This report outlines data wrangling steps taken in the Wrangle and Analyze Data Project in Data Analyst Nanodegree at Udacity.

1. Changed the ‘id’ column in df_json dataframe to ‘tweet_id’ in order to match it with other dataframes. That will make is easier to merge dataframes at the last steps. In addition, I changed the data type from int to string because ids should not be numeric, and they aren't intended to perform calculations.
2. Changed ‘timestamp’ column data type in df_archive dataframe. This step will help if I was to analyze the data based on time or date.
3. As per project instructions, only original tweets must be included in the final master dataset. I deleted the rows that contain retweets. After deleting the rows, I dropped the columns that contained information about these tweets because all the values were null.
4. Next, the ‘expanded_urls’ was the only column that had a few missing values. I could not determine why that was the case nor I could fill these values out; therefore, I dropped these values.
5. This analysis should only incorporate tweets of dogs; therefore, the next step was to delete rows that contained non-dog tweets. In order to do that, I first merged the df_pred dataframe with df_archive and created a new dataframe df_archive2. The df_pred contains 3 predictions from the neural network whether a picture shows a dog. I assumed that if any of the predictions suggest that it is a dog, I treated it as a dog. I checked a few tweets ‘by hand’ and it appeared to be true.
6. Cleaned ‘rating_denominator’ and ‘rating_nominator’ in df_archive. I changed the maximum display column width to 200 characters to be able to read the whole text values. Then, I extracted the values from text using regular expression, deleted tweets without ratings and tweets with multiple dogs. I also changed the data type to float because some values have decimal points and can be used in performing calculations.
7. Next, I cleaned names using regular expression in df_archive. A useful pattern to find these "fake names" is that they started with lowercase, instead, the "real names" always started with uppercase, moreover, the real ones usually are placed after words like "named", "is" etc...so it'is possible to clean this issue using a regex.
8. Combined 'doggo' 'floofer' 'pupper' 'puppo' in df_archive into one column. First, I created a new ‘stage’ column by adding the aforementioned columns. The new column contained 8 combinations of long string values. Next, I renamed the values based the long strings. For example, I changed 'NoneNonepupperNone' to ‘pupper’. If there was more than on stage, I called it ‘multiple’.
9. Finally, I used the left merge on ‘tweet_id’ columns to incorporate the df_json dataframe. I checked if there are no missing values after the merge and then I saved the dataframe into a csv file.