SI 206: Final Project- APIs, SQL

SAL: Lauren Fulcher, Sarah Whitman, Abby Williams

https://github.com/adwill1/206-final-project

Goals

Original goals: Our goal for this project was to find whether correlations existed between the wealth of a country and covid vaccinations around the globe. We also wanted to look at vaccination rates by continent to see if there was any correlation. We were going to try to pull information from 2 APIs and web scrape from a website to gather our data.

Achieved goals: We were able to notice a relationship between wealth and percent vaccinated, and wealth and life expectancy in a country. We were also able to distinguish the people vaccinated in each continent. These relationships are shown in our visualizations. We successfully gathered our information from our selected APIs and website.

Problems that you faced:

One of our biggest problems that we encountered was our API going down in the middle of working on the project. We were originally looking into cases and deaths from COVID instead of vaccination numbers. However, the information was wiped from that part of API. So, we had to switch to a different part of the API to gather vaccination data. Thus, we had to alter many parts of our code to accommodate this change.

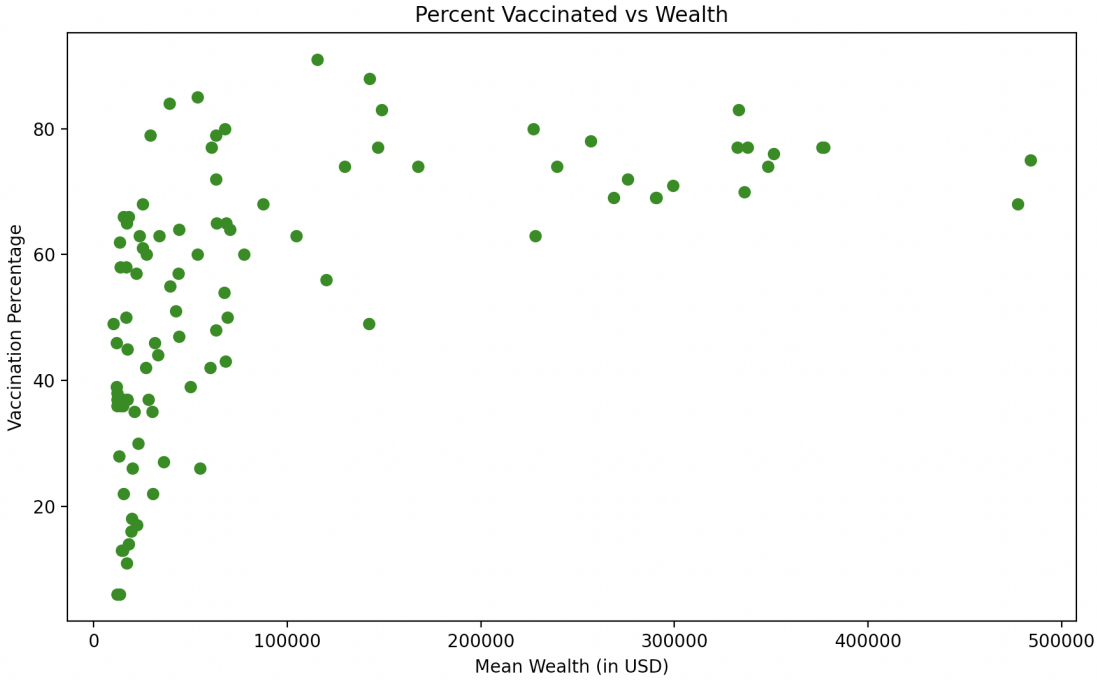
Another problem that we faced was figuring out how to only add 25 items to the database for each time the code executes. We were able to limit the number to 25 but were struggling with how to add a new 25 the next time the code executed. Eventually, we figured it out.

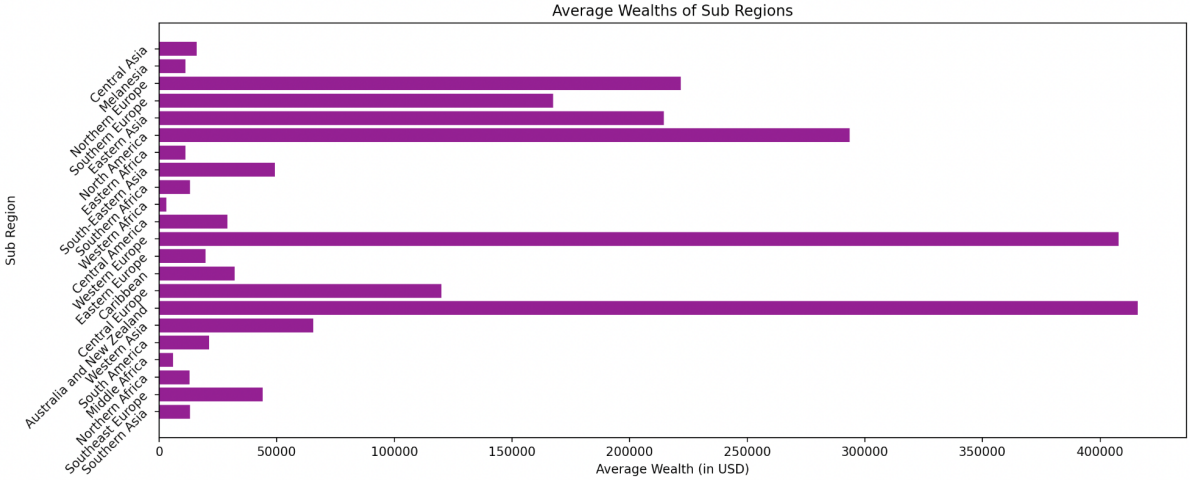
Something else that we faced was creating meaningful visualizations. There were plenty of options for creating visualizations with our data. However, we wanted to make sure that there would be some sort of learning from the output of the visualization.

