaccinations` double precision,
    `people_vaccinated` double precision,
    `people_fully_vaccinated` double precision,
    `total_boosters` double precision,
    `new_vaccinations` double precision,
    `new_vaccinations_smoothed` double precision,
    `total_vaccinations_per_hundred` double precision,
    `people_vaccinated_per_hundred` double precision,
    `people_fully_vaccinated_per_hundred` double precision,
    `total_boosters_per_hundred` double precision,
    `new_vaccinations_smoothed_per_million` double precision,
    `new_people_vaccinated_smoothed` double precision,
    `new_people_vaccinated_smoothed_per_hundred` double precision,
    `stringency_index` double precision,
    `population` double precision,
    `population_density` double precision,
    `median_age` double precision,
    `aged_65_older` double precision,
    `aged_70_older` double precision,
    `gdp_per_capita` double precision,
    `extreme_poverty` double precision,
    `cardiovasc_death_rate` double precision,
    `diabetes_prevalence` double precision,
    `female_smokers` double precision,
    `male_smokers` double precision,
    `handwashing_facilities` double precision,
    `hospital_beds_per_thousand` double precision,
    `life_expectancy` double precision,
    `human_development_index` double precision,
    `excess_mortality_cumulative_absolute` double precision,
    `excess_mortality_cumulative` double precision,
    `excess_mortality` double precision,
    `excess_mortality_cumulative_per_million` double precision
);
```

B. Global Seven Days Moving Average

`world-covid19`的 schema 可以從 create table query 中得知，query 如下：

```
CREATE TABLE IF NOT EXISTS `world_covid19`(
    `id` integer NOT NULL,
    `iso_code` TEXT NOT NULL,
    `洲名` TEXT,
    `國家` TEXT,
    `日期` Date NOT NULL,
    `總確診數` integer NOT NULL,
    `新增確診數` integer,
    `七天移動平均新增確診數` double precision,
    `總確診數/每百萬人` double precision,
    `新增確診數/每百萬人` double precision,
    `七天移動平均新增確診數/每百萬人` double precision,
    `總人口數` integer,
    `新聞稿發佈新增確診數` TEXT
  );
```

C. Taiwan Map

Taiwan_data 由於是用 pandas 讀取 csv 檔，因此 schema 可由 print(taiwan_data.info(verbose = True, null_counts = False))得出，結果如下：

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30614 entries, 0 to 30613
Data columns (total 7 columns):
 #   Column  Dtype
---  ------  -----
 0   id      int64
 1   個案研判日   object
 2   個案公佈日   object
 3   縣市別     object
 4   區域      object
 5   新增確診人數  int64
 6   累計確診人數  int64
dtypes: int64(3), object(4)
```

D. World Map

world_data 由於是用 pandas 讀取 csv 檔，因此 schema 可由 print(world_data.info(verbose = True, null_counts = False))得出，結果如下：

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 226 entries, 0 to 225
Data columns (total 13 columns):
 #   Column                  Dtype
---  ------                  -----
 0   id                      int64
 1   iso_code                object
 2   洲名                      object
 3   國家                      object
 4   日期                      object
 5   總確診數                    int64
 6   新增確診數                   int64
 7   七天移動平均新增確診數             float64
 8   總確診數/每百萬人              float64
 9   新增確診數/每百萬人             float64
 10  七天移動平均新增確診數/每百萬人  float64
 11  總人口數                    int64
 12  新聞稿發佈新增確診數             object
dtypes: float64(4), int64(4), object(5)
```

而 pandas dtype 與 python data type 的對應圖如下：

Pandas dtype	Python type	NumPy type	Usage
object	str or mixed	string_, unicode_, mixed types	Text or mixed numeric and non-numeric values
int64	int	int_, int8, int16, int32, int64, uint8, uint16, uint32, uint64	Integer numbers
float64	float	float_, float16, float32, float64	Floating point numbers
bool	bool	bool_	True/False values
datetime64	NA	datetime64[ns]	Date and time values
timedelta[ns]	NA	NA	Differences between two datetimes
category	NA	NA	Finite list of text values

V. The application's functions and the related SQL queries used for the function.

A. CSSE

在這部分的程式中，首先讀取 owid-covid-data.csv 並藉由 query 匯入到 owid-covid-data table 中，程式碼如下：

```
def load_and_create_CSSE(cursor, db):
  URL = "https://covid.ourworldindata.org/data/owid-covid-data.csv"
  response = wget.download(URL, "owid-covid-data.csv")

  sql = '''DROP TABLE IF EXISTS `owid-covid-data`'''
  execute_sql(cursor, db, sql)

  sql = '''
  CREATE TABLE IF NOT EXISTS `owid-covid-data`(
    `iso_code` TEXT NOT NULL,
    `continent` TEXT NOT NULL,
    `location` TEXT NOT NULL,
    ... #省略大部分attributes
    ...
    ...
    `excess_mortality_cumulative` double precision,
    `excess_mortality` double precision,
    `excess_mortality_cumulative_per_million` double precision
  );
  '''
  execute_sql(cursor, db, sql)

  sql = '''
  LOAD DATA LOCAL INFILE './owid-covid-data.csv'
  INTO TABLE `owid-covid-data`
  FIELDS TERMINATED BY ','
  IGNORE 1 ROWS;
  '''
  execute_sql(cursor, db, sql)
```

接著對 table 進行 normalize，程式碼如下：

