For my project I wanted to find similarities in Spotify songs/artists across regions to see if there was any correlation between region and song attributes. I did this by joining a dataset from Kaggle that included the top 100 most streamed songs on Spotify, with API calls through MusicBrainz that gave me information about a given artist. I went through the artists in the top 100 songs and made requests for each of their data, then combing it into one large dataset with Pandas. After that, I analyzed correlations between an artist's country and the characteristics of their songs, including metrics like beats per minute, valance, and length.

The main difficulty I encountered in this project was finding a dataset and API endpoint that matched together which I could use to explore my goal of region and song attributes. Originally, I was going to explore my own song characteristics by joining a large dataset of over one million Spotify songs with the Spotify API that could query my own profile. Unfortunately, although one million is a lot, the songs I listen to weren't listed in the dataset so I had to switch routes. I then found my final Kaggle dataset and figured out the direction I wanted to go in, but the Spotify endpoint didn't provide me the artists's region, which was the main information I was interested in analyzing. So I had to find a different music API, which led me to MusicBrainz, which had the information I wanted but still had some other issues. It was fairly well documented, but had lots of strange behavior. The first I encountered was authentication but not through an account but rather a project name and email, which was not validated at all. The next issue was query frequency blocking, in which repeated queries without a pause would lead to invalid response. I fixed this by using query throttling, which is just controlling the pace of my queries through Python's time library. Now that my queries were working, I had more issues with the data itself. Although MusicBrainz has information for the majority of artists, it still lacks the field I want for some artists like Calvin Harris and Harry Styles, but there isn't a great fix to this issue besides fully changing my endpoint when MusicBrainz does a great job for the majority of my data, so I didn't. Going back to the data, I also used a different field in MusicBrainz for my first iteration of the project, which resulted in issues like the United Kingdom being marked separately from Scotland, but that was sorted out by adjusting to a different attribute of "Country" instead of "Area". Figuring out API issues, as well as just finding an API that had the data I wanted, was where the majority of my time went into for this project,

Overall, I didn't find the project too difficult once I had my datasets and addressed the problems above. Pandas makes merging datasets and converting types very simple, so I just had to go through the required processes to finish the project. Pandas also makes analysis relatively simple, using methods like "describe()" to already format and examine the data.

As for the end product ETL (extract, transform, load) data processor, it is a great utility to be able to make and use in data science. They allow you to create a pipeline for processing data that is relatively simple and very effective, allowing you to get beyond data management and into the analysis. The majority of time in Data Science is spent processing data, so being able to quickly get through this step and be closer to the end goal is very valuable. Having an example ETL pipeline in my toolbox will be incredibly helpful for future projects because I can just reuse the overall format and just change what columns are used depending on what datasets I use.