Resources

# U.S. Passport Data

<https://www.kaggle.com/jboysen/us-perm-visas#us_perm_visas.csv>

# Census Entrepreneurial Data

<https://www.census.gov/data/developers/data-sets/ase.html>

Our database is called ETL\_project and has 13 tables. One of the issues we struggled with was that our data files were very large. This caused issues pulling/pushing into Git and caused our code in Python to run slowly. A solution we created was to transform

Objective:

The objective is to create a database that combines the U.S. Passport data with the U.S. Census data. To complete this we used the ETL (export, transform, load) method to create a database in SQL. The final database includes the cleaned data of the Census & Passport data, and several crosswalk files, explained in more detail below.

Final Deliverable

Our final project has 13 tables with two main datasets, Census & Passport data, and one large dataset of the Census & Passport data joined together. The Census data’s categorical variables were separated into smaller crosswalk files, used to reduce the size of the dataset. All of our files were exported as CSVs for future use.

The Passport data contained messy data, with many blank entries and categorical entries being listed in varying ways (example: the state California would be CA & CALIFORNIA.) This made it difficult for the final merge between the Census & Passport data. In order to accomplish the merge, we found a crosswalk file of U.S. state & abbreviation to match the Passport data on first state, then abbreviation, and then use the UNION function in SQL to append the files. After this dataset was created, we merged this set back into the unique list of state/abbreviation combination. This created a dataset of Passport data that had both the correct format of State for each row. This data was joined to the census data using an inner join. In its current form, the data after this last merged increased as the census data had each state listed 6 times (one for each category of years in business.)

For future projects, it would be useful to design a strategy that could aggregate this metric to prevent the duplication of rows.

# Data Process & Report

## Visa Passport Data

### Python

##### Steps

1. Export
   1. Import dependencies – pandas, sqlalchemy, numpy, datetime, config
   2. Import raw files from Kaggle, “us\_perm\_visas.csv”
2. Transform
   1. Remove unnecessary columns
   2. Combine similar columns
   3. Calculate adjusted Yearly Wage
   4. Remove duplicated columns
   5. Set Case Number as index
3. Load
   1. Make connection to PGAdmin to read in reduced data into a table
   2. Additional cleaning of cells

## PGAdmin

#### Steps

1. Load
   1. Create database ETL\_project
   2. Create table of variables
   3. Joined State/Abbreviation table to Passport data to make employer state match state format of Census data
   4. Joined Census data with Passport data
      1. This was not a perfect match, tripling the size of the dataset. Is currently unclear why this occurred, will be an opportunity for further analysis

## Census Data

### Python

#### Steps

1. Export
   1. Import dependencies – pandas, numpy, config files, sqlalchemy, config
   2. Import raw file from Census, “ASE\_2016\_00CSCBO01.csv”
2. Transform
3. Remove unnecessary columns
   1. File had two rows of headers, removed top headers and made second row of header primary headers
   2. Printed name of remaining columns
   3. Removed spaces and reduced verbosity of column names to fit in SQL data
   4. Separated files into integer columns and character columns
   5. Made copies of character columns into categorical columns
   6. Used .cat function to create numerical values associated with various categories in rows
      1. Purpose of this is to reduce the size of the final database size by removing text characters and replacing with numbers
   7. Printed out column names with SQL syntax, indicating as numeric or character
4. Load
   1. Exported copy of file as CSV
   2. Manually imported excel file into PGAdmin

## PGAdmin

#### Steps

1. Transform
   1. Created crosswalk files between categorical data and corresponding category numbers
   2. Created reduced table that dropped text versions of categorical data
2. Load
   1. Created table with Census file exported from Python
   2. Exported csv file with numeric versions of categorical data
   3. Exported csv files of all crosswalk files