ETL Project

By: Gina Cameras, Abby Ko, Wendy Meyer, Veronica Ratcliff

Date: April 28. 2020

**Introduction**

Our team chose to examine Hate Crime data at a Texas State level and at a local Austin level. The purpose of this project was to Extract raw data and the information required for analysis, Transform it into a format that can serve business needs, and Load it into a production database.

**Extraction**

Two datasets on Hate Crime were chosen; one from the Federal Bureau of Investigation (FBI) Crime Data Explorer website and the other from austintexas.org. The FBI Crime Data provides National data from 1991-2018; while the data.austintexas.org dataset provides local Austin hate crime data for 2017 and 2018. According to the FBI’s Crime Data Explorer, the Hate Crime datasets provide annual statistics on the number of incidents, offenses, victims, offenders, locations that are motivated by the offender’s bias against the victim’s perceived race, gender, religion, disability, ethnicity, or sexual orientation.

The CSV files were loaded in Jupyter Notebook using Pandas dataframes.

Data Sources & Types:

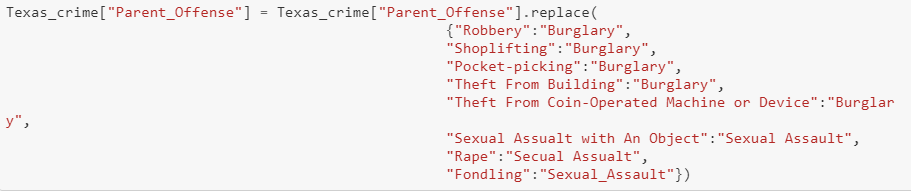
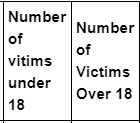
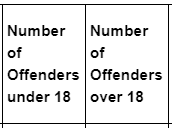
* National - Single CSV File:
  + <https://crime-data-explorer.fr.cloud.gov/downloads-and-docs>
* Austin Local - Two CSV Files:
  + <https://data.austintexas.gov/browse?q=hate%20crime&sortBy=relevance>

**Transformation**

**National Hate Crime**: The CSV file was filtered down to Texas only to compare Austin trends to state level data. We then broke out the offense column to get a simplified parent offense view. After doing that, we simplified parent offense into simpler buckets by aggregating like offenses.

**Austin Hate Crime**: We merged 2017 and 2018 data into one, and simplified the victim and offense age buckets into total count columns. The date columns were reformatted so both datasets shared the same formatting style.

Detailed steps:

* **National Hate Crime** (1 csv file):
  + Selected necessary columns
  + Filtered down to state of Texas only as we compare to Austin file
  + Broke out offense column to parent offense and offense where parent offense if the high level offense (see below screenshot):
    - 
  + From incident\_date used datetime to get day of week, day, and month
  + Re-named columns to lower-case and to align with austin hate crime file
  + Simplified parent\_offense column by combining like offenses (see below screenshot):
    - 
* Austin Hate Crime (2 csv files):
  + Merge 2017 and 2018 data files into one; renamed columns in order to do this seamlessly
  + Adjusted date column for 2018 file to match 2017 file:
    - Removed leading spaces and added slashed between values
    - Removed day of week from value and pushed into its own column
      * Example: ***02 22 2018/Thurs*** → ***02/22/2018*** in date column and ***Thurs*** in Day of week column
  + Merged two victims columns broken out by age range into total victims column
    -  -------> 
  + Merged two offenders columns broken out by age range into total offenders column
    -  -------> 
  + Created separate columns for month, day and year
  + Rename columns to align with national hate crime file.

