Project: Data Report

03.06.2024

1 Question

Analysing the trends in Greenhouse Gas Emissions with Energy Consumption by sector.

2 Data Sources

2.1 Source Description

The data sources chosen for this project are:

- **Net Greenhouse Gas Emissions:** Retrieved from Eurostat. It has columns with Country and Net emissions values from 1990-2022.
- **Final energy consumption by sector:** Retrieved from Eurostat. It has columns with Country name and the energy consumption values from 2011-2022.

  These datasets were selected because they provide comprehensive and reliable data on crucial environmental indicators. Eurostat is a trusted source for European statistics, ensuring data quality and consistency.

2.2 Data Structure and Quality

Both datasets are in TSV (Tab-Separated Values) format, which is straightforward to parse and process. The data contains various columns representing different aspects of emissions and renewable energy usage, such as country codes, years, and values.

2.3 Licenses and Usage

The data is provided under standard open-data licenses available on the Eurostat website. The license allows for free use of the data with proper attribution. We plan to fulfill the obligations by citing Eurostat as the data source in any derivative works or reports.

3 Data Pipeline

3.1 Overview

The data pipeline was implemented using Python, leveraging libraries such as `requests`, `pandas`, and `sqlite3`. The pipeline performs the following steps:

1. Downloading data from the specified URLs.
2. Reading the downloaded data into pandas DataFrames.
3. Cleaning and preprocessing the data (filling missing values, normalizing column names).
4. Storing the cleaned data into SQLite databases.

3.2 Transformation and Cleaning Steps

The data undergoes several transformation steps:

- **Filling Missing Values:** Missing values are filled with 0 to ensure completeness of the dataset.
- **Normalizing Column Names:** Column names are stripped of whitespace and converted to lowercase to maintain consistency.

3.3 Challenges and Solutions

During the development of the pipeline, the following challenges were encountered:

- **Data Quality Issues:** Some datasets had missing values. This was addressed by filling missing values with 0.

- **URL Response Handling:** Ensuring the URL responses were properly handled and any errors were caught using `response.raise`$_{f}or_{s}tatus()$.

The pipeline is designed to handle errors by raising exceptions when HTTP requests fail and logging these errors for further inspection.

4 Results and Limitations

4.1 Output Data

The output of the pipeline is two SQLite databases containing the cleaned data for greenhouse gas emissions and renewable energy shares. The data is structured in tables, making it easy to query and analyze.

4.2 Data Structure and Quality

The resultant data maintains the original structure but with cleaned and normalized values. The quality of the data is high, given the preprocessing steps to handle missing values and ensure consistency in column naming.

4.3 Data Format

SQLite was chosen as the output format due to its lightweight, standalone nature, and ease of use for data storage and retrieval.

4.4 Critical Reflection

While the data pipeline effectively automates data handling, potential issues include:

- **Data Updates:** Regular updates to the source data might require periodic execution of the pipeline.
- **Data Completeness:** Filling missing values with 0 could lead to misinterpretation. Future improvements might include more sophisticated methods for handling missing data.

5 Data Pipeline

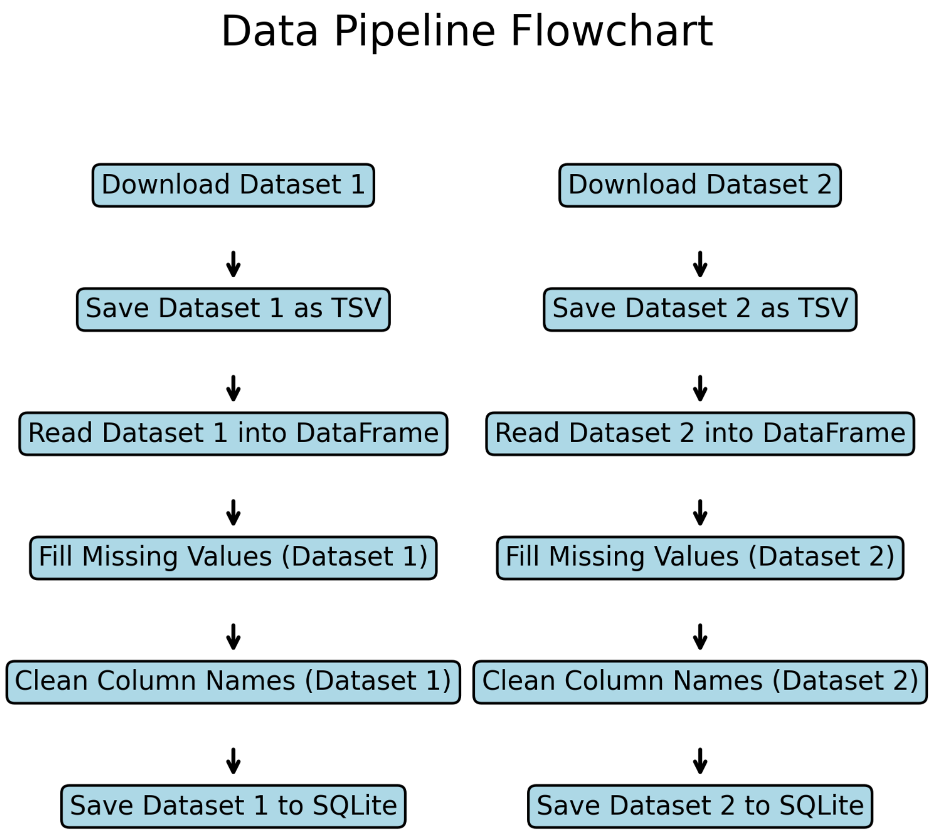
Data Pipeline Flowchart
Download Dataset 1
Save Dataset 1 as TSV
Read Dataset 1 into DataFrame
Fill Missing Values (Dataset 1)
Clean Column Names (Dataset 1)
Save Dataset 1 to SQLite
Download Dataset 2
Save Dataset 2 as TSV
Read Dataset 2 into DataFrame
Fill Missing Values (Dataset 2)
Clean Column Names (Dataset 2)
Save Dataset 2 to SQLite

Figure 1: Data Pipeline Flowchart