**Air Quality Comparison**

**San Diego vs. Los Angeles vs. Denver vs. Atlanta vs. Nashville**

**From the coast, to the mountains, to the forest…**

**Methodology**

**Data Collection**

1. Collection of pollen data was achieved using webscraping of [www.pollen.com/research/](http://www.pollen.com/research/). The data scraped encompassed tree, grass and ragweed plants documented to grow in each city. The pollens associated with these are categorized using flowering seasons (Spring, Summer, Fall and Winter), pollen type (Tree, Grass, Ragweed) and allergenicity levels (Mild, Moderate or Severe.) This data was saved as a DataFrame that was exported as a CSV file.

Challenges: A new module, Selenium, was necessary to iterate through the different chart categories (i.e. Season and Pollen Category). This module was installed using the Anaconda platform and documentation of Selenium researched to simulate ‘mouse-click’ behavior within the website.

Tech used: Webscraping, Python, Anaconda platform, Modules: pandas, requests, BeautifulSoup, Selenium

1. Collection of general city information was achieved using webscraping of <https://en.wikipedia.org/wiki/>. The categories gleaned included city name, county, city land area, elevation, population census year, population density, metro population, population rank and climate type. This data was saved as a DataFrame that was exported as a CSV file.

Challenges:

* The wiki data was not consistently tagged or titled (i.e. several states had uniquely titled html tags and some population fields were mislabeled.)
* Climate information was embedded in links in the body of the page text.
* Some cities had multiple climates listed, which were consolidated using the most common wording to optimize comparison.
* Some counties had unnecessary symbols or words.
* UNICODE characters were embedded in the elevation data.
* The population and elevation data included both metric and standard measurements, where only standard measurements were necessary for this project.
* Some elevations contained a range of values instead of a single value.

All issues were handled in the webscraping code using conditional statements, string manipulation (splitting, slicing, concatenation and character replacement), and averages of the range of values. The resulting DataFrame retained only one potential issue of a city having multiple associated counties, which \*\*could\*\* impact its ability to merge efficiently with other tables.

Tech used: Webscaping, Python, Python modules: requests, pandas, numpy, bs4 (BeautifulSoup), and regex

1. Collection of particulate pollution data began with understanding the AQS Data Dictionary, available as a PDF at <https://www.epa.gov/aqs/aqs-data-dictionary>.

I then looked at understanding the measurement parameters for the EPA AQS using the AQS Reference Table, available at <https://aqs.epa.gov/aqsweb/documents/codetables/parameters.html>. I decided to cast a wide net and include parameters I was curious about which were not used in AQI Reports. Of the 1,477 available parameters, I narrowed my list down to 11 by filtering for compounds based on the following criteria:

* 1. Compounds used in the AQI Reports, including ground level ozone (O3), particle pollution (PM10 and PM2.5), carbon monoxide (CO), nitrogen dioxide (NO2), and sulfur dioxide (SO2) which are “the most common ambient air pollutants regulated under the Clean Air Act” (source: <https://www.epa.gov/pmcourse/patient-exposure-and-air-quality-index>).
  2. Other known respiratory irritants such as smoke, carbon dioxide (CO2), nitric oxide (NO) and benzene.
  3. Compounds that were valid according to the ‘Still Valid’ field.
  4. Compounds with valid abbreviations.

In order to follow EPA’s guidelines, time between requests was kept to a minimum of 5 seconds. This prevented any disabling of my account due to violation of their Terms of Service. Data from the Annual Summaries tables was used, which contains *“calculated values of concentrations of monitor samples, which have been summarized for a year, sampling duration, and exceptional data indicator combination. Annual summaries are computed for each calendar year. They may be computed for both sample measurements and NAAQS\_Averages. They may include statistics based on any of the lower level summaries (Daily or Quarterly) or sample measurements. Part of the key is the sample measurement durations summarized (e.g., hourly, daily or NAAQS Average.)”* (source: AQS Data Dictionary [Version 2.28], section 3-22.)

Data was retrieved using nested loops governed by the following flow of iteration:

1. Over three years (2011, 2016 and 2021, covering a 10-year span set at 5-year intervals).
2. Over each state.
3. Over each county for that state.
4. Over each subset of the main list of parameters, maintaining the required max of 5 parameters per request.

The data from each request was added to a common “AQS” DataFrame which contained a total of 56 columns and 1286 rows once all the API requests had been completed.

Exploration of the resulting AQS DataFrame revealed the need for additional manipulations:

1. **Filter for parameters that had data in all four states (which unfortunately elimated measurements of smoke, carbon dioxide, and benzene.) – DO MORE YEARS?????**
2. Relabel both of Atlanta’s counties as a single joint county (i.e. Fulton, DeKalb).
3. Add a column containing the average of the four max values.
4. **Add a field indicating which percentile (e.g. 90%, 75%, etc.) each averaged measurement falls into.**
5. Format the max value date columns to display only the month.
6. Add an associated (main) city column for potential merging with other data tables.

To add an additional layer of relevance to the AQS DataFrame, I wanted to add AQI Categories (Good, Unhealthy, Hazardous, etc.) to each of the measurements to calculate the frequency of each category for each city. This was accomplished by converting the AQI Breakpoints dataset from its CSV format to a pandas DataFrame. I then performed an Inner Join, keeping only records that occur in both the AQS DataFrame and the AQI Breakpoints DataFrame. This merged dataset was saved as a new dataframe to preserve the originals and avoid losing data for the parameter (namely, Nitric oxide) and measurement durations that were unique to the AQS table. The AQS and AQI DataFrames were joined on five common fields, one of which was joined based on the measurement value from the AQS DataFrame falling between a range of two values (the low and high breakpoint) in the AQI Breakpoints DataFrame. Due to the complication of joining two tables based on a range of values, the Inner Join was performed using the *pandasql* module to take advantage of the more straightforward functionality of SQL.

Complications: Some cities did not have data for all parameters for all years. This was handled by using the average of the available years’ data..

1. Asd

**Data Anaysis**

1. First, I looked at a breakdown of pollen species per season for each city.

Conclusions:

1. Then I looked at the breakdown of pollen types for each city. The categories of pollens are trees, grasses and ragweeds.

Conclusions: Nashville has the highest number of distinct tree pollens and the second greatest number of distinct grass pollens.