Team Members:
Jay Sueno | John Bruner | Zhiyi Li

Project Proposal:
Our team will perform ETL on health and fastfood data in the USA. We plan to extract data from Kaggle. Our data will look at the number of fastfood restaurants, obesity, and life expectancy in different cities across the USA. We each will then transform the data to find our desired information. Lastly, we will use a SQL based database to load our separate tables and create a query to join them on 'city'.

Project Repository:
https://github.com/lizhiyidaniel/etl_project_group4

Datasets:

- Health data by cities 2013 - https://www.kaggle.com/noordeen/big-city-health-data
- Fastfood restaurants- https://www.kaggle.com/datafiniti/fast-food-restaurants?select=FastFoodRestaurants.csv

Roles:

- Zhiyi - https://github.com/lizhiyidaniel
    - Repository master
    - Health dataset
    - Create SQL table in Postgres
- John - https://github.com/jmbruner37
    - Health dataset
    - Create SQL table in Postgres
    - Powerpoint presentation
- Jay - https://github.com/jaysueno
    - Fast food dataset
    - Create SQL table in Postgres
    - Coordinate schema and query for SQL database

ETL Summary Report

Extract

We extracted data from Kaggle.com. We chose 2 datasets from different creators. The first is a survey of fast food restaurants across the country to get a sense of how many each city had. This dataset was extracted and transformed by Jay Sueno The second data set was a country wide survey of health issues by city. This data set was extracted and transformed by John Bruner (obesity) and Zhiyi Li (life expectancy).

All datasets were in CSV format.

Transform

Jay

At first I wanted to isolate McDonald's restaurants by city. After exploring the data I found that there were not enough restaurants per city to make it meaningful. Therefore, I switched to cleaning the data to aggregate all the fastfood restaurants by city. I first loaded the csv into pandas using .read_csv(). Then isolated the columns I wanted to manipulate by calling them df['city', 'name'] and renaming the columns with df.column(). I also dropped any empty or null values with .dropna(). I analyzed the data using .count_values() and group_by(). In the end I outputted the desired fastfood dataframe as a csv for record keeping. We then went on to loading.

Zhiyi

I use pandas to create the data frame for the health dataset;clean the data by selecting columns of interest and use value_counts to select 'life expectancy at birth' for each city; split the columns into 2(city and state) for the 'place' column so it could be used to join the restaurant dataset later in SQL. (which uses city as primary key too); city names are also not organized so I have to use value_counts() to see their unique names and rename them. Finally use groupby to calculate the mean life expectancy for each city and make it into a csv file to be added to the database later.

John

- Cleaned Health_Data set with goal of isolating data on obese adults within each city. Using pandas, I isolated a few columns of interest: 'Indicator Category', 'Indicator', 'Place', 'Value'. Then focused on selecting “Percent of Adults who are Obese” within ‘Indicator’ column. Both indicator columns were dropped once isolated adults who are obese. Renamed ‘Place’ to ‘city’ and had to perform a split function with city from state within that column. City names were not uniform so I had to rename a few cities in the same format. Eventually made an average of percent of adults which was under “value” and renamed to “Percent Obese Adults”. Final format was 2 columns: “Percent Obese Adults” and “city”. The average of per city was calculated by a groupby and converted to a csv file to be added to a database where life expectancy and restaurants csv files.

Load

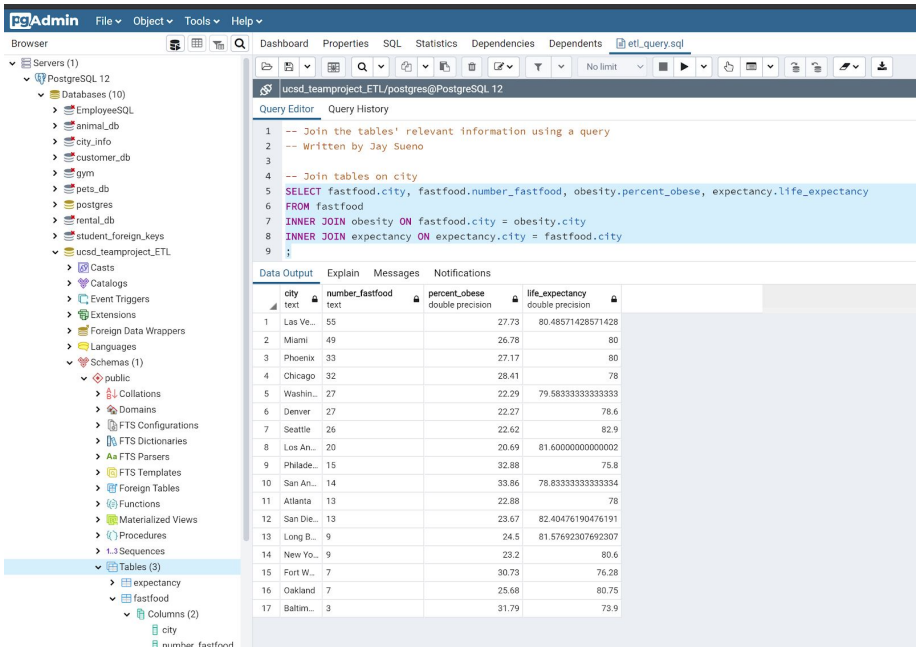
We decided to use a SQL database because we wanted to work on our datasets separately and then join them on a common primary key of city names.

We first created a database in PostgreSQL named “UCDS_teamproject_etl” and created tables using our schema.sql file - “fastfood, obesity, and expectancy”. We made sure that in each of our tables, the PRIMARY KEY was the “city”. The team then provided provided the transformed csvs to Jay. He then created a connection to the postgres server using the python library sqlalchemy’s “create_engine”. From there the dataframes were loaded into the database using the function .to_sql(name=””, con=engine, if_exists=’append’, index=False).

We then created a query.sql file to inspect our tables and to join all 3 of our tables. To inspect we used the query “SELECT * FROM (table name)”. To join the tables we use the query:

```
SELECT fastfood.city, fastfood.number_fastfood, obesity.percent_obese,
expectancy.life_expectancy
FROM fastfood
INNER JOIN obesity ON fastfood.city = obesity.city
INNER JOIN expectancy ON expectancy.city = fastfood.city
```

Our final join looks like this:



	city text	number_fastfood text	percent_obese double precision	life_expectancy double precision
1	Las Ve...	55	27.73	80.48571428571428
2	Miami	49	26.78	80
3	Phoenix	33	27.17	80
4	Chicago	32	28.41	78
5	Washin...	27	22.29	79.58333333333333
6	Denver	27	22.27	78.6
7	Seattle	26	22.62	82.9
8	Los An...	20	20.69	81.60000000000002
9	Philade...	15	32.88	75.8
10	San An...	14	33.86	78.83333333333334
11	Atlanta	13	22.88	78
12	San Die...	13	23.67	82.40476190476191
13	Long B...	9	24.5	81.57692307692307
14	New Yo...	9	23.2	80.6
15	Fort W...	7	30.73	76.28
16	Oakland	7	25.68	80.75
17	Baltim...	3	31.79	73.9