# Analysis

Based on the requirements section, a use case has been derived that describes the main scenario that the system would perform and its extensions.

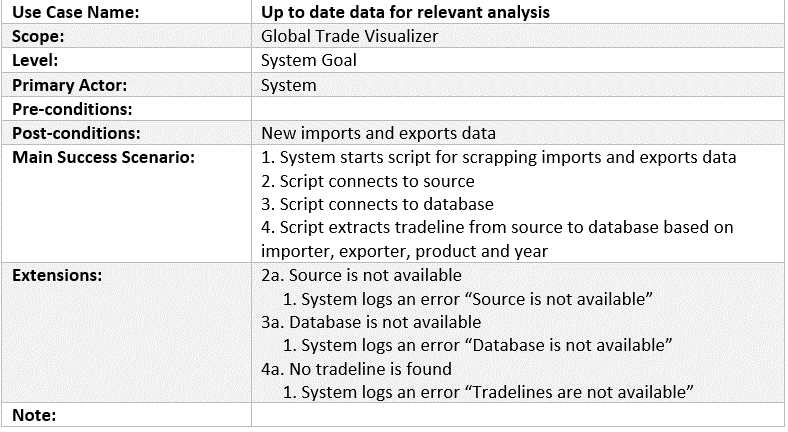


Figure 1 Up to Date Use Case

The main success scenario involves the source, the database and the tradeline that are requested based on the importer, exporter, product and year. These are the main elements that help in fulfilling the requirement of having up to date data in the system. Their interaction forms a data pipeline that feeds new information into the system. These conceptual objects form what is known as the domain model for the system.



Figure 2 Domain Model

From the diagram it can be seen that the source provides the tradeline for the database as described in the use case. The attributes of tradeline are what helps in identifying them for the extraction. The domain model is one of the main contributors in defining the design in the next section.

# Design

A more detailed approach will provide a design class diagram for the objects extracted from the domain model. Combining it with the information in the use case, an interface is added to provide the processing of the tradeline. The result is presented in the following diagram:

# 

Figure Design Class Diagram

The actor of the use case generates events as it interacts with the system. For the current use case, the interaction can be described with a sequence diagram:

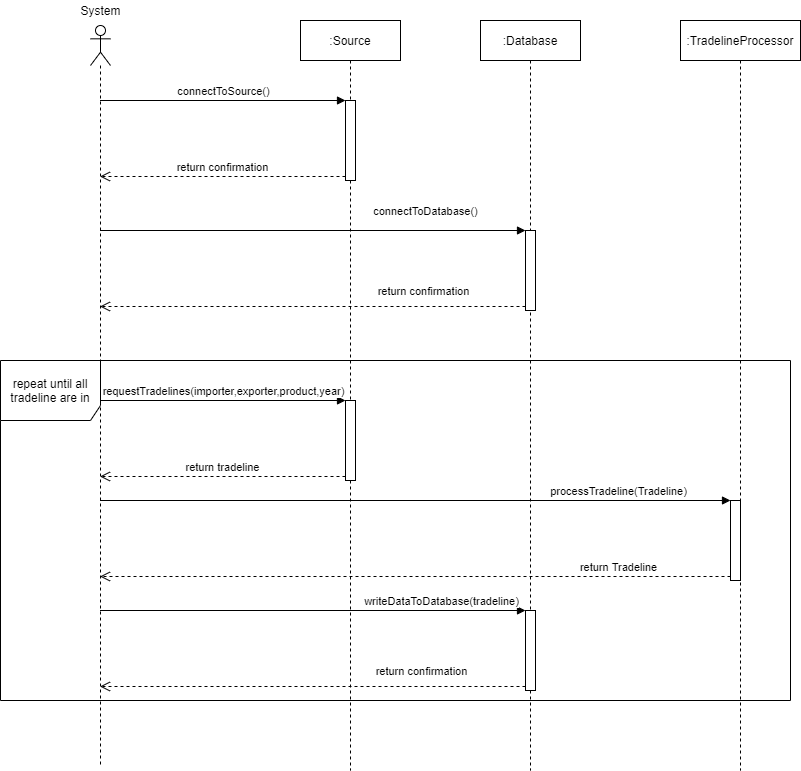


Figure 4 Sequence Diagram

After the connections are established to the source and database, a looping occurs until all the data has been brought in the database. In this loop, the data is processed and prepared so the database can accept it. The responsibilities of the source and database are to act as the provider and holder of tradelines for the system.