```
def normalize_CSSE(cursor, db):
  sql = '''DROP TABLE IF EXISTS `iso_to_country`'''
  execute_sql(cursor, db, sql)

  sql = '''
  CREATE TABLE IF NOT EXISTS `iso_to_country`(
    `iso_code` TEXT NOT NULL,
    `continent` TEXT NOT NULL,
    `location` TEXT NOT NULL
  );
  '''
  execute_sql(cursor, db, sql)
  sql = '''
  INSERT INTO `iso_to_country` (`iso_code`, `continent`, `location`)
  SELECT distinct(`iso_code`), `continent`, `location` FROM `owid-covid-data`
  '''
  execute_sql(cursor, db, sql)
  sql = '''
  ALTER TABLE `owid-covid-data`
  DROP COLUMN `continent`,
  DROP COLUMN `location`;
  '''
  execute_sql(cursor, db, sql)
```

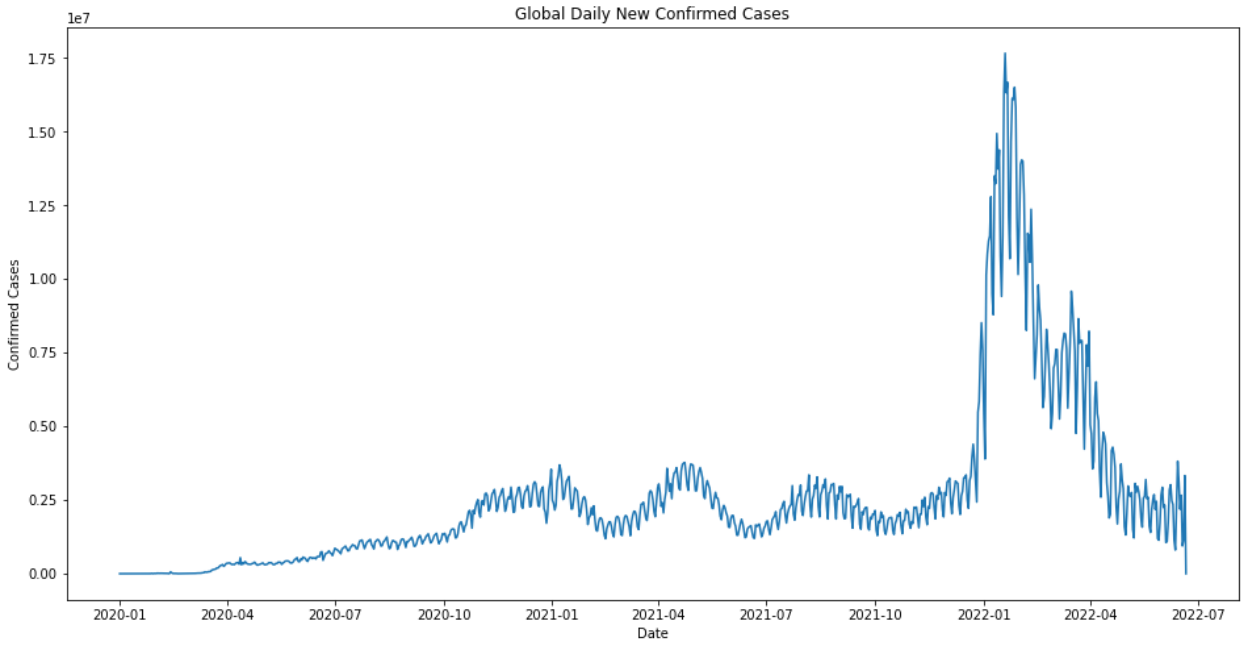
在將 table 匯入與處理後，再藉由 query 取出資料，畫出四張折線圖，分別如下：

1. Global Daily New Confirmed Cases

   在這張折線圖中，X 軸顯示日期，Y 軸顯示全球每日新的 covid19 確診數量(單位為 1e7)，query 如下：

