ETL Project

Presented by Brett Chadwick, Kenn Knisley, Mike Kozar

The goal of this homework is to describe the following:

* **E**xtract: show our original data and how it was formatted.
* **T**ransform: what we did to clean up and present the data.
* **L**oad: show the final databases and why we chose this.

This project consisted of two sources of data. One was called by API while the other was a CSV. Each one was extracted, transformed, and loaded separately to show how each step was done. We wanted to create two sets of data that held information on the Dow Jones 30 stocks. This information could be used for anything from research on past performance to projections of future performance. For Extract and Transform, there are two different ways we went through the information, but the Load was the same for both to allow us to use a single database.

The Extract and Transform of the API was completed by the following:

1. Extract:
   1. A base URL was given from FinancialModelingPrep.com and a list was created to show each ticker for all the Dow Jones 30 stocks.
   2. Next we created empty lists for the storage of all the data collected from the API that would be used to make the DataFrame.
   3. Once those lists were created, a for loop was set up to go through the site and collect the information for each of those lists and place them within the lists.
2. Transform:
   1. The information that was taken out of the API call was then placed inside a Pandas DataFrame for ease of use for the end user.
   2. This DataFrame was then organized by columns as well as received update column

names.

* 1. The DataFrame was then transferred to a CSV in order preserve the data and to prevent pulling from the API in the future.

The Extract and Transform of the CSV was completed by the following:

1. Extract:
   1. The CSV was extracted from the GitHub file “RawStock”
   2. The use of OS was done to allow us to pull the file for each different stock.
   3. We then stored those names into a list of stocks using the stocks\_file\_name.
   4. A for loop was written to pull each individual file out of stocks\_file\_name and turn the data into a DataFrame.
2. Transform:
   1. The first thing done was to clean up the tickers and remove extra text from the file names.
   2. The next step was to create a dictionary where we took the ticker symbol and matched them up to the individual company name associated with that ticker. A for loop was run to match them up based on the list in alphabetical order by the ticker symbol.
   3. The DataFrames were concatenated to combine the 29 individual DataFrames into one large DataFrame.
   4. After this, we choose to update which columns we wanted to see in the data set: Ticker, Date, Open, and Close.
   5. To create information not presented in the CSV, we calculated the Daily Percent Change of each stock and added that as a new column called Daily Percent.
   6. The next step merged the dictionary created to add the company name with the DataFrame to give us the ticker and company name as columns.
   7. The DataFrame was then transferred to a CSV in order to make sure all the data was saved to a single location, rather than having to pull both sets (dictionary and CSV) in the future.

The last part of the project was to Load, which was the same for both sets of data.

1. Load:
   1. Connected to local MongoDB through PyMongo, created a new database name as well as collection names.
   2. The database was named FinanceDB, and a collection for the stock data was named Stock\_Data. Another collection for the annual reports was created and was named Annual\_Reports.
   3. Used pandas to read in the outputted csv files, converted those files to JSON format and then exported the JSON information to the MongoDB database.
   4. Mongo was chosen for two main reasons; 1. The data is stored in the form of JSON style documents. If you ever wanted to create an API, this would already have the information in a style that could easily be used for that. 2. The performance of MongoDB is higher than compared to relational databases.

The greatest challenge that this project ended up presenting our team was to make sure we pulled relevant information without making the data set so large that it was difficult to find information in it. After the Transform, we believe the information is organized and easy to find if ever needed.

We have also included both CSV files in our data on GitHub to allow others to view the data if they would like.