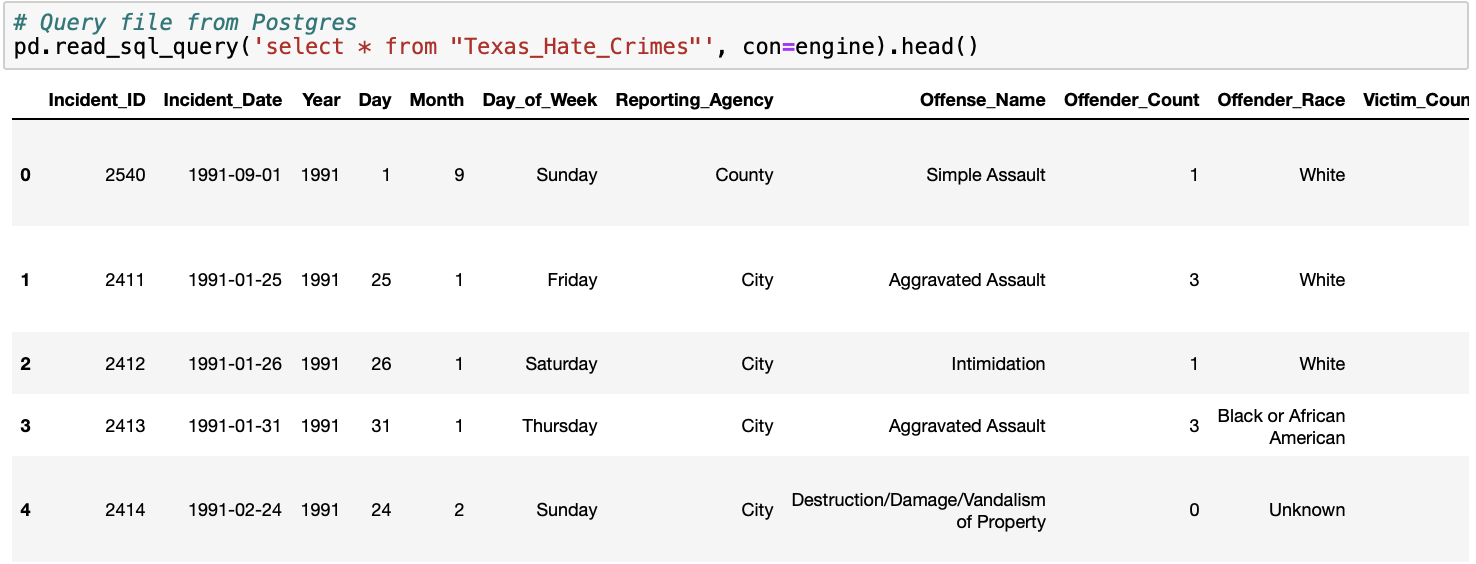
**Loading**

In this section, the focus was to ensure the transformed files were served over to pgAdmin/PostgreSQL. We chose PosgreSQL because we preferred a structured database, given the data was already structured in CSV files. We also wanted to query the data after loading, so PostgreSQl was the preferred database.

* Used SqlAlchemy to create a connection from the jupyter notebook’s dataframe to pgAdmin.
  + Austin\_crime dataframe to Austin\_Hate\_Crimes table in pgAdmin
  + Texas\_crime dataframe to texas\_crime table in pgAdmin
* Used our *queries.sql* file to hold our sql for building the schemas in pgAdmin using the QueryTool
  + Creating the two tables and their respective columns
  + Creating the *hate\_crimes\_db* database in pgAdmin to hold the tables
  + *"Texas\_Hate\_Crimes"* Table contains

