NYC EMS Response Times

Group 1: Holly Cornett, Alejandra Gomez, Dominic Marin, Caleb Thornsbury

Project Proposal

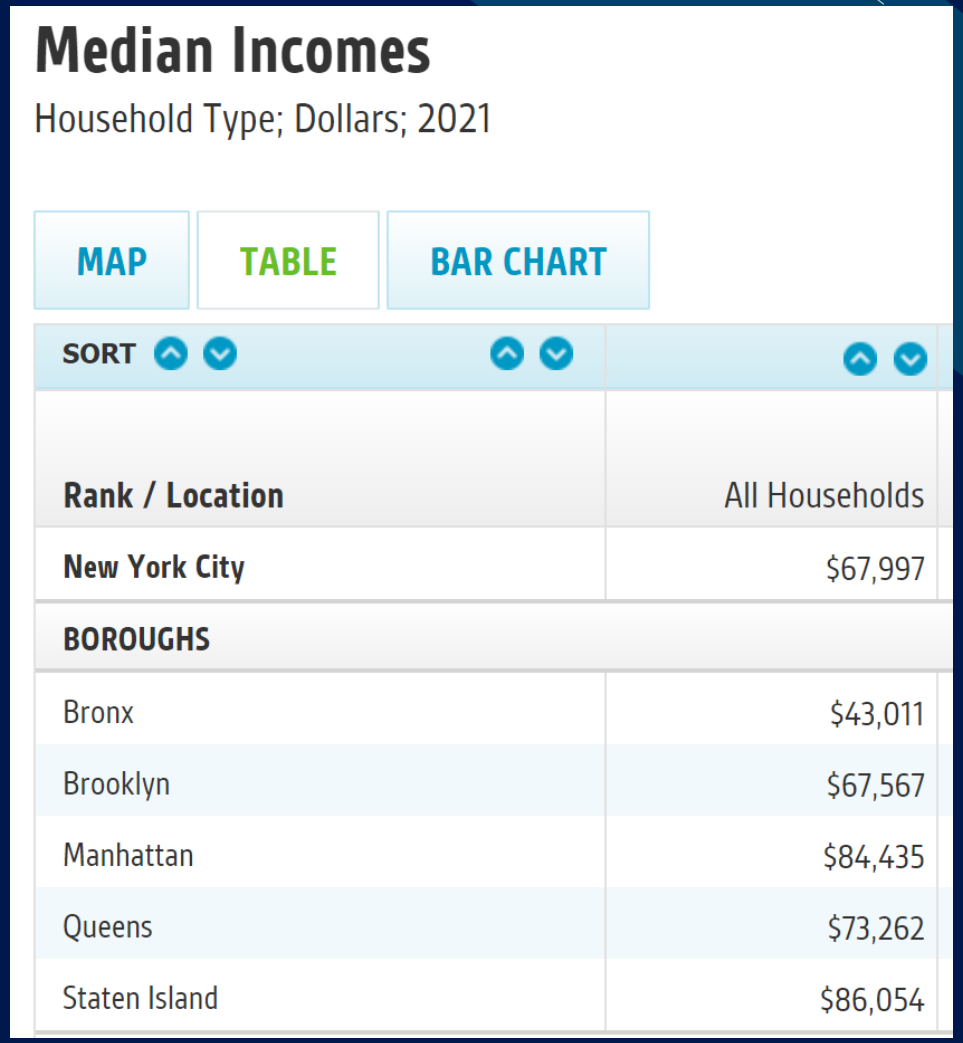
- The aim of our project is to uncover patterns in EMS response times in NYC boroughs.
- We'll examine relationships between response time v. income
- Response time before and during the COVID-19 pandemic (Feb 22, 2020 – April 22, 2020)
- Response time v. initial severity level reported.

Data

- EMS Response Times in NYC dataset
  - Source: kaggle.com
  - Data is generated by the EMS Computer Aided Dispatch System
  - Data spans from the time the incident is created in the system to the time the incident is closed in the system
  - Dataset hosted by the City of New York. The city has an open data platform and they update their information according to the amount of data that is brought in.

Data

- Average Incomes in NYC
  - Source: data.cccnewyork.org
  - Data is pulled from U.S. Census Bureau
  - Median household income
  - Dollar amounts for all years adjusted to constant 2021 dollars using the Consumer Price Index Research Series



Median Incomes

Household Type; Dollars; 2021

MAP | TABLE | BAR CHART

SORT	
Rank / Location	All Households
New York City	$67,997
BOROUGHS	
Bronx	$43,011
Brooklyn	$67,567
Manhattan	$84,435
Queens	$73,262
Staten Island	$86,054

Data Analysis

- Imported data into python
- Cleaned data with Pandas
- Created data frames
- Exported csv files

ems_data.ipynb

C: > Users > agomez1 > Downloads > ems_data.ipynb > import pandas as pd

+ Code + Markdown | Run All | Clear All Outputs | Outline ···

[158]
```
import pandas as pd
```

[159]
```
file_one = "ems-incident-dispatch-data.csv"
```

[160]
```
file_one_df = pd.read_csv(file_one)
```

```
c:\Users\Owner\anaconda3\anaconda\envs\PythonData1\lib\site-packages\IPython\core\interactiveshell.py:3457: DtypeWarning: Columns (7,11,17,21,27,28,29,30) have mixed types
  exec(code_obj, self.user_global_ns, self.user_ns)
```