```
sql = '''
SELECT date, sum(new_cases) FROM `owid-covid-data`
GROUP BY date
ORDER BY date
'''
```

   折線圖如下：

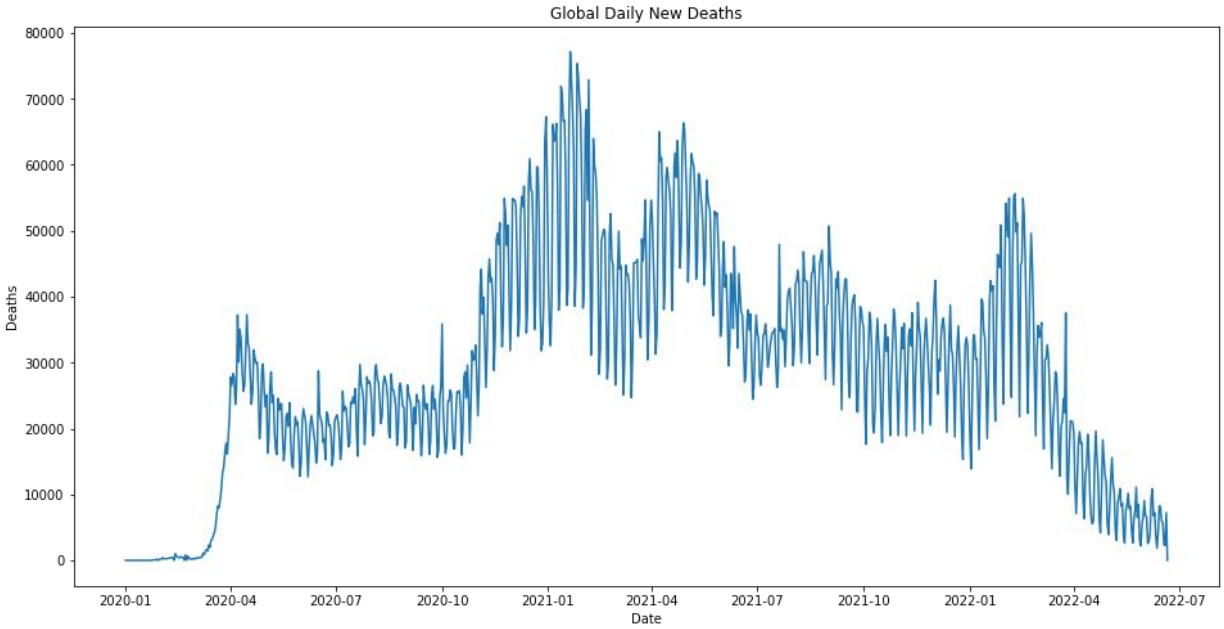


2. Global Daily New Deaths

   在這張折線圖中，X 軸顯示日期，Y 軸顯示全球每日新的 covid19 死亡數量，query 如下：

```
sql = '''
SELECT date, sum(new_deaths) FROM `owid-covid-data`
GROUP BY date
ORDER BY date
'''
```

折線圖如下：

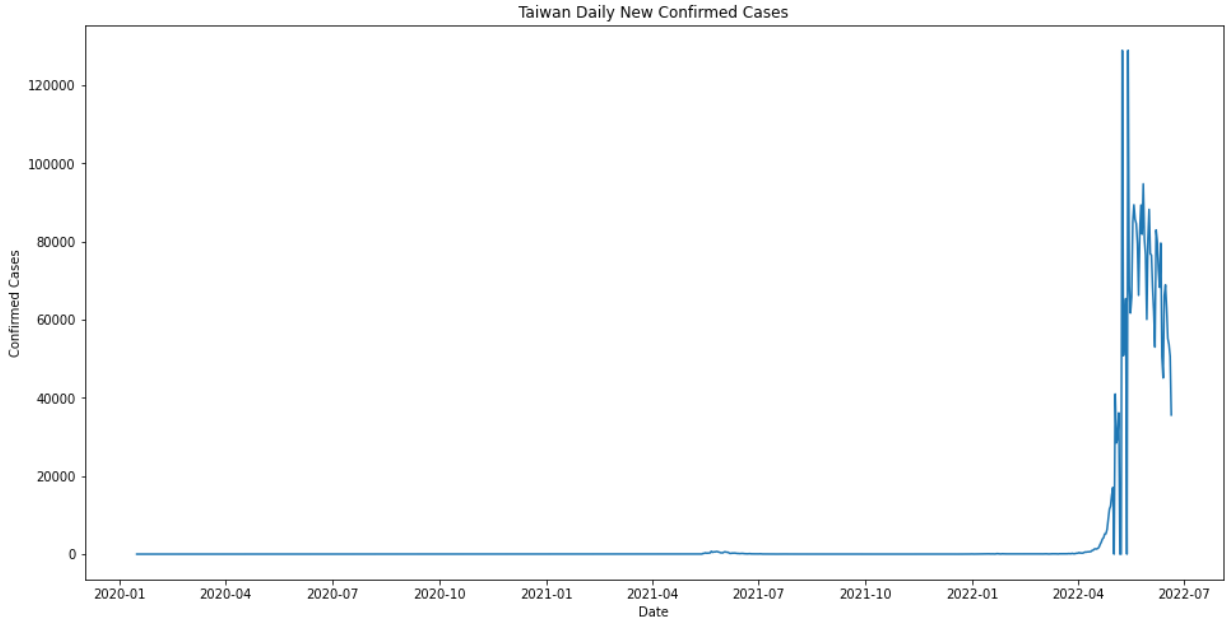


3. Taiwan Daily New Confirmed Cases

在這張折線圖中，X 軸顯示日期，Y 軸顯示台灣每日新的 covid19 確診數量，query 如下：

