**Project Report:**

**Statement of project:** Create a database for NJ demographic and crime data by county in preparation for crime analysis.

**Methodology:** The following methodology was adopted:

**1.Extraction:**

**NJ county demographic data** were extracted from the US Census 2017 Profile data: https://api.census.gov/data/2017/acs/acs5/profile?get=

An API key can be obtained from https://api.census.gov/data/key\_signup.html

API data returned as list of lists and converted to pandas dataframe. Please see jupyter notebook in the “codes” folder for the python code.

**NJ county 2017 crime data** were extracted from the FBI’s Crime Data Explorer API: https://crime-data-explorer.fr.cloud.gov/api

An API key can be obtained from https://api.data.gov/signup/

Data collected from contributing Agencies (town, state and county police departments) which each have their own “ori” code. Data collected for two different tables. One for Agencies: their name, ori code, and county locations. The second for crimes recorded by each agency. The ori code keys link the two tables.

Data is returned as nested dictionaries which needs some cleaning up before converting into a dataframe. The function “de\_nest” was built to convert the Agency API response to a dataframe.

The crime API data is paginated which required nested for loops to extract data from each specific “offense” page. The function “unpack” was built to convert the Crime API response to a dataframe. Please see jupyter notebook in the “codes” folder for the python code.

**NJ.gov:** County and zip codes of NJ were scrapped using beautiful soup.

**2.Transformation:**

**Crime data:** The agencies data and FBI crime data as mentioned above were merged using ORI key. Unwanted columns were dropped. Columns were renamed and reorganized. Column with county names were converted to title. The transformed data were saved as tables for subsequent loading.

**Demography data**: This dataset included population, Birth to Unmarried Women, High School Graduation percentage, Unemployment, Below poverty level, Mean Income, Sex-Ratio per hundred women and Median age of people of counties’ in New Jersey.

Data was filtered by .iloc[] function. Then dataset was converted into DataFrame. Columns were renamed, typos corrected, unwanted columns were dropped, headers were rearranged. The transformed data were saved as tables for subsequent loading.

**Zip code county data:** This dataset had information on city, zip codes and county. The zip codes had a ‘0’ that was read as string while reading the csv in jupyter notebook. The transformed file had cleaned zip codes.

Jupyter notebook with codes for transformation can be referred in the “codes” folder. Output tables and csv files can be referred in the “output\_files” folder.

**3.Loading:**

Database was created both on PostgreSQL and MongoDB cluster.

**Add Postgres part…**

To load data, we chose non-relational “NoSQL” database, MongoDB. The **Forrester Wave™: Big Data NoSQL, Q3 2016** identifies a number of reasons why developers and architects are choosing NoSQL over traditional RDBMS, including elastic scalability, extreme read and write performance, flexible data models, lower costs, and more.

To load data, MongoDB cluster was set up on AWS. A user was created and IP address was set up. MongoDB Compass was installed to download, visualize and manipulate data from database. MongoClient was used to communicate with MongoDB using pymongo. Collections were loaded as dictionaries as NoSQL database like MongoDB provides support for JSON-styled, document-oriented storage systems. The database is available in MongoDB as demography\_crime\_db. The collections are “demography”, “crime”, “county”.

Please see jupyter notebook in the “codes” folder for the python code.