SI 206 Final Report

Jia-Tong Choo, Xiyu Hu
https://github.com/jtchoo/SI206-FinalProject

Goals

- We planned to use two APIs which one has the stringency of a government and the other has the confirmed and death cases of COVID-19, both for a list of different countries.
- Compare two APIs and see if the stringency of a government actually plays an important role in COVID-19.
- Create at least two visualizations to help us understand how different countries' governments are responding to the pandemic.

Goals Achieved

- Web scraped from a website using beautiful soup to convert a country's Alpha-3 code to its full country name as well as using an API to get all confirmed/death cases and government stringency.
- Successfully calculated the death rate for each country and matched them from their Alpha-3 code to the full country name.
- Get both their actual stringency and legacy stringency to compare and get what the government is doing now can be improved.
- Created two visualizations. One focuses on the COVID-19 cases and the other focuses on how the government is doing.

Problems

- At first, we tried to use two APIs but later found out that there's one API that only has the Alpha-3 code and the other only has the full country name. To solve this, we found a website that has a table showing which Alpha-3 code is corresponding to which country name.
- The API that we are using is pretty messy with all the data. We struggled a bit on getting what we actually wanted and using the JSON beautiful printer online really helped us with getting some specific information.
- At first, we did not fetch any data that has multiple same data in a column from the API, which caused us not to be able to use join in order to do the calculations. We ended up fetching the data from the API and importing it to our table.