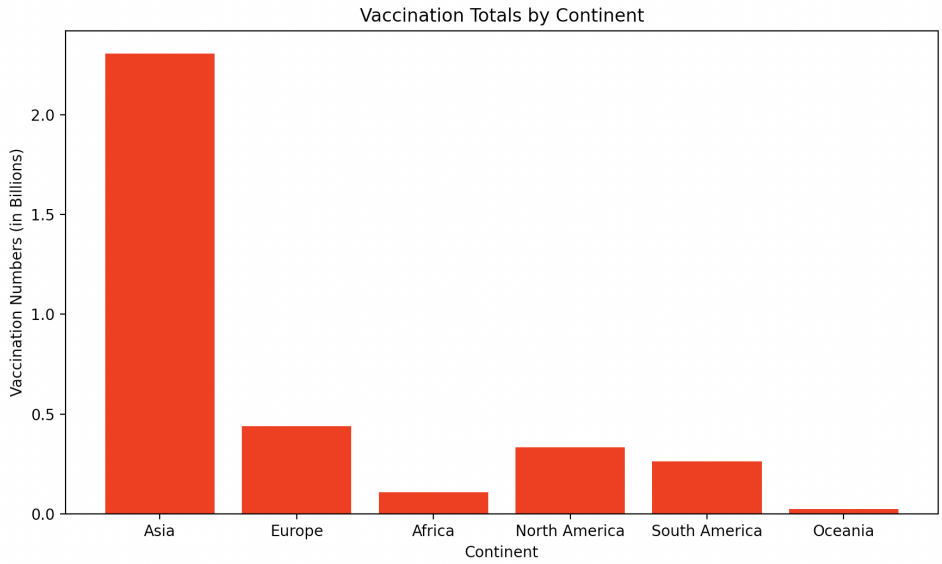
Calculation file CSV Output:

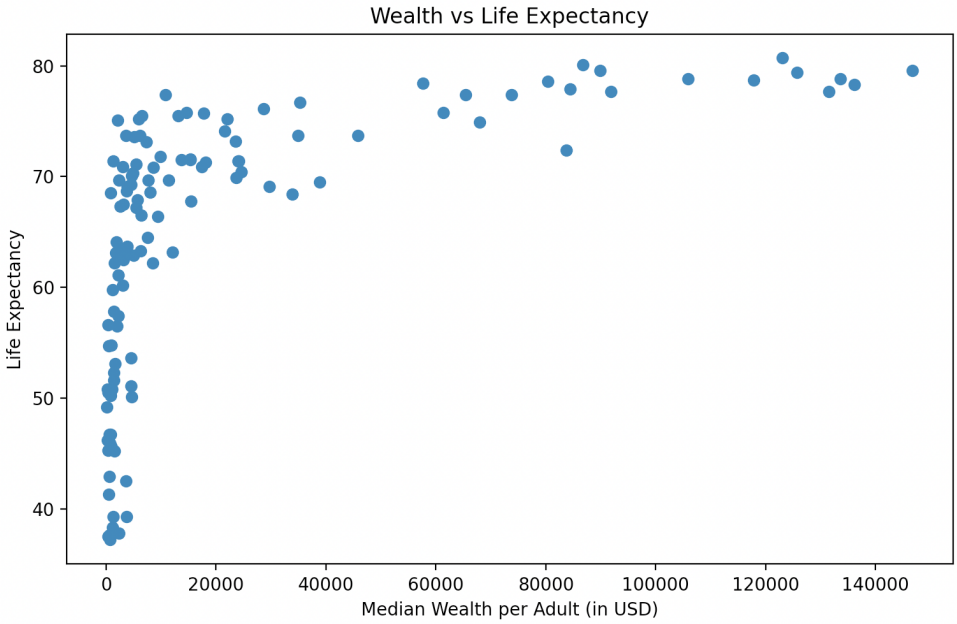
```csv
Country,Percent Vaccinated
Albania,35
Angola,10
Argentina,68
Armenia,17
Australia,75
Austria,69
Azerbaijan,46
Bahrain,68
Bangladesh,26
Barbados,48
Belarus,30
Belgium,76
Belize,49
Benin,7
Bolivia,37
Bosnia and Herzegovina,22
Botswana,22
Brazil,66
Brunei,84
Bulgaria,27
Burkina Faso,2
Burundi,0
Cambodia,81
Cameroon,2
Canada,77
Central African Republic,7
Chad,0
Chile,85
China,83
Colombia,51
Comoros,28
Costa Rica,64
Croatia,50
Cyprus,49
Denmark,77
Djibouti,3
```

