ETL Project

GEORGIA TECH DATA VISUALIZATION BOOT CAMP

GROUP 7 – PROJECT 2

ETL DATA ANALYSIS PROJECT , December 5, 2020

Participants: Carmen Grayson, Michael Klein, Javier Perez, and Robert Varalla

SUMMARY

Will evaluate the correlation of the number of positive Covid-19 test results in each county and the 2020 Presidential Election for the state of Georgia.

DATA SOURCE

> Georgia Secretary of State
>
> Georgia Department of Health
>
> Georgia Income

METHODOLOGY

We will evaluate, load, transform (merge & clean) and upload the given data sources.
The data was acquired on 12/01/2020.

ETL

> The following recounts our step-by-step of importing and cleaning data in Jupyter Notebook, and subsequently exporting the cleaned dataframe to a SQL server, with the ability for the server to be accessed beyond the local machine of the creator.

> Beginning by importing environments for the code. This included the traditional Pandas module, as well as `SQLAlchemy` and a few others. Next, we imported the module `dotenv` and used the `load_dotenv` to access our local environment and get the user and password to log into the Postgres server.
> Following module import, our team read in our data files from the aforementioned sources, using the `os` module as well as the `read_excel` attribute from Pandas.
> The Excel files can be found in the folder Resources.
> Importing the dataframes, we began the data cleaning process. Both of our datasets were complete upon download. No rows were removed, and the datasets contain no null values or rows. Most of the cleaning process involved the election data from this year. Mainly, the process involved the renaming of columns, and reformatting of the frame to simplify the headers.

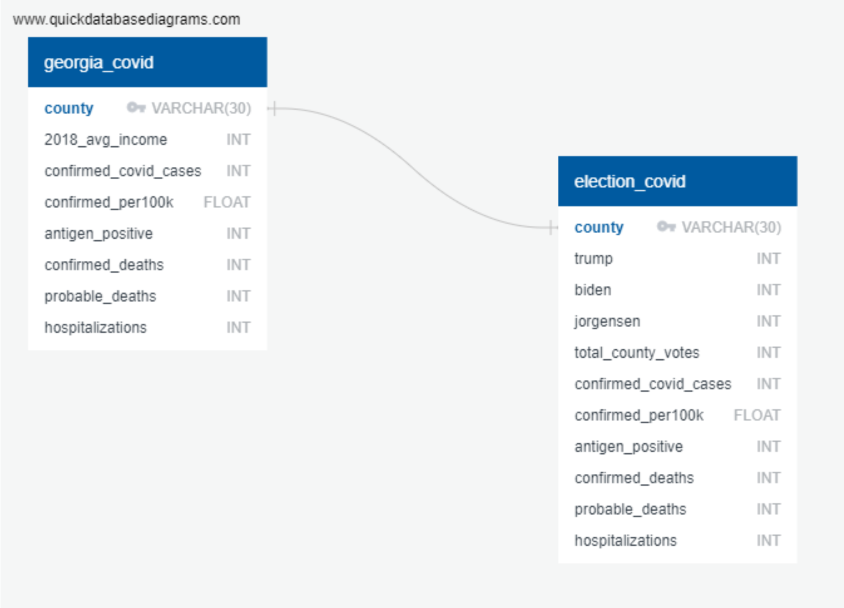
> Formatting column names and headers, the original header row from the imported dataset was removed with `df.drop`, then reset the index of the dataframe to account for the dropped row, and to ensure that the county indices will match our COVID-19 dataset index. The next step was to join together our datasets. We merged the tables on the *County* column, using the election dataframe index. This was an inner join. After the combination of tables, we performed some final data cleaning and made some cosmetic changes for digestibility and clarity. This process included renaming COVID and votes columns to provide clarity between the two values provided their close proximity within the dataframe.

> Following our completion of the data cleaning process, we shifted our focus to uploading the data to a SQL server. This process required some additional quick cleaning, which involved renaming columns to match column names created in our SQL database. This was done to allow for easier column calls within the database, considering using the original names would degrade usability of the database in SQL.
> The queries to create the tables can be found in sql file: etl_project.sql.
> The following process was used to connect, check the connection and upload the data to the database:

1. `create_engine` to create a connection to the Postgres database ***etlproject_db***.
2. `engine.table_names()` to check the the connection and obtain the table names.
3. `sql_df.to_sql` to export the clean dataframe to SQL.
4. `SELECT *` commands in pgAdmin as well as in SQLAlchemy in our notebook. to check the data was successfully uploaded.

For some hypothetical analysis, we subsequently decided to add additional data to our database. We chose to gather some income data for GA. Average income for each county was imported and then combined with the same COVID data frame used previously. We then uploaded this to the database to allow for some additional levels of possible analysis.

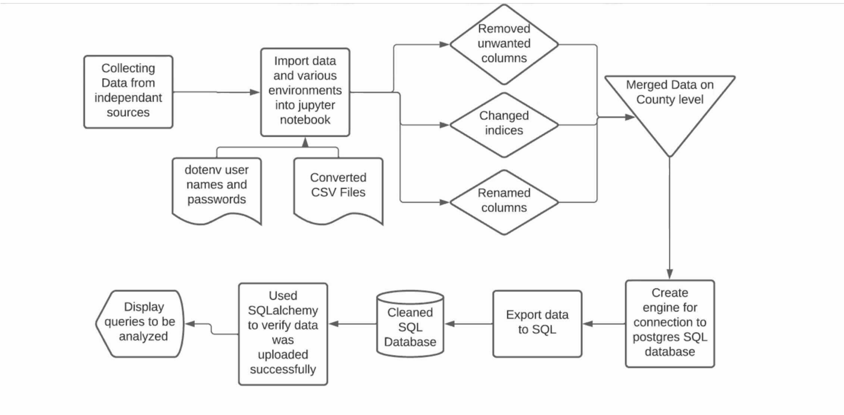
We then updated the ERD to reflect the changes:



georgia_covid	
county	VARCHAR(30)
2018_avg_income	INT
confirmed_covid_cases	INT
confirmed_per100k	FLOAT
antigen_positive	INT
confirmed_deaths	INT
probable_deaths	INT
hospitalizations	INT

election_covid	
county	VARCHAR(30)
trump	INT
biden	INT
jorgensen	INT
total_county_votes	INT
confirmed_covid_cases	INT
confirmed_per100k	FLOAT
antigen_positive	INT
confirmed_deaths	INT
probable_deaths	INT
hospitalizations	INT

The final database was loaded into SQL because it is a relational database software, and below you can find the process flowchart.


Collecting Data from independant sources
Import data and various environments into jupyter notebook
dotenv user names and passwords
Converted CSV Files
Removed unwanted columns
Changed indices
Renamed columns
Merged Data on County level
Create engine for connection to postgres SQL database
Export data to SQL
Cleaned SQL Database
Used SQLalchemy to verify data was uploaded successfully
Display queries to be analyzed