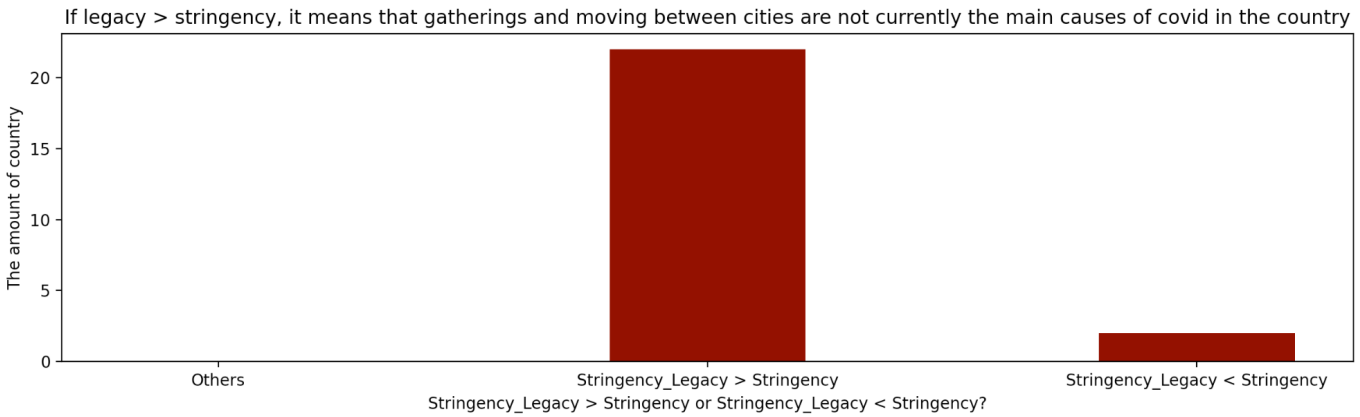
Calculation

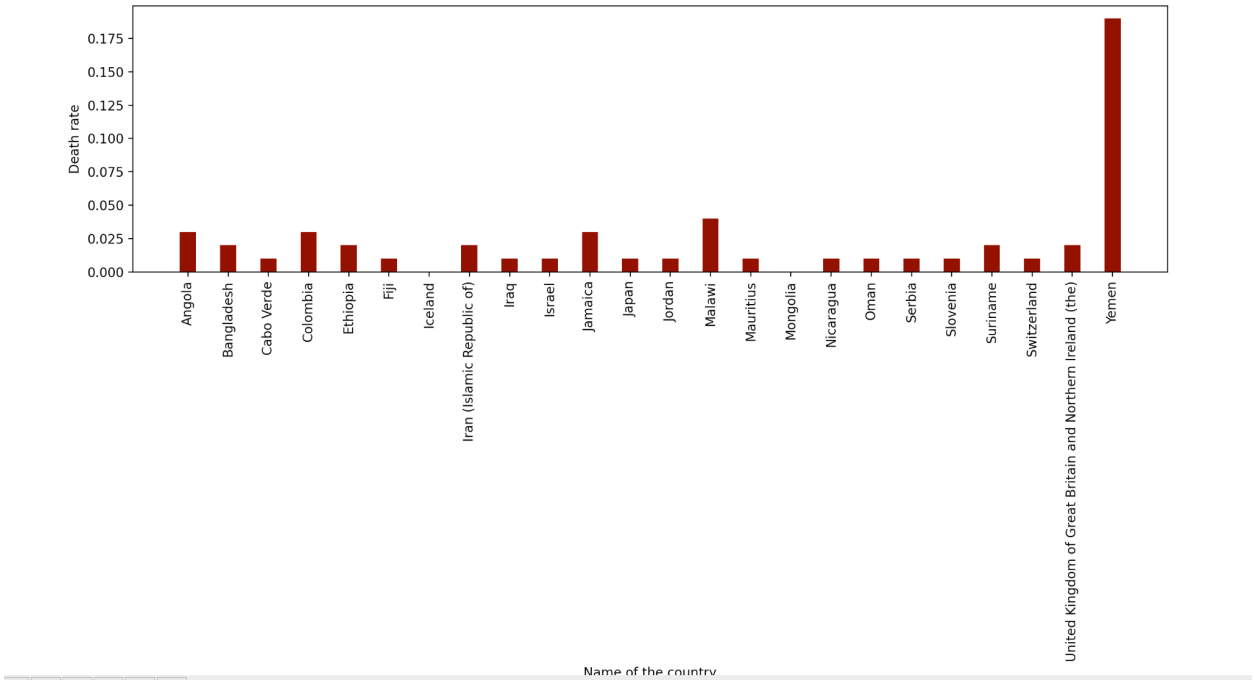
country_death_rate.txt

```
The death rate
======================================================================

Angola's death rate is 0.03

Bangladesh's death rate is 0.02

Cabo Verde's death rate is 0.01

Colombia's death rate is 0.03

Ethiopia's death rate is 0.02

Fiji's death rate is 0.01

Iceland's death rate is 0.0

Iran (Islamic Republic of)'s death rate is 0.02

Iraq's death rate is 0.01

Israel's death rate is 0.01

Jamaica's death rate is 0.03

Japan's death rate is 0.01

Jordan's death rate is 0.01

Malawi's death rate is 0.04

Mauritius's death rate is 0.01

Mongolia's death rate is 0.0

Nicaragua's death rate is 0.01

Oman's death rate is 0.01

Serbia's death rate is 0.01

Slovenia's death rate is 0.01

Suriname's death rate is 0.02

Switzerland's death rate is 0.01

United Kingdom of Great Britain and Northern Ireland (the)'s death rate is 0.02
```

VIsualizations

***These are the visualizations that we got after running it ONCE. There will be more data/countries showing as you run more times.**





Instructions for our code

1. Run the "**SI206_final.py**" file. After running, check if you have the covid.db file and there will be a "Country" table in the database.

2. Run the "**API2.py**" file. You will see there's another table showing which is "CounrtyCode". It will only have 25 if you only run once. Run the program 4 times to make the table store 100 rows.

3. After having two tables, run the third python file, which is the "**gathering_data.py**" file. It will create a txt file in your folder that has our calculation data in it. At the same time, it will also pop up the visualization for both our charts.

Documentations

SI206_final.py

```
def setUpDatabase(db_name):
    """
    Set up the database and the input should be the name of your choice.
    Returns the cursor and the connection to the database.
    """

def get_country():
    """
    No inputs. Request the website url that we are going to scrape from.
    Return a dictionary that has the country full name as the key and Alpha-3 code as the value.
    """


def setUpCountryTable(data, cur, conn):
    """
    Takes the dictionary that we got from get_country(), the cursor, and the connection of the database as inputs.
    Will create the "Country" table.
    Insert countries' full name and their Alpha-3 code into the table.
    Returns nothing.
    """



def main():
    """
    Takes nothing as input and call the functions above
    """
```