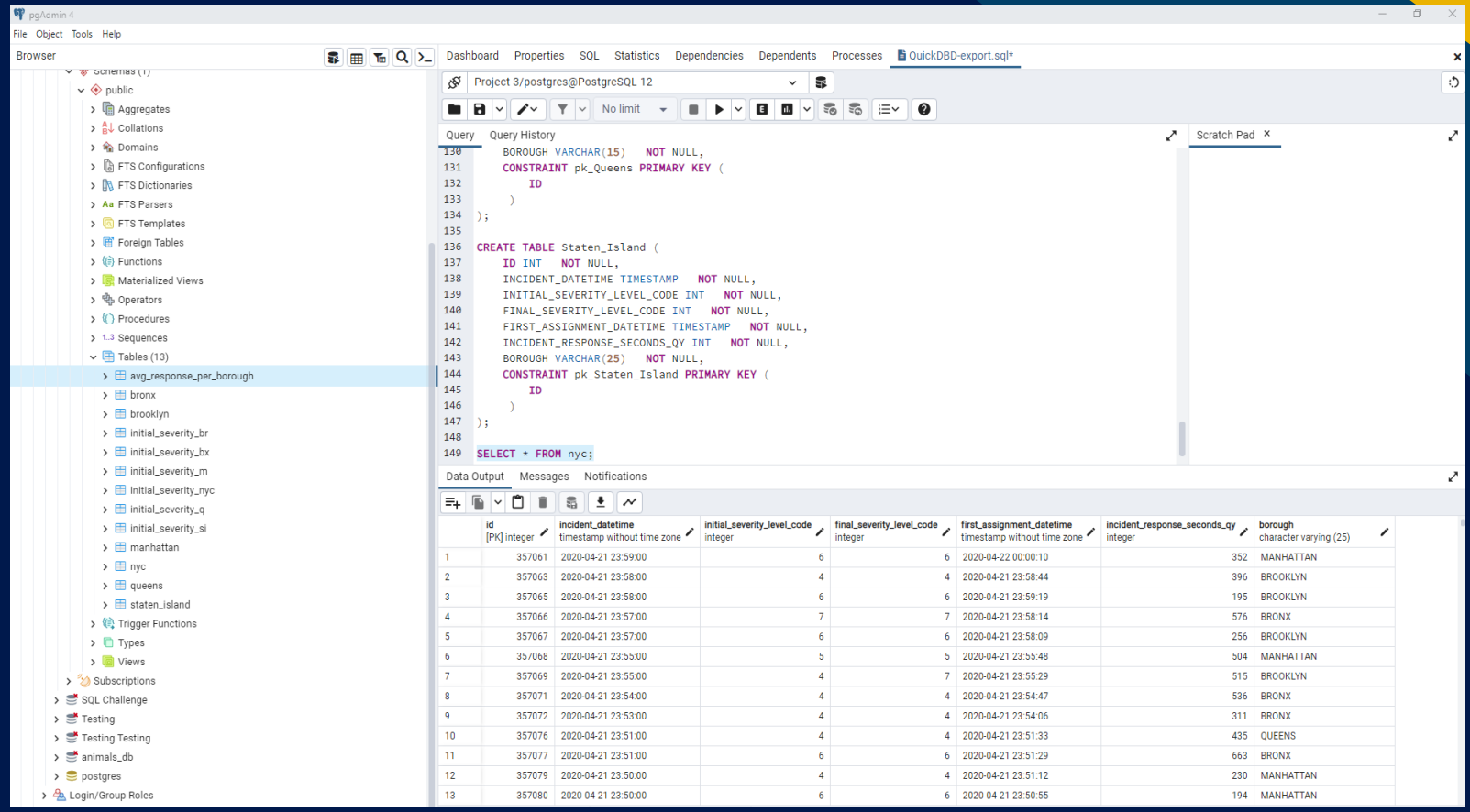
LOADING IN DATA TO CLEAN

[161]
```
file_one_df
```

	CAD_INCIDENT_ID	INCIDENT_DATETIME	INITIAL_CALL_TYPE	INITIAL_SEVERITY_LEVEL_CODE	FINAL_CALL_TYPE	FINAL_SEVERITY_LEVEL_CODE	FIRST_ASSIGNMENT_DATETIME
0	202134231	2020-07-31T23:59:51.000	UNC	2	UNC	2	2020-08-01T00:00:19.000
1	202134230	2020-07-31T23:59:50.000	UNC	2	UNC	2	2020-08-01T00:00:15.000
2	202134229	2020-07-31T23:59:27.000	MVAINJ	4	MVAINJ	4	2020-07-31T23:59:38.000
3	202134228	2020-07-31T23:58:54.000	SICK	6	SICK	6	2020-08-01T00:00:30.000
4	202134226	2020-07-31T23:58:43.000	EDP	7	EDPC	7	2020-07-31T23:59:17.000
...	...	...	...	...	...	...	...
20755328	50010005	2005-01-01T00:00:58.000	INJMIN	7	INJMIN	7	2005-01-01T00:02:34.000

