ETL (Extract, Transform, Load) Pipeline Documentation

December 4, 2023

Overview

This documentation outlines the ETL pipeline designed to handle data from the Federal Reserve Economic Data (FRED). The data focuses on outstanding commercial real estate loans held by all commercial banks, measured in billions of dollars across various dates. The pipeline is implemented in Python using a Google Colab notebook. Afterwards, the transformed data will be loaded into Google BigQuery for further analysis.

Prerequisites

Before running the ETL pipeline on Google Colab, please do the following:

- Google Account: You need a Google account to access Google Colab and BigQuery.
- FRED API Key: Sign up for a FRED API account to obtain the API key. After signing up, get your API key from your FRED account.
- Google Cloud: Create a Google Cloud account and project. Then, get the BigQuery API and get your project ID from your Google Cloud account.

Note: Please make sure your Google Colab and Google Cloud are associated with the same email address.

ETL Process

Data Ingestion

Customizable Parameters

When interacting with the FRED API, you have the flexibility to customize the extraction process by changing various parameters. The following parameters are available for customization:

- `observation_start`: Specifies the start date for the observation.
- `observation_end`: Specifies the end date for for the observation.
- `limit`: Sets the maximum number of data entries to extract.
- `sort_order`: Determines the sorting order of the data ('asc' for ascending, 'desc' for descending).
- `unit`: Specifies the unit for data transformation. Available units include:
  - 'lin': Loan Amount in Billions.
  - 'chg': Loan Amount Change from Previous Week in Billions.
  - 'ch1': Loan Amount Change from Previous Year in Billions.
  - 'pch': Weekly Percent Change.
  - 'pc1': Yearly Percent Change.
  - 'pca': Compounded Annual Rate of Change.

  - 'cch': Continuously Compounded Rate of Change.
  - 'cca': Continuously Compounded Annual Rate of Change.
  - 'log': Index.

After customization, data is extracted from the FRED API in JSON format. If you choose not to customize these parameters, default values are automatically applied from the default parameter dictionary.

Data Transformation

You can optimize the data transformation experience by selecting multiple units such as unit_list = ['lin', 'chg', 'pch']. The number of datasets corresponds to the number of items in the unit_list. If you decide to include all the available units in the unit_list, after the data extraction process, you should get 9 datasets.

First Data Frame:

The primary goal is to create a merged dataframe that consists of one date column and an extra column for each selected unit.

- Data Cleaning: Remove rows with unavailable data denoted by `'.'`. For instance, if you set the observation start date to January 10, 2003, and the earliest available data coincides with this date, the loan amount will be present. However, for certain units like '`ch1`' (Loan Amount Change from Previous Year in Billions), the value may be `'.'` since there is no previous year data for comparison.
- DataFrame Creation: Create a dataframe by extracting only the date and value columns from each dataset dictionary. Redundant columns like `observation_start` and `observation_end` are excluded from the dataframes, as these parameters were specified before data extraction.
- Data Merging: Merge multiple dataframes using an inner join. Rows with no available data are removed during this process so only columns with the same date are merged.
- Column Renaming: Rename the value column to correspond to the unit name to enhance you understanding. For example, '`lin`' becomes Loan Amount in Billions and '`chg`' becomes Loan Amount Change from Previous Week in Billions.
- Data Type Adjustment: Change the data type of unit columns to `float64`, and the date column to `datetime`.

Second Data Frame:

The primary goal of is to enable you to choose an aggregated column, group them by years, and calculate various metrics from the selected column.

- Adding Year Column: Introduce a new column, Year, and extract the year information from the date column.
- Aggregating a Column: Choose a column to be aggregated by years and calculate various metrics such as total, average, highest, and lowest values.

Now, both data frames are ready. The next step is to load them into Google BigQuery for further analysis.

Data Loading

The two data frames are loaded to the target database, Google Big Query. To maintain consistency, the schema design for the BigQuery tables align with the data types of the existing data frames. The for loop will automatically determine the column types of the existing data frames and apply them to the new tables. To ensure the data loaded successfully, a query is then executed to print a few lines from the loaded data.

Running the ETL Pipeline on Google Colab

To execute the ETL pipeline successfully, download the provided Python script from GitHub and run each cell in sequence. This step-by-step process will make the pipeline run smoothly. After completing the pipeline, you can proceed to analyze the transformed data in Google BigQuery. Additionally, you have the option to export the data to platforms such as Looker Studio for advanced visualization and analysis.

Note: Prior to running the ETL pipeline, please have all the necessary prerequisites.

Troubleshooting

Please follow these steps:

- Make sure that you have installed the required libraries by executing the following commands:

  ```
  !pip install pandas
  !pip install google-cloud-bigquery
  ```

- Make sure to enter your Google Cloud project ID in the code.
- Make sure that your Google Colab email address matches the one you used to sign up for Google Cloud.

References/ Resources

- FRED API Series Observations Documentation
- Google Cloud BigQuery Python Client Library
- Documentation for to_gbq Function