API2.py

```
def create_country_table(cur, conn):
"""
Takes the cursor and the connection to the database as inputs.
Loads the API data into the json file.
Create the "CountryCode" table
Insert the data into the table.
Returns nothing.
"""


def setUpDatabase(covid_cond):
"""
Set up the database and the input should be the name of your choice.
Returns the cursor and the connection to the database.
"""


def main():
"""
Takes nothing as input and call the functions above
"""
```

gathering_data.py

```
def join_table(cur,conn):
"""
Takes the cursor and the connection to the database as imputs.
Join the two tables and select the data that we need for the later calculations.
Return a tuple of the data that we need.
"""

def join_table2(cur,conn):
"""
Takes the cursor and the connection to the database as inputs.
Join the two tables and select the data that we need for the later calculations.
Return a tuple of the data that we need.
(The difference between this join and the previous join is - we select different datas.)
"""

def calculation(cur,conn):
"""
Takes the cursor and the connection to the database as inputs.
Returns a list of the result of our calculations.
"""

def write_data_file(file_name, cur, conn):
"""
Takes the file name, the cursor and the connection to the database as inputs.
Write things into the file.
Return nothing.
"""

def country_list(cur,conn):
"""
Takes the cursor and the connection to the database as inputs.
Returns a list of country that we are going to use in later visualizations.
"""
```

```
def death_rate_list(cur,conn):
"""
Takes the cursor and the connection to the database as inputs.
Returns a list of death rate that we are going to use in later visualizations.
"""

def calculation_stringency(cur,conn):
"""
Takes the cursor and the connection to the database as inputs.
Calculate whether the legacy_stringency is larger than the actual_stringency.
Return a list of amounts for both.
"""


def visualization1(cur, conn):
"""
Takes the cursor and the connection to the database as inputs.
Create both stringency chart and the death rate chart.
Return nothing.
Pop up the two visualizations.
"""


def main():
"""
Takes nothing as input and call the functions above
"""
```

Resources

Date	Issue Description	Location of Resource	Result
11/16	Needed to find two APIs with useful and relevant data	https://covidtracker.bsg.ox.ac.uk/about-api	The theme and contents were able to be used by us.
11/18	Confused about too many dictionaries in our API's data. Hard to extract the data we wanted.	http://jsonviewer.stack.hu/	Showed me the clarified structure of my API and made me understand and was able to extract the matched parameters.
11/30	Trying to use beautiful soup to fetch a table from a website.	https://stackoverflow.com/questions/2935658/beautifulsoup-get-the-contents-of-a-specific-table	Managed to use beautifulsoup to extract the country name and its corresponding Alpha-3 code text from a website.

12/1	Needed to select data from API and then set up a database to create a table	https://datatofish.com/create-table-sql-server-python/	Used cur, conn(SQL command) to select the data.
			`cur.execute('CREATE TABLE IF NOT EXISTS CountryCode ("code" TEXT PRIMARY KEY, "confirmed" INTEGER, "deaths" INTEGER, "stringency" FLOAT, "date" TEXT, UNIQUE("code", "date"))')`
			`conn.commit()`

12/2	Needed to join the two tables	https://www.w3schools.com/python/python_mysql_join.asp	Using SQLcommand: `"SELECT Country.Country, CountryCode.deaths, CountryCode.confirmed,`

			`CountryCode.stringency FROM Country JOIN CountryCode WHERE Country.threeDigits = CountryCode.code"` I could select the data I wanted and join two tables together to do the further calculation.
12/3	Cannot find the result of the tuple after we tried the JOIN method.	https://github.com/toriastahl/Final-Project	We should set an empty list first and append the result of join_ tables into that empty ist.
12/6	Trying to get the second value in a group of “td” with beautiful soup	https://stackoverflow.com/questions/38028384/beautifulsoup-difference-between-find-and-select	Manage to get it with using .select to get its second child
12/8	The values of the x-axis are overlapping each other	https://stackoverflow.com/questions/10998621/rotate-axis-text-in-python-matplotlib	Was able to turn them 90 degrees so they won’t overlap
12/9	Needed to make a bar chart using matplotlib	https://www.geeksforgeeks.org/bar-plot-in-matplotlib/	I was able to create a more complex bar chart
12/11	Hard to find the implication when we calculate the difference between stringency actual and stringency legacy.	https://www.bsg.ox.ac.uk/sites/default/files/Calculation%20and%20presentation%20of%20the%20Stringency%20Index.pdf	This resource helped us know the stringency legacy is what the government is doing according to 7 out of 9 policy levels.