```
sql = '''
SELECT date, sum(new_cases) FROM `owid-covid-data`
WHERE iso_code='TWN'
GROUP BY date
ORDER BY date
'''
```

折線圖如下：

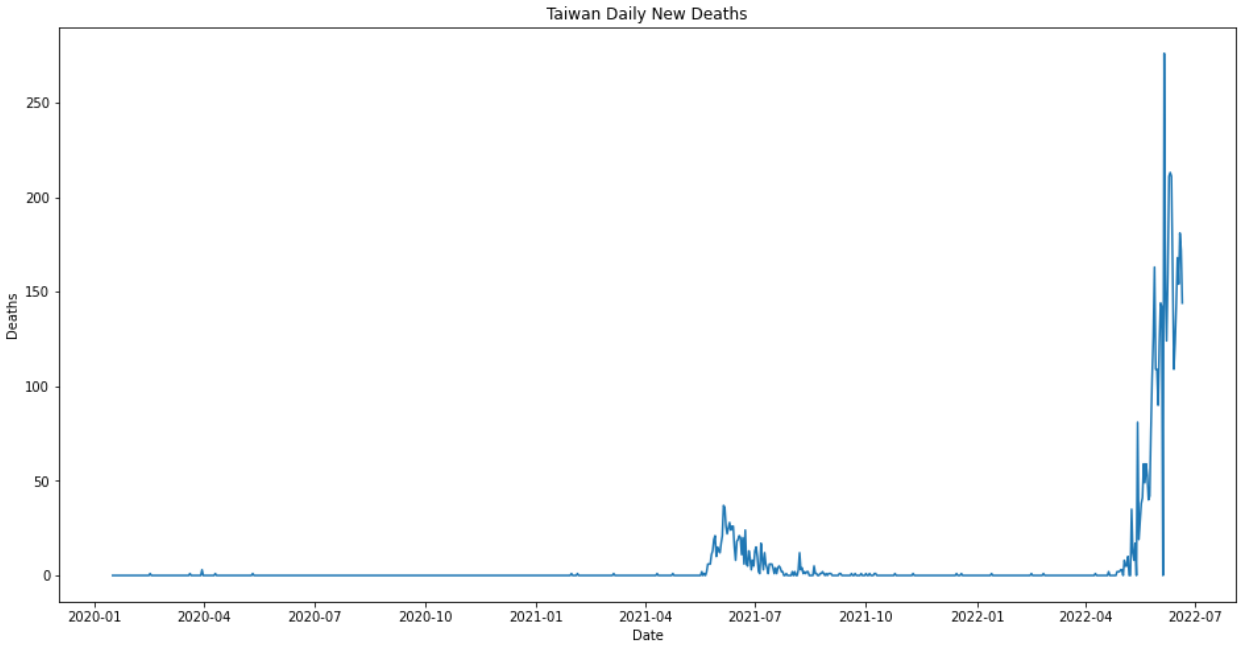


4. Taiwan Daily New Deaths

在這張折線圖中，X 軸顯示日期，Y 軸顯示台灣每日新的 covid19 死亡數量，query 如下：

```
sql = '''
SELECT date, sum(new_deaths) FROM `owid-covid-data`
WHERE iso_code='TWN'
GROUP BY date
ORDER BY date
'''
```

折線圖如下：



B. Global Seven Days Moving Average

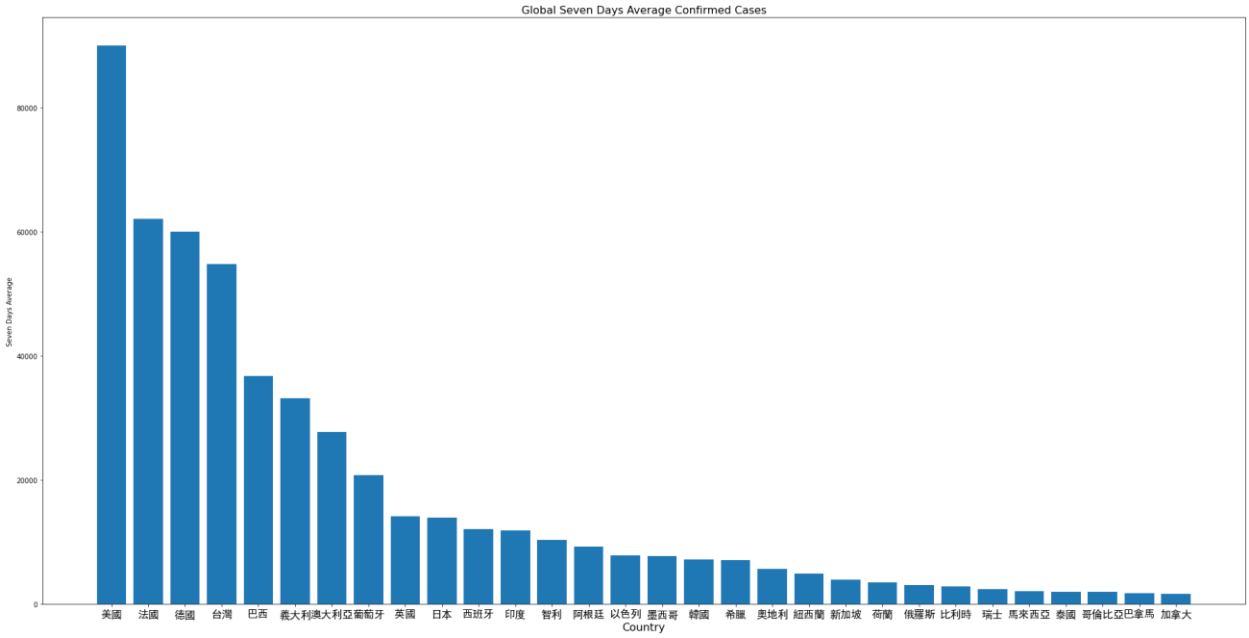
在這部分的程式中，首先讀取 owl_world_v2_2022-06-21.csv 並藉由 query 匯入到 world_covid19 table 中，程式碼如下：