Visualizations:


Percent Vaccinated vs Wealth
Vaccination Percentage
Mean Wealth (in USD)
20
40
60
80
0
100000
200000
300000
400000
500000


Average Wealths of Sub Regions
Sub Region
Average Wealth (in USD)
Central Asia
Melanesia
Northern Europe
Southern Europe
Eastern Asia
North America
Eastern Africa
South-Eastern Asia
Southern Africa
Western Africa
Central America
Western Europe
Caribbean
Eastern Europe
Central Europe
Australia and New Zealand
Western Asia
South America
Middle Africa
Northern Africa
Southeast Europe
Southern Asia
0
50000
100000
150000
200000
250000
300000
350000
400000





Instructions for running code:

1. Make sure that “Combined.db” does not already exist as a database in your computer files. If it is, delete it. This is necessary so that the database can be created.
2. The first file that needs to be run is “combined.py”. This file needs to be run 10 times, so that it gathers all of the data while adhering to the 25 count limit.
3. Open “Combined.db”. You should see 4 tables in the database.

4. Next, “calculations.py” must be run only once. This will do the calculations, write the CSV file for data output, and also produce the 4 visualizations of our data.

Code documentation:

In order of combined then calculations

Function Name	What it does (including input & output)
get_countries_from_api( )	No inputs. Calls to the covid API to get all of the country names. It returns a list of all country names.
get_continents_from_api( )	No inputs. Calls to the covid API to get all of the continent names. It returns a list of all unique continent names.
get_ppl_vax_from_api( )	No inputs. Calls to the Covid API to get the vaccination number from each country. It returns a list of the vaccination numbers.
get_life_ex_from_api( )	No inputs. Calls to the Covid API to get the life expectancy from each country. It returns a list of the life expectancy or “N/A” if that information was not found in the API.
create_full_dictionary( )	No inputs. This calls the first four functions to get the info from the API. Then it combines the information into a dictionary with the keys being the Country name and the value being a sub-dictionary that has people vaccinated, life expectancy, continent, and creates a unique country id. This dictionary is returned.
create_continents_table(cur, conn, data_dictionary)	Takes in database cursor and connection as inputs, as well as the dictionary that is outputted from create_full_dictionary. This creates the continents table in the database; nothing is returned.
create_covid_info_table(cur, conn, data_dictionary)	Takes in database cursor and connection as inputs, as well as the dictionary that is outputted from create_full_dictionary. This creates the CovidInfo table in the database; nothing is returned.
get_country_name( )	This function does not take any inputs. This is the function that web scrapes the wikipedia

	page to get each country's name. It returns a list of the data.
get_dist_weath_mean( )	This function does not take any inputs. This is the function that web scrapes the wikipedia page to get each country's mean wealth. It returns a list of the data.
get_dist_wealth_median( )	This function does not take any inputs. This is the function that web scrapes the wikipedia page to get each country's median wealth. It returns a list of the data.
create_dict(country_list, means, medians)	This takes in the lists from the three web scraping functions above, and organizes them into a dictionary. The key is the country name, the value is a tuple of (mean wealth, median wealth)
setUpWealthDB(cur, conn, wealth_dict)	This takes in the wealth data dictionary and creates the table if it doesn't exist with country name, mean wealth, and median wealth. It fills this table with the data dictionary.
get_europe_data( )	No inputs. This calls the countries rest API for countries in the europe region. It returns a list of tuples containing the country, population, and sub region.
get_americas_data( )	No inputs. This calls the countries rest API for countries in the Americas region (North and South America). It returns a list of tuples containing the country, population, and sub region.
get_africa_data( )	No inputs. This calls the countries rest API for countries in the africa region. It returns a list of tuples containing the country, population, and sub region.
get_oceania_data( )	No inputs. This calls the countries rest API for countries in the oceania region. It returns a list of tuples containing the country, population, and sub region.
get_asia_data( )	No inputs. This calls the countries rest API

