**Extract**

We worked with three total datasets which comprised four tables. Our first data set was a sqlite database that contained two tables with information on Super Mario Maker custom level data. Our second data set was a csv that contained information on the populations for multiple countries, which we knew we could compare with the Super Mario Maker country data. Our third dataset was a json file that contained translations for the two-letter abbreviations of country names, which we planned to use to translate the abbreviated country format of the Super Mario Maker data to the full-length name format of our country population dataset.

**Transform**

To begin getting our data ready for analysis, we began work on combining both of the tables in the Super Mario Maker dataset. To do this we started by converting the sqlite database into pandas dataframes through a jupyter notebook - after that was done, we were able to drop the columns that we didn’t plan on using in our final dataframe, such as the names of individual levels and authors. After we finished editing both the tables, we were able to merge them on a shared “author\_id” column.

Additionally, since we decided to get rid of level and author names in order to maintain dataframe readability, we wanted to remove duplicate country listings while keeping the previous numeric values while also adding the amount of levels each country had – in order to do this, we began by creating a list of level counts and country abbreviations that lines up with one another. After creating our lists, we moved on to aggregating the data in the “country” column to take the sum of the number of plays, medals, and likes – we then merged this data with a dataframe we created using the two lists created earlier; we were now done working with the Super Mario Maker dataset.

Once we had the Super Mario Maker dataset cleaned and ready to work with, we imported our json country abbreviation translation file into our jupyter notebook as a pandas dataframe. In order to merge the abbreviation translations with our Super Mario Maker dataframe, we had to rename the columns containing the country abbreviations to match each other; after merging, we were done working with the json file (for now.)

At this point, we were ready to pull in our country population csv file, which like the others, we converted into a pandas dataframe through our jupyter notebook. After dropping the columns we didn’t need, we renamed the columns to match our work-in-progress dataframe and proceeded to merge them together; however this caused us to lose our country abbreviations column, which we still wanted – this is why we decided to work with our json file again.

Bringing our json file back in, we converted it into a pandas dataframe and renamed the columns as we did earlier, then simply merged that dataframe back into our work-in-progress dataframe. At this point, our dataframe-in-progress was looking pretty good, however we now had duplicate country information – since we believed this was due to how the country csv file was structured, we decided to drop the duplicates that contained the lowest population count, as those were likely older entries, and we wanted the most recent data available to us. In order to do this , we began by sorting our data by the lowest-to-highest population, and then removed duplicates while keeping the last entry. After re-organizing the dataframe to be organized by their index values, our dataframe was complete.

**Load**

Now that our dataframe was complete, we opened MySQL Workbench and created a database called “mario\_db,” which we would use to store the table containing our complete dataframe’s information. Using SQLAlchemy, we were able to very easily transfer this data into our MySQL database as a table named “mario\_table;” this marked the end of our ETL process.