```
def load_and_create_world_covid19():
  URL = "https://covid-19.nchc.org.tw/api/download/owl_world_v2_2022-06-21.csv"
  response = wget.download(URL, "world_covid19.csv")

  sql = '''DROP TABLE IF EXISTS `world_covid19`'''
  execute_sql(cursor, db, sql)

  sql = '''
  CREATE TABLE IF NOT EXISTS `world_covid19`(
    `id` integer NOT NULL,
    `iso_code` TEXT NOT NULL,
    `洲名` TEXT,
    `國家` TEXT,
    `日期` Date NOT NULL,
    `總確診數` integer NOT NULL,
    `新增確診數` integer,
    `七天移動平均新增確診數` double precision,
    `總確診數/每百萬人` double precision,
    `新增確診數/每百萬人` double precision,
    `七天移動平均新增確診數/每百萬人` double precision,
    `總人口數` integer,
    `新聞稿發佈新增確診數` TEXT
  );
  '''
  execute_sql(cursor, db, sql)

  sql = '''SET NAMES big5'''
  execute_sql(cursor, db, sql)

  sql = '''
  LOAD DATA LOCAL INFILE './world_covid19.csv'
  INTO TABLE `world_covid19`
  CHARACTER SET big5
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 ROWS;
  '''
  execute_sql(cursor, db, sql)
```

在將 table 匯入後，再藉由 query 取出資料，畫出折線圖 Global Seven Days Average Confirmed Cases，X 軸顯示國家名稱，Y 軸顯示 covid19 七日移動平均新增確診數量，圖中會顯示該數量最高的 30 個國家，query 如下：

```
sql = '''
SELECT `國家`, `七天移動平均新增確診數` FROM `world_covid19`
ORDER BY `七天移動平均新增確診數` DESC
'''
```

長條圖如下：



C. Taiwan Map

在這部分的程式中，首先，與之前不同的是，利用 pandas package 讀取 covidtable_taiwan_cdc4_1_2022-06-21.csv 到 taiwan_data 中，接著對 taiwan_data preprocess 與去除掉不必要的 attributes，程式碼如下：

```
def load_and_preprocess_cdc_taiwan():
  URL = "https://covid-19.nchc.org.tw/api/download/covidtable_taiwan_cdc4_1_2022-06-21.csv"
  taiwan_data = pd.read_csv(URL, encoding = 'big5')
  print('Schema before preprocess taiwan_data')
  print(taiwan_data.info(verbose = True, null_counts = False))
  print('Examples before preprocess taiwan_data')
  print(taiwan_data.head())
  taiwan_data = taiwan_data[taiwan_data['區域']=='全區']
  taiwan_data=taiwan_data.drop(taiwan_data.columns[[0,2,4]], axis=1)
  print('Schema after preprocess taiwan_data')
  print(taiwan_data.info(verbose = True, null_counts = False))
  print('Examples After preprocess taiwan_data')
  print(taiwan_data.head())
  return taiwan_data
```

由於這部分中要畫出地圖，因此還需要台灣各市區的界線，從 Google Drive 中匯入.shp 檔後，利用 geopandas package 讀取到 taiwan_shp 中，接著對 taiwan_shp 去除掉不必要的 attributes，程式碼如下：

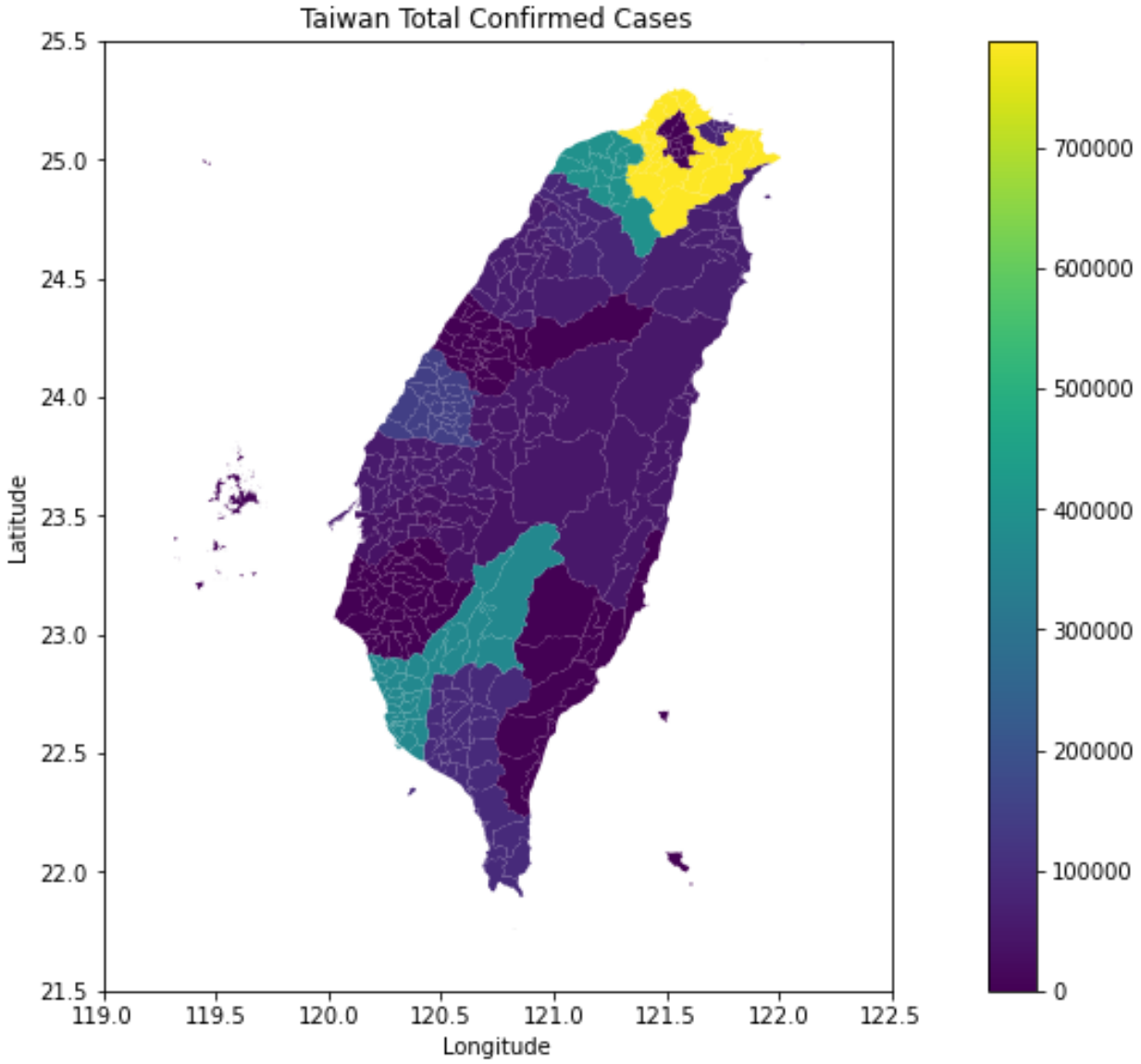
```python
def load_and_preprocess_taiwan_shp():
  taiwan_shp = gpd.read_file('./TOWN_MOI_1100415.shp', encoding='utf-8')
  print('Before preprocess taiwan_shp')
  print(taiwan_shp.head())
  taiwan_shp = taiwan_shp.drop(taiwan_shp.columns[[0,1,4,5,6]], axis=1)
  print('After preprocess taiwan_shp')
  print(taiwan_shp.head())
  return taiwan_shp
```

在處理完 taiwan_data 與 taiwan_shp 後，利用 pandas dataframe 的語法分析資料並畫出兩張分級著色圖，分別如下：

1. Taiwan Total Confirmed Cases
   在這張分級著色圖中，X 軸顯示經度，Y 軸顯示緯度，各顏色代表各縣市累計 covid19 確診數量，程式碼如下：