"Incident\_ID" INT PRIMARY KEY NOT NULL,

"Incident\_Date" Date NOT NULL,

"Year" INT NOT NULL,

"Day" INT NOT NULL,

"Month" INT NOT NULL,

"Day\_of\_Week" TEXT NOT NULL,

"Reporting\_Agency" VARCHAR NOT NULL,

"Offense\_Name" TEXT NOT NULL,

"Offender\_Count" INT NOT NULL,

"Offender\_Race" VARCHAR,

"Victim\_Count" INT NOT NULL,

"Location" VARCHAR NOT NULL,

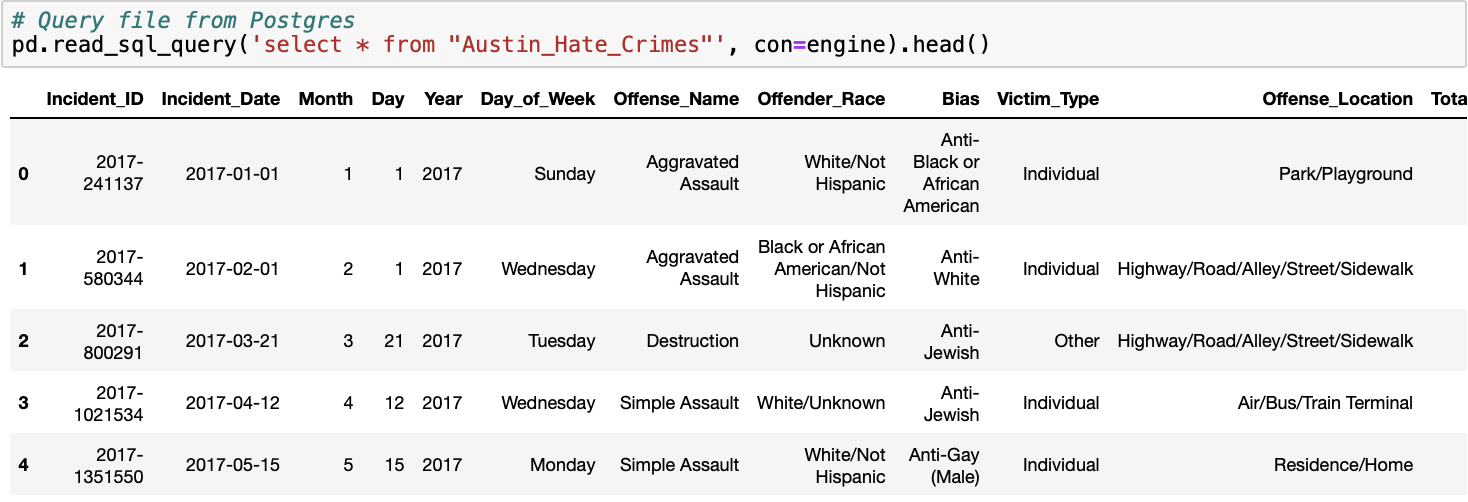
"Bias" VARCHAR NOT NULL,

"Victim\_Type" VARCHAR NOT NULL,

"State" VARCHAR NOT NULL,

"Population\_Group" VARCHAR NOT NULL,

"Parent\_Offense" VARCHAR

* + *"Austin\_Hate\_Crimes"* Table contains

"Incident\_ID" VARCHAR PRIMARY KEY NOT NULL,

"Incident\_Date" DATE NOT NULL,

"Month" INT NOT NULL,

"Day" INT NOT NUll,

"Year" INT NOT NULL,

"Day\_of\_Week" VARCHAR NOT NULL,

"Offense\_Name" VARCHAR NOT NULL,

"Offender\_Race" VARCHAR NOT NULL,

"Bias" VARCHAR NOT NULL,

"Victim\_Type" VARCHAR,

"Offense\_Location" VARCHAR NOT NULL,

"Total\_Victims" INT NOT NULL,

"Total\_Offenders" INT NOT NULL

* Displayed the tables in pgAdmin through the jupyter notebook using sql commands

**Forward-Looking**

These datasets provide hate crime data at a state and local level. The final tables loaded to PostgreSQL can be used to analyze the following:

* Trend of hate crime by day of week to see if crimes occur over the weekend more than weekdays
* Categorize and compare number of offenses by location to understand what location sees the most crime.
* Compare Austin total offenses to Texas total offenses - are the type of crimes similar at the local and state level?
* Look closer at offender race at state and local level - which race committed the most hate crimes?
* Drill in to see what bias showed up the most - compare that YOY and state and local level.
* At the Texas level, there exists a Population\_Group column - analyze to see if crimes occur more in urban areas versus rural areas.

**Tips & Tricks**

* Data collection is important for downstream data cleaning and analysis.
* .rpartition and .lpartition - Proved to be useful as we originally planned to use a split, but found that the partition method was more focused on what we needed to extract from the date column.
* Concat versus merge - We found concat to be useful versus merge because concat allows to stack multiple files with the same column names.
* Overall, studying the data is important and performing data exploration is useful for any ETL process.