GROUP 4
ETL Project

Adam Van Geleuken
Andriani Christanty
Roxana Ugaz

For this ETL Project, our group created a detailed database of Pokemons featured in the Pokemon Go game. Hopefully users of Pokemon Go could benefit from accessing this comprehensive database, and use the information in it to their advantage. For this group project, we loaded our datasets into Jupyter Notebook and worked with these files using Python programming language with the Pandas dependency.

We extracted one of our datasets as a CSV from the website *Kaggle* (https://www.kaggle.com/rounakbanik/pokemon/version/1). The author of this repository provided this dataset consisting of detailed data about Pokemon : 801 rows and 41 columns. This consisted of details ranging from each pokemon's attack effectiveness against various types of Pokemon, to their Japanese names and spawn rate.

In order to view and analyse the data from this CSV properly, since it was quite large, we changed the settings of our Jupyter Notebook to enable us to view all the columns, as the default setting would not let us view all of the available columns (function : pd.set_option('display.max_columns', None).

Our second data set was a CSV obtained from *data.world* (https://data.world/ljvmiranda921/pokemon-go-dataset/workspace/file?filename=pkmn-go.csv). This was the data of the Pokemon in the Pokemon Go game. This dataset contained 11 columns and 146 rows, mainly the data only relevant to the game. We combined this dataset with the first one, using the *right join* method (Pokemon Go file as the second file). We then used this CSV as the basis to find which Pokemon we would keep information for. We used the column "name" as the basis of the joins as that column connected the two datasets and performed further data transformations on these data.

Our first step when looking at the combined dataset was to drop columns that were not relevant for this project. We also set the column containing names of the Pokemon Go Pokemon as the first column, to enhance the readability of the document. We then looked at the info (df.info()) of the dataframe to view the non-null values in each column. From there, we could see that there were some rows containing some null values.

With those null values, we have decided that some of them are relevant. For example, for null values in the column Type 2 - some Pokemon do not have Type 2. Our group decided to keep these data and change the “NaN” into “None” to enhance the readability of the database. We got rid of some rows that were completely populated by “NaN” since those had no use for our database. We ended up with a table consisting of 144 rows and 44 columns.

Our group decided to load the final database (Pokemon2_db) into a non-relational database, that is MongoDB. As Pokemon and their information were object-oriented, we felt that it was appropriate to do so. Furthermore our target audience would be Pokemon Go users who may not have extensive programming experience, and would likely benefit from being able to access all the data at once. We have also outputted the file in a CSV format to address our assumed target audience. If needed, though, anyone accessing the database would be able to form relational tables, based on this newly-developed database. We provided a document (DOC_ID) of our table and the type of variables each column had.

We hope that the updated Pokemon Go database might be of value to the community, and any further work in this topic will be interesting to see.