	for countries in the asia region. It returns a list of tuples containing the country, population, and sub region.
combine_list(europe, americas, africa, oceania, asia)	Takes a list of tuples (from each function that calls the api) and combines them into one list. Adds an id to each country. Returns a list sorted by country name of tuples with country id and country, population and subregion
setUpCountryDatabase(cur, conn, country_list)	Inputs a list of country id and tuple (from combined_list) and creates a table to input data from the country_list. Returns nothing
setUpDatabase(db_name)	It takes the name of the database as a string input. It returns the cursor and the connection to the database.
main( )	Takes no inputs and returns nothing. Calls all of the functions; creates the database and produces the visuals.
calc_percent_vaccinated(cur, conn)	Uses JOIN to select the mean wealth from wealth table, population from Country information table, number of vaccinated and country from the CovidInfo table. The data then calculates the percent of vaccinated out of the population of a country and is then put into a dictionary with the country as the keys and the wealth and percent as the values which is returned
create_percent_vax_vis(percent_dict)	This takes in a dictionary with the key as the country name and the value as a tuple of corresponding wealth per adult and the calculated percent of vaccinated population. It uses matplotlib to create a scatter plot of each country's percent vaccinated and mean wealth.
wrtie_csv(file_name, percent_dict)	Inputs a file name and dictionary with percentages of percent vaccinated per country. Writes the data to a csv file with country and percent vaccinated as rows. Returns nothing
get_wealth_of_subreg(cur, conn)	Use JOIN to select the mean wealth from the Wealth table and sub region from the Country

	information table. Returns a list of tuples, with these two pieces of data as the tuple
create_subreg_mean_dict(full_list)	Takes in the list of tuples of mean wealth and subregion, and creates a dictionary with the subregions as the keys, and a list of all of its wealths as the value. Returns this list.
calc_avg_wealth(full_dict)	Takes a dictionary of subregions and their wealths, and calculates the average wealth in each value of the dictionary (by finding the sum of the wealths and then dividing by the length of the list). Returns the results in another dictionary where average wealth is the value of the subregion key.
create_wealth_subreg_vis(org_dict)	Takes in a subregion (key) and average wealth (value) dictionary, and creates a horizontal bar graph with the information using matplotlib. Does not return anything.
get_continent_vaxxes(cur, conn)	Takes in database cursor and connection as inputs. It selects the country, continent, and people vaccinated from both the continents and covidInfo tables using join. It returns this list of tuples.
create_cont_vax_dict(info_list)	Takes in the tuples list from get_continent_vaxxes as input. It creates a dictionary from the tuples with the continent name as the key and a list of all the vaccination numbers from that continent as the value. It returns this dictionary.
calc_cont_vax_total(cont_vax_dict)	Takes in the dictionary from create_cont_vax_total. Under each continent key, the value is a list and this function sums the values in the list to get the total vaccination numbers by continent and then divides by 1 billion. It creates a new dictionary with the continent as the key and the total vaccination number as the value. It returns this dictionary.
create_cont_vax_visual(total_dict)	Takes in the dictionary from calc_cont_vax_total as input. It creates a list of the x values from the continents in the

	dictionary and the y values from the vaccine totals in the dictionary. It plots this information as a bar graph and changes the color of the bars to red.
calc_avg_life_exp_of_cont(cur, conn)	This selects the life expectancy from our CountyInfo table and the median wealth from our Wealth table. It returns a list of tuples that contain these two pieces of data for each corresponding country.
create_extra_vis(list_tups)	Takes in a list of tuples that contain a median wealth and the life expectancy for a country. Creates a scatter plot of the data using matplotlib. Does not return anything.
main()	Takes no input. This calls all of the functions to do the calculations and show all of the visualiations.

Documentation of resources used:

Date	Issue Description	Location of Resource	Result (Did it solve the issue?)
12/4/21	We were having trouble figuring out how to have only 25 lines added to the database at a time	https://www.w3schools.com/sql/sql_count_avg_sum.asp	No; we went to office hours for a better understanding
12/5/21	We were having trouble figuring out how to have only 25 lines added to the database at a time	https://stackoverflow.com/questions/759244/sql-server-the-maximum-number-of-rows-in-table	No; we went to office hours for a better understanding
12/12/21	The wealth of subregion visualization was long and hard to fit on one page, so we needed to slant the x axis titles	https://stackoverflow.com/questions/10998621/rotate-axis-text-in-python-matplotlib	Yes, the graph is more easily readable now

12/12/21	We wanted to change the colors of our plots, but could not find slides on it	https://www.codespeedy.com/how-to-change-line-color-in-matplotlib/	Yes, we were able to change color
12/10	We did not know how to combine more than two tables when selecting data from a database.	https://stackoverflow.com/questions/5099420/sql-join-format-nested-inner-joins	Yes, we joined three tables together in one select execution.