Database

- Used QuickDBD to create the tables and codes
- Inserted ID column to create an index for every table
- Inserted into PG Admin 4
- Imported csv files into each table
- Selected tables to verify information was correct



Flask API

- Created an app.py file to hold our Flask API
- Used SQL Alchemy to create the flask to our Postgres SQL server
- Created different app routes to pull information from selected database tables
- Returned the information as json to use d3 in javascript

```python
# Create our session from Python to the DB.


app = Flask(__name__)

CORS(app)
@app.route("/Manhattan1")
def manhattan1():
    query = "select * from avg_response_after_covid"
    df = pd.read_sql_query(query, engine)
    json_df = df.to_json()

    return json_df

@app.route("/Manhattan2")
def manhattan2():
    query2 = "select * from avg_response_before_covid"
    df2 = pd.read_sql_query(query2, engine)
    json_df2 = df2.to_json()

    return json_df2

@app.route("/Manhattan3")
def manhattan3():
    query3 = "select * from initial_severity_m"
    df3 = pd.read_sql_query(query3, engine)
    json_df3 = df3.to_json()

    return json_df3
@app.route("/Staten Island1")
def staten_island1():
    query = "select * from avg_response_after_covid"
    df = pd.read_sql_query(query, engine)
    json_df = df.to_json()

    return json_df

```

Javascript

- Used the geojson data for NYC from class activity along with json from Flask API to create different plotly charts

```javascript
// Creating the map object


let myMap = L.map("map", {
  center: [40.7128, -74.0059],
  zoom: 11
});

// Adding the tile layer
L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
}).addTo(myMap);

// Use this link to get the GeoJSON data.
let link = "https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-classroom/v1.1/15-Mapping-Web/nyc.geojson";

// The function that will determine the color of a neighborhood based on the borough that it belongs to
function chooseColor(borough) {
  if (borough == "Brooklyn") return "yellow";
  else if (borough == "Bronx") return "red";
  else if (borough == "Manhattan") return "green";
  else if (borough == "Queens") return "orange";
  else if (borough == "Staten Island") return "blue";
  else return "black";
}
function getUrlForBorough1(borough) {
  return `http://127.0.0.1:5000/${borough}1`;
}
function getUrlForBorough2(borough) {
  return `http://127.0.0.1:5000/${borough}2`;
}
function getUrlForBorough3(borough) {
  return `http://127.0.0.1:5000/${borough}3`;
}

//let selector = d3.select('#selDataset').value
//console.log(selector);
function BuildCharts(value){
//let selector = document.getElementById('selDataset').value
let selector = value
// Getting our GeoJSON data
d3.json(link).then(function(data) {
  // Creating a GeoJSON layer with the retrieved data
```

HTML & CSS

```html
<!DOCTYPE html>
<html lang="en-us">
  <head>
    <meta charset="UTF-8">
    <title>NYC Boroughs</title>

    <!-- Leaflet CSS -->
    <link rel="stylesheet" href="https://unpkg.com/leaflet@1.3.3/dist/leaflet.css"
    integrity="sha512-Rksm5RenBEKSKFjgI3a41vrjkw4EVPlJ3+OiI65vTjIdo9brlAacEuKOiQ5OFh7cOI1bkDwLqdLw3Zg0cRJAAQ=="
    crossorigin=""/>

    <!-- Leaflet JavaScript code -->
    <script src="https://unpkg.com/leaflet@1.3.3/dist/leaflet.js"
    integrity="sha512-tAGcCfR4Sc5ZP5ZoVz0quoZDYX5aCtEm/eu1KhSLj2c9eFrylXZknQYmxUssFaVJKvvc0dJQixhGjG2yXWiV9Q=="
    crossorigin=""></script>

    <!-- D3 library -->
    <script src="https://d3js.org/d3.v5.min.js"></script>
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>

    <!-- Our CSS -->
    <link rel="stylesheet" type="text/css" href="./style.css">

  </head>
  <body>
    <h1 align="center">Ambulance Response Time for NYC and its 5 Boroughs</h1>
    <select onchange="BuildCharts(this.value)" id="selDataset">
        <option value="Before_Covid">Before Covid</option>
        <option value="During_Covid">During Covid</option>
    </select>

    <div id="map"></div>
    <h2 align="center">Average Response Time per Borough</h2>
    <div id="plot"></div>
    <h3>Initial Severity Level of Calls in Borough</h3>
    <div id="pie"></div>


    <script type="text/javascript" src="./plots.js"></script>

  </body>
</html>
```

```css
body {
    padding: 0;
    margin: 0;
  }

  #map,
  body,
  html {
    height: 73%;
  }

  #plot,
  body,
  html {
    height: 65%;
  }

  #pie,
  body,
  html {
    height: -10%;
  }
```

Web Page

Thank you.