Project Report:

Extract:

Initially our data sets were .csv files that we found off of data.gov. They were data for a "Part 121" Aircraft, which was a classification for aircraft larger than 10 passenger seats. Additionally they were all tracked by year, making this dataset interesting for longitudinal analysis, should we want to examine that further or perform analyses after our ETL process.

Transform:

Our transformation process, while initially simple in concept, was not very simple in execution. We imported the CSV’s into Pandas, however, we realized they were messy and formatted with several rows and columns as titles and headlines for datatypes to be tracked. Referencing the original CSV’s, we deleted the excess rows and columns, and then resaved them as new CSV files. After this, we imported them into Pandas again, and started the deeper cleaning process. We used .dtypes to examine what kind of data we had, and found that one of our columns was an object and not either an Int, or a float. Once we discovered that it was a float we tried to pass in 0 for “No Fatalities” which was the string data that kept the column from also being an Int column. Once we found that, we decided to drop the column as it was duplicate information (fatalities by enplanements) and it was already covered by other duplicate information. Then, we merged everything on “year” which was common among all three of the datasets. Once we did this, we wanted to reorder and rename the columns so we used dataframe notation to pass in a reorganized dataframe, and .rename in order to make the column titles more conducive to Postgres and SQL.

Load:

Loading the data was one of the more difficult tasks, as we attempted to do so the pythonic way as opposed to writing a SQL query to build the tables. We decided to use SQL as all of our data was now structured data, and we wanted to use the most appropriate database. After the cleaning process, we used .iloc with a 0 index, to pull all of the titles of the columns for easier building of our table. We imported all our necessary dependencies and created an engine to get a connection to PgAdmin. From there, we built out a class of Airplanes, and we made a SQL table called “plane\_datass”. From there, we passed in the Column and data type into all of the column names, as copied from Svitlana’s slacked exercise earlier in the week. We then used .to\_SQL and all the necessary information for the documentation, to pass the data into our SQL database. After this, we used a pandas .read\_SQL notation and passed through a general SQL Query to verify the data had made it to SQL. We also double checked by going into PgAdmin, to refresh the tables and see the data within there as well.