Documentation for Python Script: "Data Extraction, Transformation, and Loading (ETL) Pipeline"

Introduction

This Python script defines an ETL (Extract, Transform, Load) pipeline using popular libraries such as Pandas, Prefect, and SQLAlchemy. The script's primary purpose is to retrieve data from an coincap API, transform it into a desired format, and load it into a PostgreSQL database.

Libraries Used

The script uses the following libraries:

- `os`: Provides functions for interacting with the operating system.
- `json`: Allows parsing JSON data.
- `requests`: Enables making HTTP GET requests to an API.
- `pandas`: Provides data manipulation and analysis capabilities using DataFrames.
- `prefect`: A workflow automation and scheduling library.
- `sqlalchemy`: Provides tools for working with SQL databases.

Prefect Flow

The ETL process is encapsulated within a Prefect flow called `Extract_Load_transform`. This flow consists of three primary tasks:

1. **`get_asset_data`**: This task fetches data from the CoinCap API and saves it as a CSV file. The function accepts two arguments, the API URL, and the path to save the CSV file. It follows these steps:

   - Sends a GET request to the specified API URL.
   - Checks if the request was successful (HTTP status code 200).
   - Parses the JSON response into a Pandas DataFrame.
   - Ensures that the destination directory for the CSV file exists.
   - Saves the DataFrame as a CSV file in the specified path.
   - Returns the path of the saved CSV file.

2. **`transform_asset`**: This task transforms the raw asset data into a cleaned format for further analysis. It takes the path of the CSV file as input and performs the following transformations:

   - Loads data from the CSV file into a Pandas DataFrame.
   - Cleans the data by replacing None values with appropriate defaults (e.g., 0).
   - Converts specific columns to their appropriate data types (e.g., float).
   - Removes the 'explorer' column from the DataFrame.

   - Returns the cleaned DataFrame.

3. **`load_postgres`**: This task loads the transformed data into a PostgreSQL database using SQLAlchemy ORM. It accepts the cleaned DataFrame as input and follows these steps:

   - Defines the PostgreSQL database connection URL.
   - Creates an SQLAlchemy engine to connect to the database.
   - Uses the `to_sql` method to insert the DataFrame into the 'asset' table in the database.
   - Sets `if_exists='append'` to add data to the table without overwriting existing records.
   - Sets `index=False` to avoid saving the DataFrame index.
   - Prints a message indicating that the data upload is complete.

Execution

To execute the ETL pipeline, the script defines an `if __name__ == "__main__":` block that runs the `Extract_Load_transform` Prefect flow when the script is directly executed. This initiates the entire ETL process.

Configuration

Ensure you configure the following parameters as needed before running the script:

- `url`: The URL of the CoinCap API for data extraction.
- `csv_file_path`: The path to save the CSV file containing raw data.
- Database connection URL (`db_url`) in the `load_postgres` task.

Running the Script

To run the script:

1. Ensure that you have the required libraries installed. You can install them using `pip install <library-name>` if necessary.

2. Configure the parameters mentioned in the "Configuration" section.

3. Execute the script, and it will initiate the ETL process, including data extraction, transformation, and loading into a PostgreSQL database.

4. Monitor the script's progress and check the output messages for any errors or notifications.

Conclusion

This script provides a foundation for building ETL pipelines in Python, combining data retrieval from external sources, data transformation using Pandas, and data storage in a relational database. You can adapt and extend this script to meet your specific ETL requirements and integrate it into larger data processing workflows.