**Steps to Extract and Transform the Home Price Data:**

**Extract**

* The first step is to read in the original CSV file. This data set came from kaggle, and is titled: Indiana - Property Sales Disclosure. The link is: https://www.kaggle.com/shoreviewanalytics/indiana-property-sales-disclosure. This file is 2GB!
* The file was downloaded directly from the website.

**Transform**

* This dataset contains 87 columns, including data on the buyer, seller, sales price, lot description, tax information, and sales date.
* The following columns were selected: Parcel1\_Acreage, PropStreet, PropCity, PropState, PropZip, C6\_Sales\_price\_Assessor, Buyer1Street, Buyer1City, Buyer1State, Buyer1ZIP, Conveyance\_Date
* A .dropna() was performed on the data to drop any rows that contain an NaN.
* There were many inconsistencies in the Property City column, including misspellings and differences in the cases of the letters.
* We obtained weather data for 107 Indiana cities. Only data for those cities was included in the final home price dataset. To do this, the city list was copied over from the weather jupyter notebook. Then the cities needed to be converted to upper case letters to match up with the city entries in the home price dataset. A for loop was setup to include only those 107 cities.
* Columns were renamed to be more descriptive. New column names: Acreage, Property Street, Property City, Property State, Property Zip, Sale Price, Buyer Street, Buyer City, Buyer State, Buyer Zip, Conveyance Date.
* Duplicate data was dropeed using: .drop\_duplicates()
* Finally, the cleaned file was saved as a csv.

**Steps to Extract and Transform the Weather Data:**

**Extract**

* Our original plan was to get data for the specific location and day of each home sale using an API call from Open Weather Map. Historical data was unavailable without a paid subscription. This was true of many other weather websites as well.
* Weather data was scraped from the following website: <https://www.usclimatedata.com/climate/indiana/united-states/3184>
* Chromedriver was used to collect html data from the website and BeautifulSoup was used to extract parts of the data that we needed.
* The first piece of information that was needed was the url endpoint for each city to obtain weather data from the tables. This was done by scraping the endpoints.
* The following code was used to obtain data from the tables of average weather data. An empty dataframe titled average\_monthly was created to hold the scraped data. A for loop was created to cycle through the urls and the pandas function .read\_html() was used to collect the data from the tables on the page. A column was added to include the name of the city. This was done by splitting the url string using the pandas function .split(“/”), and then indexing the component containing the name of the city.

average\_monthly = pd.DataFrame()

for url in url\_list:

main\_url = "https://www.usclimatedata.com" + url

tables = pd.read\_html(main\_url)

df\_1 = tables[0]

df\_2 = tables[1]

df\_3 = pd.merge(df\_1,df\_2, on = "Unnamed: 0",how="inner")

url\_split = url.split("/")

df\_3["City"] = url\_split[2].upper()

average\_monthly = average\_monthly.append(df\_3)

**Transform**

* The columns were transposed and then the files merged back together so that the averaged weather data are columns.
* A Couny column was added so that the census data and the weather table can be joined. The home price data can be joined via the city name.
* Please note that in this dataframe some “NaN” values were intentionally left in. Data is present for all cities for the average high and low temperatures, but snowfall data is only available for 64 of the 107 cities.

**NOTES**

* The final csv file for the home price data: home\_price\_final.csv
* The final csv file for the weather data: average\_indiana\_weather.csv
* All work to extract and transform the home price data is in: Home\_Price\_Data.ipynb
* All work to extract and transform the weather data is in: weather\_data.ipynb