### Team Members: Anirban, Anthony, Dean, Delano, Paula

Our group was tasked with selecting 2 or more data sources and performing data cleanup and analysis using ETL methodology, i.e., Extract, Transform, and Load.

Our group began our data search using Kaggle and Data World. After searching data sets/topics, we decided to use fast food data to include restaurant name, address, city, state, and website. We found fast food data on Kaggle, and we decided to use Yelp API. The common variable for both datasets was the zip code. With that in mind, we extracted the Fast Food Datafiniti csv file from Kaggle. Then, API data for fast food locations in Charlotte, Durham, and Raleigh were called from the Yelp API site using Pandas and JSON.

The Fast Food Datrafiniti csv file contained 10,000 rows of data representing all of the US. The Yelp API call resulted in 150 rows of data representing Charlotte, Durham, and Raleigh.

We chose a SQL database using pgAdmin postgres to load our data results. We created a database titled ETL Project and created tables for each dataset titled “fast\_food\_datafiniti” and “fast\_food\_yelp”.

We transformed/cleaned each dataset by creating the tables using “Not Null” for all data fields removing rows of data that weren’t complete. The “fast\_food\_datafiniti” contained 10-character zip codes, so wrote code to trim zip codes longer than 5 characters.

Once the data was transformed, we merged the “fast\_food\_datafiniti” and “fast\_food\_yelp” tables joining on the zip code. Many of the fast food restaurants had the same zip code; therefore, the zip code wasn’t a unique identifier, so the join wasn’t clean. We re-evaluated our data and determined that we needed a zip code data file with the zip code being a unique identifier to create clean joins.

We found a Population by Zip Code dataset from the 2010 census that contained gender and age information by zip code. Using this dataset, we created a SQL table titled “population\_by\_zip” containing 1,622,831 rows of data. Many of the data fields had no values; however, these rows of data weren’t removed because our focus was on the zip code field. Because the data was collected by gender and age, there were multiple rows of data for each zip code. To transform the dataset making the zip code the unique identifier, we aggregated the population data by zip code and dropped gender and age information. After aggregation 33,119 rows of data remained.

The next step was to join the “fast\_food\_datafiniti” table with the “population\_by\_zip” table on the zip code and create a new SQL table titled “fast\_food\_population\_base”. We chose to not filter the “fast\_food\_datafiniti” table by the city (Charlotte, Durham, and Raleigh.) The new table created by the merge contained 9,972 rows of data, 28 fewer rows than the original “fast\_food\_datafiniti” table. The “population\_by\_zip” data was 9 years old and new zip codes were most likely added after the population data was gather.

The final step was to join the “fast\_food\_yelp” table with the “population\_by\_zip” table on the zip code and create a new SQL table titled “fast\_food\_population\_base\_yelp”. This table had the same comparable data fields as the “fast\_food\_population\_base” table; however, it was filtered by city (Charlotte, Durham, and Raleigh.) The new table created by the merge contained 150 rows of data, same as the original “fast\_food\_yelp” table.