```python
def plot_taiwan_total_confirmed_cases(taiwan_data, taiwan_shp):
  taiwan_total_cases = taiwan_data.groupby('縣市別').sum()
  print('Taiwan total confirmed cases')
  print(taiwan_total_cases.head())
  left = taiwan_shp.set_index('COUNTYNAME')
  right = taiwan_total_cases
  result = left.join(right)
  result = result.fillna(0)
  print('Results after join')
  print(result.head())
  plt.rcParams['figure.figsize'] = [16, 8]
  fig, ax = plt.subplots(1, 1)
  map = result.plot(column='新增確診人數', ax=ax, legend=True)
  map.set_xlim(119, 122.5)
  map.set_ylim(21.5, 25.5)
  ax.set_title('Taiwan Total Confirmed Cases')
  ax.set_xlabel('Longitude')
  ax.set_ylabel('Latitude')
```

分級著色圖如下：

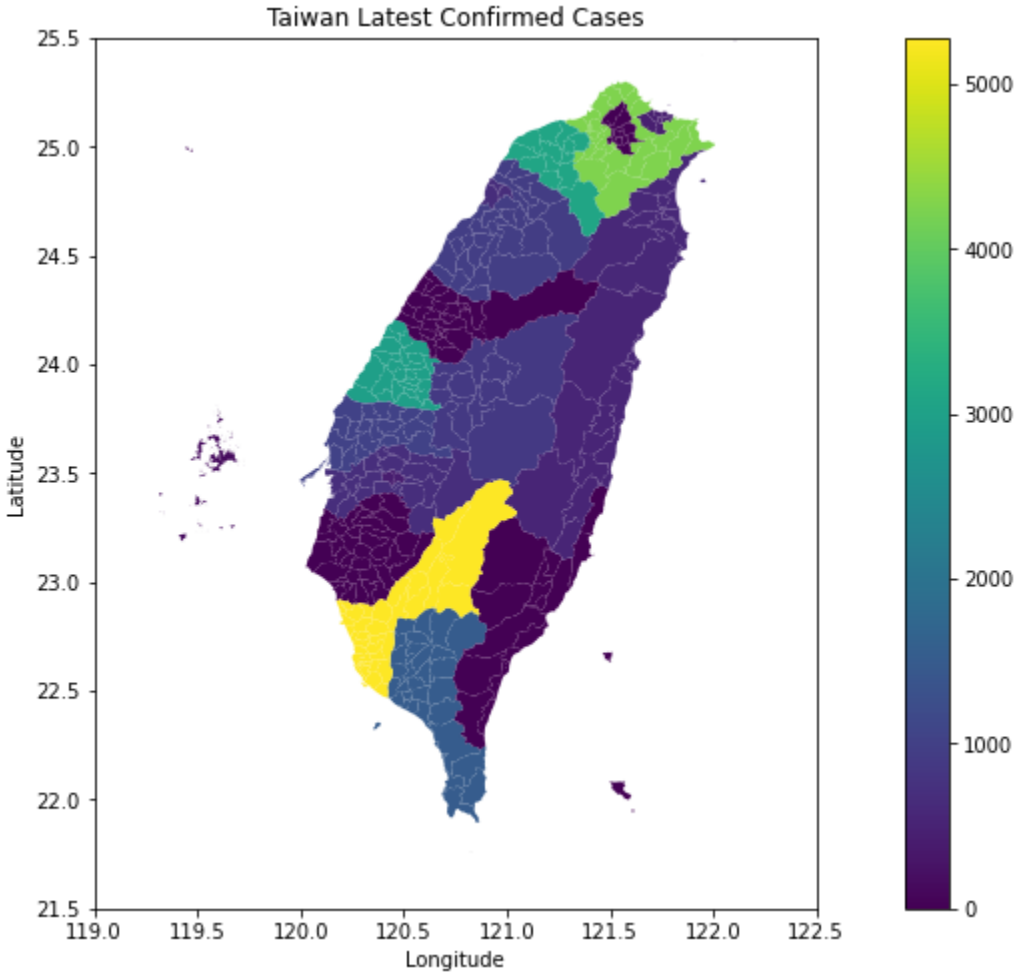


2. Taiwan Latest Confirmed Cases

   在這張分級著色圖中，X 軸顯示經度，Y 軸顯示緯度，各顏色代表各縣市最近一日 covid19 確診數量，程式碼如下：

