#### Avocado ETL Project

#### 

For the purpose if this project, all data sources were extracted as CSV files, manipulated as Python data frames and loaded into a relational database. The following report outlines our ETL process.

#### Extract:

Our data came as CSV files from the following sources:

#### [Volume Data from Hass Avocado Board](https://hassavocadoboard.com/)

* 3 CSV files: 2016, 2017, 2018
* Based on: “incoming volume of avocados arriving into the U.S. Market from all suppliers. Packed and loose volume is combined and reported in total weight; however, imported avocados typically arrive in packed lugs whereas domestic volume is reported in bulk weight before being packed.”

[Avocado Prices from Kaggle](https://www.kaggle.com/neuromusic/avocado-prices)

* 1 CSV file 2015-2018
* Contains weekly retail sales volume information and includes further detail such as avg. price

[USDA Avocado Imports Data: 6 CSVfiles (2009-2018)](https://data.ers.usda.gov/reports.aspx?programArea=fruit&stat_year=2009&top=5&HardCopy=True&RowsPerPage=25&groupName=Noncitrus&commodityName=Avocados&ID=17851#Pb60530d15b234e1eaff8d30d7f226957_4_110)

* Avocados: U.S. imports by value ($1,000)
* Avocados: U.S. imports by volume (1,000 pounds)
* Avocados: U.S. imports unit value ($ per pound)
* Avocados: U.S. exports by value ($1,000)
* Avocados: U.S. exports by volume (1,000 pounds)
* Avocados: U.S. exports unit value ($ per pound)
* By month, contains tables import and export value, volume, and value per pound.

Transform:

In order to gain a better understanding of avocado price fluxuations, we focused on U.S. imports, exports, and price of fresh avocados. Each dataset presented its own sets of challenges and extraneous columns.

The three CSV files with import volume data from the Hass Avocado Board website were already fairly clean datasets with identical column structure. However, we converted each of them to pandas dataframes, concatenated them into a single dataframe, and dropped columns of lesser interest to hold only the total import volume of avocados into the U.S. for each month in 2016, 2017, and 2018.

The Kaggle data was also fairly clean to begin with; therefore, the majority of the cleaning process was done through manipulating the ‘Date’ column using the ‘datetime’ dependency in order to structure the dataframe similar to our other datasets. From there, we rearrange and deleted columns we found unnecessary.

The CSV files we exported from the USDA data tables yielded unconventionally structured information. We loaded each CSV file into a Pandas Dataframe to delete unnecessary fields. The resulting table presented a column for each month and a row for every year, but the table lacked meaningful column names. We set the ‘Market year’ as the index key, combined the three data frames using the index key, and renamed the column names appropriately. As the final step, we reshaped the data frame with melt() function in order to store the data in the same structure as the other tables we explored. Lastly, we used the pd.to\_csv() function to create a new CSV with the cleaned data.

Load:

We decided to load the data into a MySQL database because we wanted to work with a relational database. The CSV files shared common values such as dates and prices, however, we chose to go with the dates (Months and Years) because we wanted to be able to access the data with ease. We felt this would allow users to easily compare information for a given timeframe from the different tables.

We then created a Remote Desktop Connection String with the credential information for the MySQL database ("<username>:<password>@127.0.0.1/ETL\_Project") and used SQLAlchemy to interact between the data frames stored in our Jupyter Notebook and our MySQL Database. With this, we created a local engine and connect to MySQL and Python. With the Data Frame’s stored already in the notebook, with the help of the Panda’s Python Package, the data was easily transferred from CSV to a MySQL DataBase.