Data Project #1 One Pager

Claire Lee and Samyu Krishnasamy

The biggest challenge in building this ETL pipeline was considering all necessary data sources (local files, remote URLs, and APIs) within a single function. To approach this challenge, we made sure that the code could correctly detect the sources of data (API, URL, or local file) and execute the appropriate ingestion method for each. Additionally, we also needed help managing different data formats (CSV, JSON, and SQL) and having to convert between them. To tackle this, we had to gain more knowledge about data structures and file I/O functions.

Another challenge that we faced was error handling and making sure the project produced informative error messages in case of failures. For example, we had to implement error messages for network issues during API calls or file format mismatches. It was important to handle these edge cases to ensure that the pipeline would not crash and that we would be able to find the error quickly with helpful feedback.

Working with CSV and JSON data turned out to be easier than anticipated, thanks to Python's built-in libraries such as CSV and JSON, which simplified the process of reading, writing, and modifying these formats. Libraries like SQLite3 also made it easy to write data into a SQL database. Once we set up a structure for handling each data type, switching between these formats became a lot more manageable and easy to understand.

An unexpected difficulty was managing the transformation between different data formats in a generalizable way. For example, mapping JSON data into a tabular format like CSV or SQL was more complex than JSON data can be deeply nested or irregular. Ensuring that the transformation was done in a lean, reliable way that did not lose any important data was more complex than expected. Furthermore, we ran into a lot of issues tackling API when a JSON response from an API did not match the expected structures. This required more time than anticipated as we had to handle edge cases like empty data, missing columns, or unexpected file formats.

This utility could be useful for other data projects as it is versatile and able to be used in many other data science projects. It allows for the integration of multiple data sources which is essential for future projects that involve working with data that comes from various sources. The utility allows for a unified format for the analysis of these different sources. Having the ability to add or remove columns during data transformation also makes it easier to clean and preprocess data before using it in machine learning models or analytics. Overall, this tool could serve as a foundational utility in any data pipeline that requires frequent data updates or transformations from multiple sources. It is also scalable so new data formats or APIs can be integrated by adding functions for the new sources.