```python
def plot_taiwan_latest_confirmed_cases(taiwan_data, taiwan_shp):
  latest_taiwan_data = taiwan_data[taiwan_data['個案研判日']=='2022-06-19']
  print('Taiwan latest confirmed cases (Before preprocessing)')
  print(latest_taiwan_data.head())
  latest_taiwan_data=latest_taiwan_data.drop(latest_taiwan_data.columns[[0,3]], axis=1)
  print('Taiwan latest confirmed cases (After preprocessing)')
  print(latest_taiwan_data.head())
  left = taiwan_shp.set_index('COUNTYNAME')
  right = latest_taiwan_data.set_index('縣市別')
  result = left.join(right)
  result = result.fillna(0)
  print('Results after join')
  print(result.head())
  plt.rcParams['figure.figsize'] = [16, 8]
  fig, ax = plt.subplots(1, 1)
  map = result.plot(column='新增確診人數', ax=ax, legend=True)
  map.set_xlim(119, 122.5)
  map.set_ylim(21.5, 25.5)
  ax.set_title('Taiwan Latest Confirmed Cases')
  ax.set_xlabel('Longitude')
  ax.set_ylabel('Latitude')
```

分級著色圖如下：



D. World Map

在這部分的程式中，利用 pandas package 讀取 owl_world_v2_2022-06-21.csv 到 world_data 中，接著對 world_data 去除掉不必要的 attributes，程式碼如下：

```
def load_and_preprocess_world_data():
  URL = "https://covid-19.nchc.org.tw/api/download/owl_world_v2_2022-06-21.csv"
  world_data = pd.read_csv(URL, encoding = 'big5')
  print('Schema before preprocess world_data')
  print(world_data.info(verbose = True, null_counts = False))
  print('Examples before preprocess world_data')
  print(world_data.head())
  world_data = world_data[['iso_code', '總確診數','七天移動平均新增確診數']]
  print('Schema before preprocess world_data')
  print(world_data.info(verbose = True, null_counts = False))
  print('Examples before preprocess world_data')
  print(world_data.head())
  return world_data
```

由於這部分中要畫出地圖，因此還需要全球各國的界線，從 Google Drive 中匯入.shp 檔後，利用 geopandas package 讀取到 world_shp 中，接著對 world_shp 去除掉不必要的 attributes，程式碼如下：

```
def load_and_preprocess_world_shp():
  world_shp = gpd.read_file('./all_countries.shp', encoding='utf8')
  print('Before preprocess world_shp')
  print(world_shp.head())
  world_shp = world_shp[['iso_a3','geometry']]
  print('After preprocess world_shp')
  print(world_shp.head())
  return world_shp
```

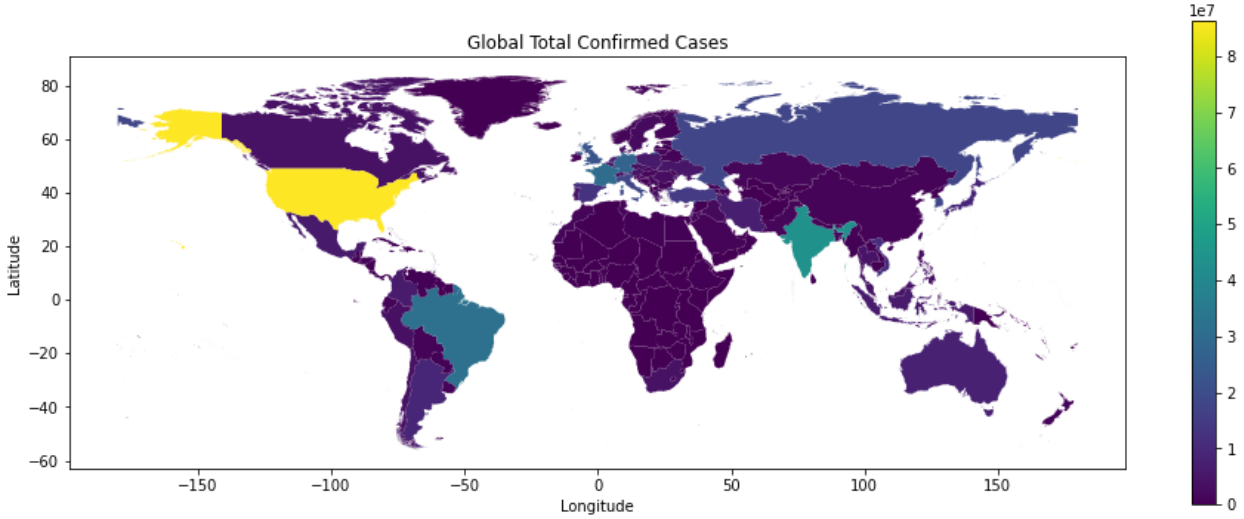
在處理完 world_data 與 world_shp 後，利用 pandas dataframe 的語法分析資料並畫出兩張分級著色圖，分別如下：

1. Global Total Confirmed Cases

   在這張分級著色圖中，X 軸顯示經度，Y 軸顯示緯度，各顏色代表各國累計 covid19 確診數量，程式碼如下：

