Extract, Transform, & Load (ETL) Project

2016 Primary Results



Brian Labelle

Richard Harris

Rice University Data Analytics Bootcamp

September 7, 2019

13-ETL-Project

Huy Nguyen

**E**xtract

For this assignment, our group asked the question does income level influence a voter’s selection during the presidential election cycle. In order to do this, we selected the following three datasets in the table below to assist us with performing this analysis.

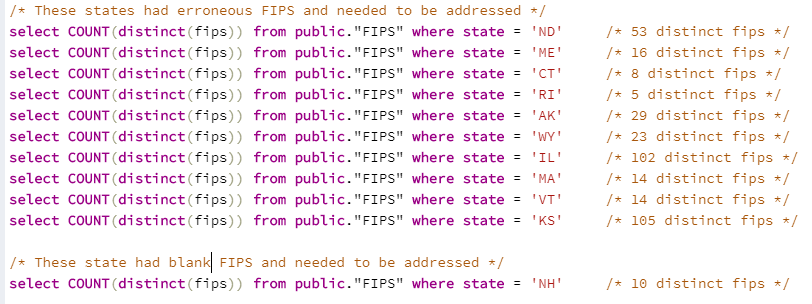
|  |  |  |  |
| --- | --- | --- | --- |
| **Datasets** | **Source** | **Format** | **Purpose** |
| 2016 US Election – Primary Results | Kaggle | CSV | Used to determine which candidates received the most votes by county and FIPS |
| Individual Income Tax Statistics | Kaggle | CSV | 2014 income tax data by zip code |
| US Zip Codes to County to FIPS Crosswalk | Kaggle | CSV | Zip Code to FIPS cross-reference |

**2016 US Election – Primary Results:** The fields included in this dataset are state, state abbreviation, county, FIPS, party, candidate, votes, and fraction votes. We used this file to get candidate’s votes by FIPS, which is the census ID number for each county in the US. Zip codes were not provided with this data. This file was imported directly into Jupyter Notebook using Pandas.

**Individual Income Tax Statistics:** The fields included in this dataset are state, zip code, adjusted gross income (AGI), salaries & wages, and other selected tax items. The latest year available was 2014. There were no county FIPS ID available in this file. This data was also retrieved directly into Jupyter Notebook using Pandas.

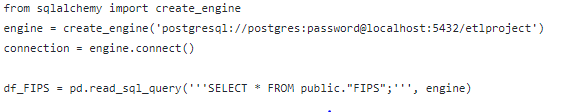
**US ZIP Codes to County to FIPS Crosswalk:** The fields included in this dataset are zip code, county, state abbreviation, and FIPS. This file enables us to merge the election results with the income tax data because it contains both zip codes and FIPS. This file was pulled into Postgres then subsequently retrieved in Pandas.

**T**ransform

Before retrieving the data into Jupyter Notebook, we had to clean the data in the US Election dataset using Postgres SQL because there were states that either did not have a FIPS IDs or the ID was erroneous. The errors were cleaned by applying the FIPS from randomly generated county value in the state with the correct FIPS to the blank or erroneous one in that same state. We noticed errors in the primary elections table with Vermont having over 240 distinct counties where in fact Vermont only has 14.  
  
  
Once the “cleaned” dataset was retrieved into Pandas, we used “converters” in the pd.read\_csv script for leading zeros since FIPS is a five-digit code (the code is listed below).



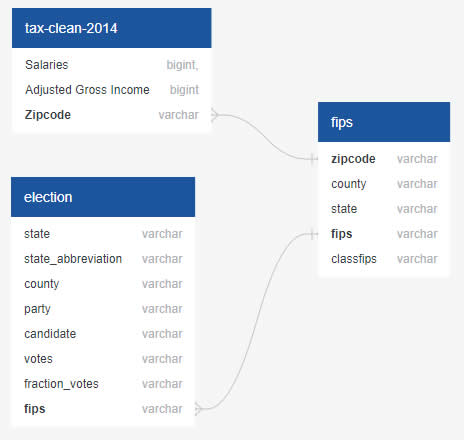
Next, we imported the Zip Code to FIPS Crosswalk into a Postgres Database and created a table via SQL. Then, using SQLAlchemy, the data was retrieved into Panda as follows:



After retrieving the Crosswalk, we could begin merging the first two sets of data. We had to rename the headers in the US Election dataset, so that we could merge on the field “FIPS”. Once the data was merged, duplicate columns such as county and state were dropped from the dataset because they were no longer needed.

The 2014 income tax dataset was imported into pandas last. Again, this data was needed for the adjusted gross income and the salaries & wages data. Like the FIPS import, zip codes with leading zeros were being retrieved with four digits; therefore, the function “converters” had to be utilized to add the zero back to the beginning to facilitate the merge with the election results. The income tax dataset has over 60 columns, so before merging with the Election/FIPS data frame, we copied out the columns zip code, AGI, and salaries. Now, we were able to use the zip code column in the tax data frame to merge with the new election data.

Database Diagram



Final Database



**L**oad

Upon completion of final database build, we executed a pandas to MS Excel export in order to preview the data prior to exporting it to Postgres. With the amount of data exported, the process took nine minutes to create the output. Finally, we injected our merged data frame (2016 Election Results | 2014 Individual Tax Statistics via the FIPS Crosswalk union table) directly into the Postgres database called “etlproject” where it created a table called “electiontaxfips”.

|  |
| --- |
|  |
|  |  |
|  |  |
|  |  |
|  |  |