horizontal line

**Abhishek Bhatt**

ETL Project

**16 November 2019**

# OVERVIEW

\*E\*xtract: your original data sources and how the data was formatted (CSV, JSON, pgAdmin4, postgreSQL, etc).

\*T\*ransform: what data cleaning or transformation was required.

\*L\*oad: the final database, tables/collections, and why this was chosen.

# Extract

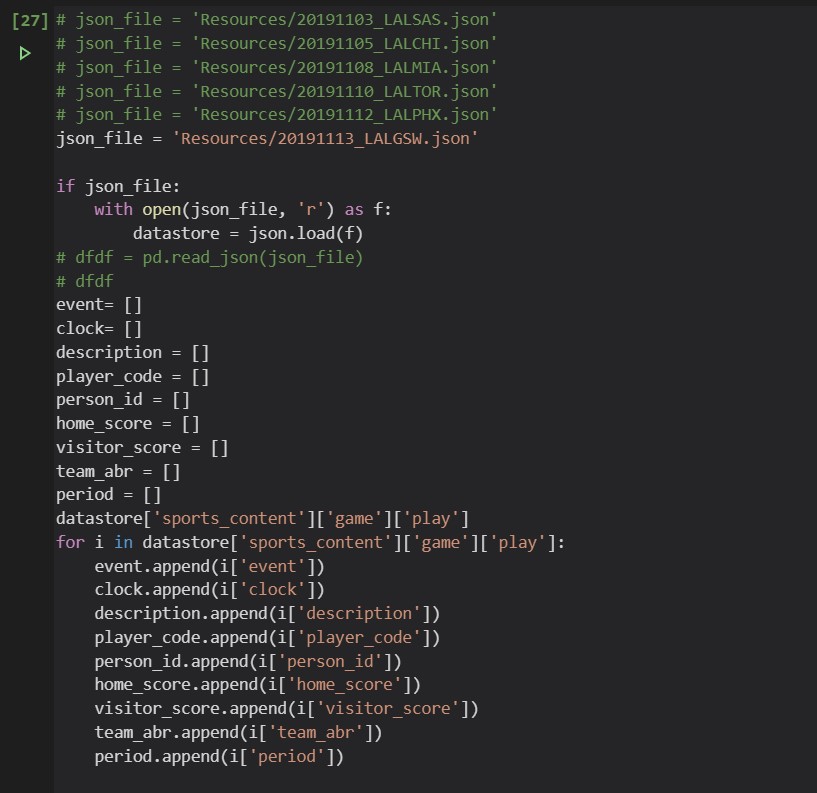
JSON data was found at an NBA stats database (data.nba.net). This database contained detailed play-by-play information for each NBA game. CSV data for individual players and individual team was found at FiveThirtyEight (github.com/fivethirtyeight). The CSV data contained aggregated data point for individual players and teams over an entire season. Extracts for several individual games will be taken from the JSON location and single extract will be made from the CSV data as those cover an entire season.

# Transform

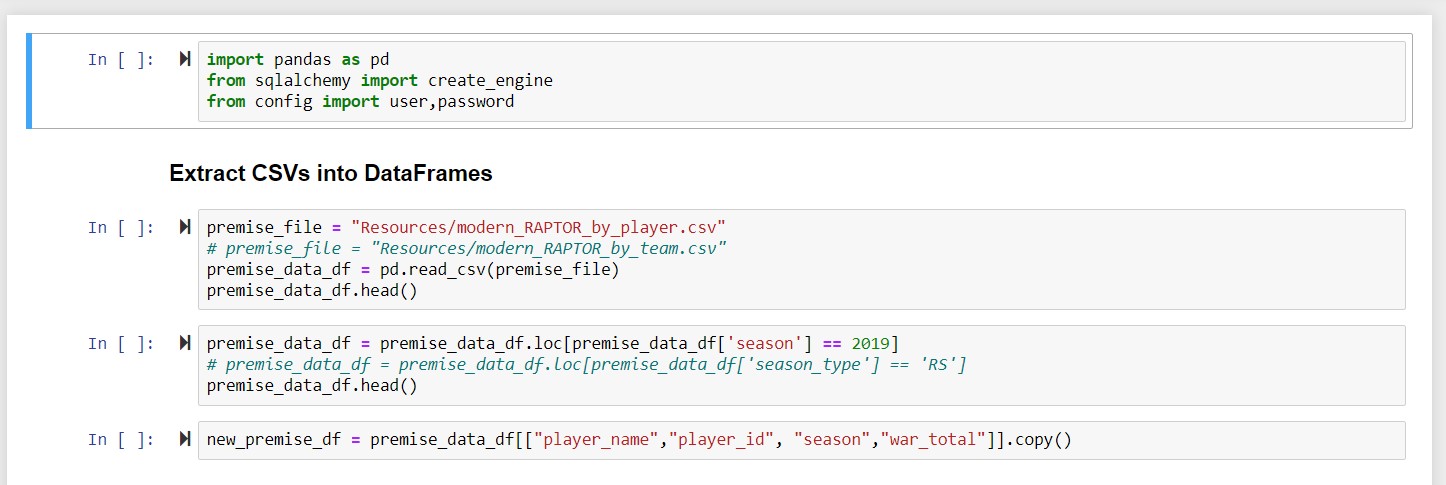
As is very common from raw data, we did not need all the information contained within each data source. For example, The JSON play-by-play data needed to be drilled down to get the necessary information.

# 

The play-by-play data was nested within a set of dictionaries found at sports\_content->game->play. To access this portion of JSON, we used the following code.



Json\_file stores the file into a directory. With open loads the json into a readable object. We create empty lists in which to load the data. Once we have the empty lists, we run a for-loop to load the contents of each section of game data at their respective location. This loop runs through each item at the sports\_content->game->play location.



Loading the CSV into a DataFrame is a bit easier. We are able to load the CSV into a DataFrame directly. Once we have it in the DataFrame, we are able to isolate the columns we want; in this instance, we want player\_name, playerID, season, and war\_total.

# Load





Loading the two Data Frames is very similar. We set the username and password into an object. We create the engine. Then we confirm the tables in the database. Last, we load the DataFrame into SQL with its respective table name. We chose to load individual game data so we can look over how players have started this season and compare it to how they performed over the course of last season. Loading game data into individual tables allows for individual events to be selected more easily. Loading aggregated player and team info into their own respective tables keeps aggregated data away from individual game data.

## Code

Code can be found at the following location: