**Documentation of the Census API Python Scraper**

The tool, located in [**M:\Research\Price CSI\Research\Projects\SOLACHAN\Data\Census API\Python**], is a single python file which imports several modules and is composed of four individual functions. The tool scrapes census.api.gov using their developer resource, navigating to the datasets, years, and geographies desired by the user.

Each time the code is run it takes in only one dataset in order to minimize errors and exercise more control over the collection of data. Each year of data is stored in a unique file, titled [DATASET\_YEAR.csv]. Once a dataset has been downloaded in full, it is placed in a unique file and moved to [**M:\Research\Price CSI\Research\Projects\SOLACHAN\Data\Census API\Python\Raw Data**].

Currently, for the purposes of the SOLACHAN project, the code expects to receive data from ACS 1-Year Estimates, either in the form of a detailed table beginning with “B” or “S” or a data profile table beginning with “DP.” Each type of data has a unique url structure, and therefore if another type of data is to be pulled, it would require adding a new condition within the *create\_path\_list* function using the data’s code and url structure.

The Census API stores table data as JSON-encoded content, meaning that when using the requests module to view url content, content must be interpreted in terms of JSON elements.

After giving the code table input, check the [**Census API\Python**] folder. You should see a file created for the first year of data in the dataset. If the size of the file has increased from 0KB, then data has been written. Do not open the file while the code is running. The file size will continue to increase as data is written to the file. When all the available data has been pulled for a year, the code will move on to the next year.

Imports

The script imports a few modules (these sometimes have to be downloaded via Pycharm or the command prompt) which allow the code to execute special functions. The imported modules central to this code are:

**import csv**: allows the code to create and write to csv files

**import requests:** allows the code to redirect to a webpage and read the contents

Function: def main()

This function, located at the bottom of the code, establishes and contains all information necessary to allow all other functions to run, and determines how the code is executed. In this function, the following variables are contained:

**table**: this line prints a prompt to the user which appears once the code is run. The user types their desired table code (ie. B25001, DP04), presses enter, and the table code is stored in the table variable used to create the website path list

**years:** this is a list of years that the user would like data on. This list can be edited, but years must be written as strings (contained in quotation marks).

**geos:** this is a list of the desired geography filters, as the census API expects to receive them. This list can be edited, with each individual geography as a string in the correct code format. For reference, see: <https://api.census.gov/data/2023/acs/acs1/examples.html>

**key:** this is a key derived from the Census API website “Request Key” form. If code does not output data, check a url from the path list manually. If the url does not work, or an error message is displayed, a new key might be needed.

**path\_list:** list of paths created using table code, year list, geography list, and key

**file\_list:** list of file names that the code will write data to

The very last line of the code must be “main()” otherwise the code will not run.

Function: def create\_path\_list()

This function takes in the table code, list of years, list of geographies, and the Census API key and creates a list of paths (urls) from which to collect data.

Function: def create\_filename\_list()

This function takes in the table code and the list of years and combines them to create a list of files that the code will write data to.

Function: def get\_data()

This function loops through every path and year, sending a request to each path, reading the content in JSON, and outputting data from each path to the relevant file. In order to monitor the progress of the code and check for any errors, each time data is written a line is printed identifying the file.

**Steps:**

1. Download Pycharm Community Edition (or any Python interpreter) and the latest version of Python
2. Open the file [**M:\Research\Price CSI\Research\Projects\SOLACHAN\Data\Census API\Python\census\_api\_tool.py**] in Pycharm
3. Run the file
4. Immediately, you should see the prompt, “Enter table code:” Type the table code you wish to use, and press Enter
5. The code will display a list of paths and a list of files and then the message, “Writing to files”
6. Wait until you see the message, “Done.”
7. Check if the files that have been created are as desired

**Errors**

Errors may occur for the following reasons:

1. Network issue/timeout: If a code is running smoothly and displays an error unexpectedly, it is most likely a random disconnection from the network. Delete files created for the current dataset and run the code again with the same input. OR Delete the last file (the one that is incomplete), update the years to exclude those that are complete, and run the code again for the same dataset—just remember to add the full year list back for the next dataset.
2. Data not found: If the code is running but no data is being written to files, check the table code and the urls manually, as the requests may not be reaching the JSON elements they expect.