```
def plot_world_total_confirmed_cases(world_data, world_shp):
  left = world_shp.set_index('iso_a3')
  right = world_data.set_index('iso_code')
  result = left.join(right)
  plt.rcParams['figure.figsize'] = [16, 6]
  fig, ax = plt.subplots(1, 1)
  map = result.plot(column='總確診數', ax=ax, legend=True)
  ax.set_title('Global Total Confirmed Cases')
  ax.set_xlabel('Longitude')
  ax.set_ylabel('Latitude')
```

分級著色圖如下：

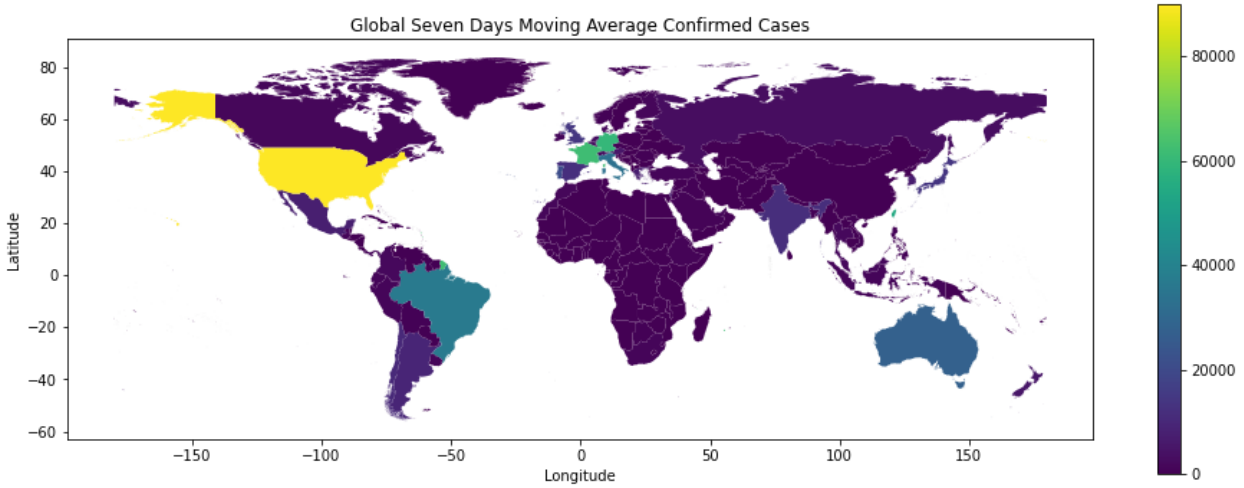


2. Global Seven Days Moving Average Confirmed Cases

在這張分級著色圖中，X 軸顯示經度，Y 軸顯示緯度，各顏色代表各縣市最近七日移動平均新增 covid19 確診數量，程式碼如下：

```
def plot_world_moving_avg_confirmed_cases(world_data, world_shp):
  left = world_shp.set_index('iso_a3')
  right = world_data.set_index('iso_code')
  result = left.join(right)
  plt.rcParams['figure.figsize'] = [16, 6]
  fig, ax = plt.subplots(1, 1)
  map = result.plot(column='七天移動平均新增確診數', ax=ax, legend=True)
  ax.set_title('Global Seven Days Moving Average Confirmed Cases')
  ax.set_xlabel('Longitude')
  ax.set_ylabel('Latitude')
```

分級著色圖如下：



VI. Comparison between SQL and Pandas in Python

這裡會比較在 python 中使用 SQL 與使用 pandas 的一些差異

A. SQL

1. SQL 是一個用來與 database 溝通的語言，大部分的 relational database management system 都用 SQL，可以用來處理 tabular data
2. 需要較多的 database domain knowledge，學習不易
3. 語法中常需要巢狀迴圈
4. 需要手動輸入 schema
5. 能夠對 database 操作的可能性較多
6. 在大型 table 中效率較高，因此如果效能要求高，先使用 SQL 比較好

B. Pandas

1. Pandas 是一個 python library，可以用來處理 tabular data
2. 比較容易上手，且 python 本身好用、延伸性佳，比較方便 develop 和 debug
3. 不需要許多巢狀迴圈
4. 可自動判斷 schema
5. 能夠對 database 操作的可能性較少
6. 在大型 table 中效率較低