**EXTRACT:**

Datasets were taken from NYC Open Data:

* New York City Restaurant Inspection Results : <https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j>
* New York City Food Poisoning Complaint Count :reluctantly settled on <https://data.cityofnewyork.us/Social-Services/food-poisoning/gjkf-etq5>

Originally, my plan was to create a dataframe for New York City’s restaurant inspection results and group the data and the violation count by each restaurant (>20,000 establishments) and scrape the star ratings from YELP using splinter to query each restaurant and scrape the star rating (see [code](https://github.com/elieu17/ELT/blob/master/yelp_scraper.ipynb)). However, due to time constraints, I tabled the code in favor of scraping for the number of hits for each restaurant queried in Google (see [code](https://github.com/elieu17/ELT/blob/master/nyc_rstrnt_violations%20-%20with%20google%20scraper.ipynb) on lines 94, 105, and 106) . However, due to visibility issues with splinter’s *button.click()* function, I realized that this too would prove futile in terms of meeting the deadline. Thus, I reluctantly resorted to cleaning up a second set of data – New York City Food Poisoning.

**TRANSFORM:**

For both CSV files, I used Python’s pandas library of functions to removed the columns I deemed unnecessary. In the end, I boiled down both CSV’s to their zipcodes (operating as the index for both) followed by their respective counts of food inspection violations and food poisoning counts.

**LOAD:**

The finished product, albeit underwhelming in terms of volume, serves to bridge the connection between the number of restaurant food inspection violations and the number of food poisoning cases reported in a given zip code. If given more time, I would have expanded the scope to include more areas around New York as well as historical data.