***Introduction***

For our capstone project, we are interested in exploring how asthma incidence rates may be impacted by environmental factors, specifically focusing on regional air quality and quantifiable measures of local industry activity. Our goal is to develop a machine learning model that will take in real-time air quality data to predict the number of asthma-induced emergency room visits for any pre-selected county. Additionally, we hope to explore the correlation between various industries and asthma incidence rates by considering both industry size and revenue outputs as measures of industry activity.

To perform this analysis, we have collected three primary data sources: the California Department of Public Health’s Asthma Emergency Department Visit Rates, the US Environmental Protection Agency’s National Air Quality Data API, and the 2012 US Census’ Survey of Business Owners Industry Dataset. We include more details on each individual dataset below, but it is important to note that we were only able to find asthma-incidence data that was aggregated on the annual scale. While we had intended on developing an ML model to make daily predictions of asthma-induced ED visits, we have decided to design our analysis for annual study. Additionally, we will be focusing on the county-level and our ML model will be trained using California asthma-incidence data, but our intention is to implement the model on countries throughout the whole United States.

The cleaning and transformation of this data is crucial in ensuring the accuracy and viability of our results and analysis. This is especially the case as we are not working with ideal data and are treating this exploration as a sort of proof-of-concept. Additionally, to perform our analysis we must connect the three disparate datasets in order to observe and establish the relationships that exist between them. Further, by defining a consistent database structure, we hope to make our data more accessible and usable across a variety of platforms and services.

***Data Sources***

2012 US Census Survey of Business Owners

SB1200CSA05 - Statistics for All U.S. Firms by Industry, Gender, and Receipts Size of Firm for the U.S. and States: 2012. (2015, December 15). Retrieved February 4, 2022, from https://data.census.gov/cedsci/table?q=SB1200CSA05&tid=SBOCS2012.SB1200CSA05.

* Contains information on number of business firms, sales and receipts of business firms, number of employees for business firms, and business firm owner gender demographics
* Data is aggregated by US counties and states and can be filtered to show firm totals or totals by specific industries

***2012 Survey of Business Owners Census Data***

Extract:

Data was collected from <https://data.census.gov/cedsci/> and we specifically extracted the data set titled: SB1200CSA05 - *Statistics for All U.S. Firms by Industry, Gender, and Receipts Size of Firm for the U.S. and States: 2012*. The data was downloaded as a Zip file containing a data CSV, a metadata CSV, and an HTML text file. For our analysis, we have primarily utilized the data CSV but have also referenced the metadata CSV to better understand the information contained within the data columns. After downloading the data we:

1. Uploaded it to our shared project data lake
2. Created a data brick specifically for cleaning/transforming the census data
3. Established a mount point connecting to our data lake
4. Read the data CSV into a spark data frame

Transform:

1. Read data into Pandas data frame using .toPandas()
2. Drop top row to remove duplicate header values
3. Filter data frame to get firm totals only by industry, not demographic breakdowns by sex
   1. df = df[df['SEX'] == '001']
   2. Don’t forget to reset index after this step!
4. Drop columns unnecessary for analysis:
   1. ['GEO\_ID','SEX','SEX\_LABEL','RCPNOPD\_S','FIRMNOPD\_S','PAYANN\_S','EMP\_S','RCPPDEMP\_S','FIRMPDEMP\_S','RCPALL\_S','FIRMALL\_S','NAICS2012','NAICS2012\_F','RCPSZFI\_LABEL','RCPSZFI','PAYANN','index','GEO\_ID\_F']
5. Split [‘NAME’] column into [‘COUNTY’] and [‘STATE’] columns
   1. df.NAME.str.split(‘,’,expand=True)
   2. Drop [‘NAME’] and reorder columns so that [‘COUNTY’] and [‘STATE’] are first columns after index
   3. Order: [['COUNTY','STATE','YEAR','NAICS2012\_LABEL','FIRMALL','RCPALL','FIRMPDEMP','RCPPDEMP','EMP','FIRMNOPD','RCPNOPD']]
6. Drop duplicate and blank rows from data frame
7. Replace ‘S’ fields with value of 0 for columns [‘RCPALL’], [‘RCPPDEMP’], and [‘RCPNOPD’]
8. In [‘EMP’] column, replace non-castable int types to the median value of the respective range
   1. i.e. df['EMP'].replace('100 to 249 employees', '175', inplace=True)
9. Cast [‘FIRMALL’], [‘FIRMPDDEMP’], [‘FIRMNOPD’], [‘RCPALL’], [‘RCPPDEMP’], [‘RCPNOPD’], and [‘EMP’] .astype(‘int’) and then confirm proper data types
10. Replace fields equal to 0 with np.nan
11. Reset index
12. Iterate through [‘COUNTY’] column to remove the last 7 characters from each field (i.e. ‘ County’) so we are left with only the name of each county
13. Drop any extra index columns if necessary

Load:

1. Create new data frame for state information
   1. Contains one column for state name + one column for state abbreviation
   2. Make sure to drop duplicate values if necessary (51x2)
2. Write state data frame to SQL table dbo.State in order to assign unique identifying key
   1. To prevent adding duplicates to SQL database, first load in existing data from dbo.State and read this into a spark data frame
   2. Set this loaded dataframe to equal only [[‘STATE\_ABBR’,’STATE\_NAME’]] columns
   3. Convert/ensure that your state data frame is a spark data frame
   4. Use the .subtract(loaded\_df) function on your original state data frame
   5. Write the result of the previous step into the dbo.State table
3. Read data from SQL dbo.State table back into data brick and create a for loop to iterate through main data frame
   1. Currently, main data frame contains state name information
   2. In loop, match state name values and convert main data frame fields to corresponding STATE\_IDs
4. Repeat steps 1-3 for county information
   1. In step 3 loop, be sure to match both COUNTY\_NAME and STATE\_ID information before converting main data frame county name information to corresponding COUNTY\_IDs
5. Drop the [‘STATE’] column from main data frame
6. We now have a structure that almost matches the dbo.CensusIndustryData table we devised in our SQL database ERD
   1. Including unique COUNTY\_ID foreign keys that connect to both the dbo.County and dbo.State tables (!!)
   2. Convert main data frame into a spark data frame to prepare for SQL loading
7. Write main data frame to the SQL dbo.CensusIndustryData table
8. Congratulations! You’ve successfully extracted, transformed, and loaded the data into the SQL database.

***Conclusion***

To ensure the accuracy and viability of our analysis, we have performed three separate ETL processes to integrate and relate each of our individual datasets. In addition to cleaning and transforming the data, we have also established a relational database structure and have stored the cleaned data for later reference. By doing so, we will be better able to explore the relationships that exist between asthma-incidence rates, regional air quality, and industry activity. Further, storing our data in a normalized SQL database ensures that it will be accessible and consistent across all platforms and services that we end up using for our analysis.