Operations are the building block for designing this use case. A language that is flexible, when it comes to designing these, is Python. It supports multiple programming paradigms, including structured, object-oriented, and functional programming. Being a process similar to a data pipeline, a more functional style of programming will help.

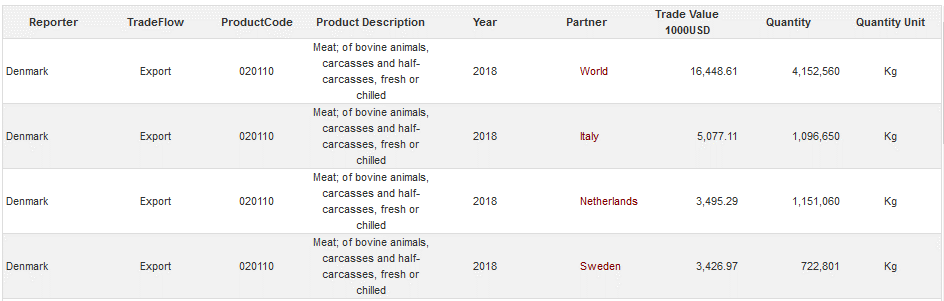
In the following section, the implementation of the operation will be detailed.

# Implementation

Following the design, the implementation of the use case was made in Python. The source for the tradelines is the World Integrated Trade Solution’s website. The destination is an SQL database.

The code works by connecting to the source which will be used in creating get requests to the website. The source acts as a browser to the webpages. With special headers, most websites will see the connection as just another person.

The following is an example of tradelines from the source:



This is the data that is meant to be stored in the database.

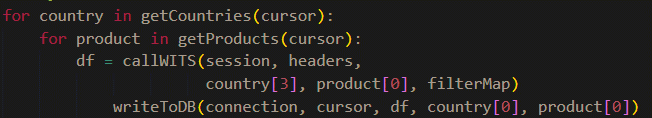
A connection is then established to the database which returns a cursor and a connection. The cursor is used to executes queries and retrieve the responses. The connection is to commit what the cursor has done.

The get requests are sent with exporter country, product and year as parameters. After the response is received, it is cleaned through a filter for special words like “SELECT”, “INSERT”,” DATABASE”, “DROP” etc. This protects the database from attacks, by not enabling any special words to get through. The second way the database is protected is by having the insert statements be parameterized.

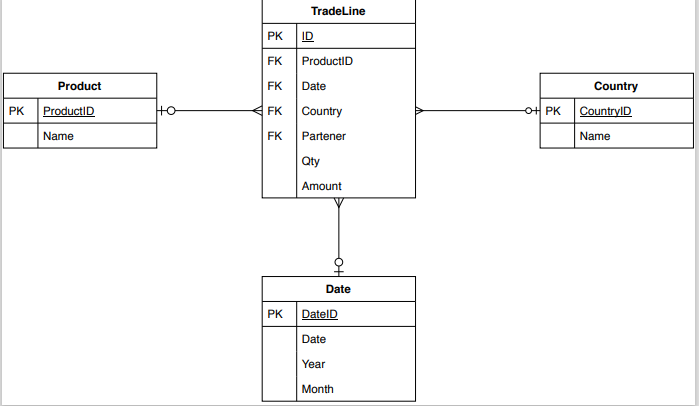


Cleaning the results and having them as parameters protects from malicious attacks. This is important when working with a lot of requests to a web. The next step is to write to the database. An insert statement is executed through the cursor and then committed if there are no errors. If this is not successful, the error is caught and the code proceeds. This will enable the process to be shortened if errors appear along the way and it must be restarted.

The process of requesting the data, cleaning it and then persisting it in the database is done for each country and product that is available. The code receives the available countries and products from the database.



The SQL Database contains multiple tables that work to complete the use case.



For each product, country or date there are multiple tradeline entries. After the countries and products available are obtained from the database, they are used to send requests to the source to obtain the s linked to them. The date is the latest one available.

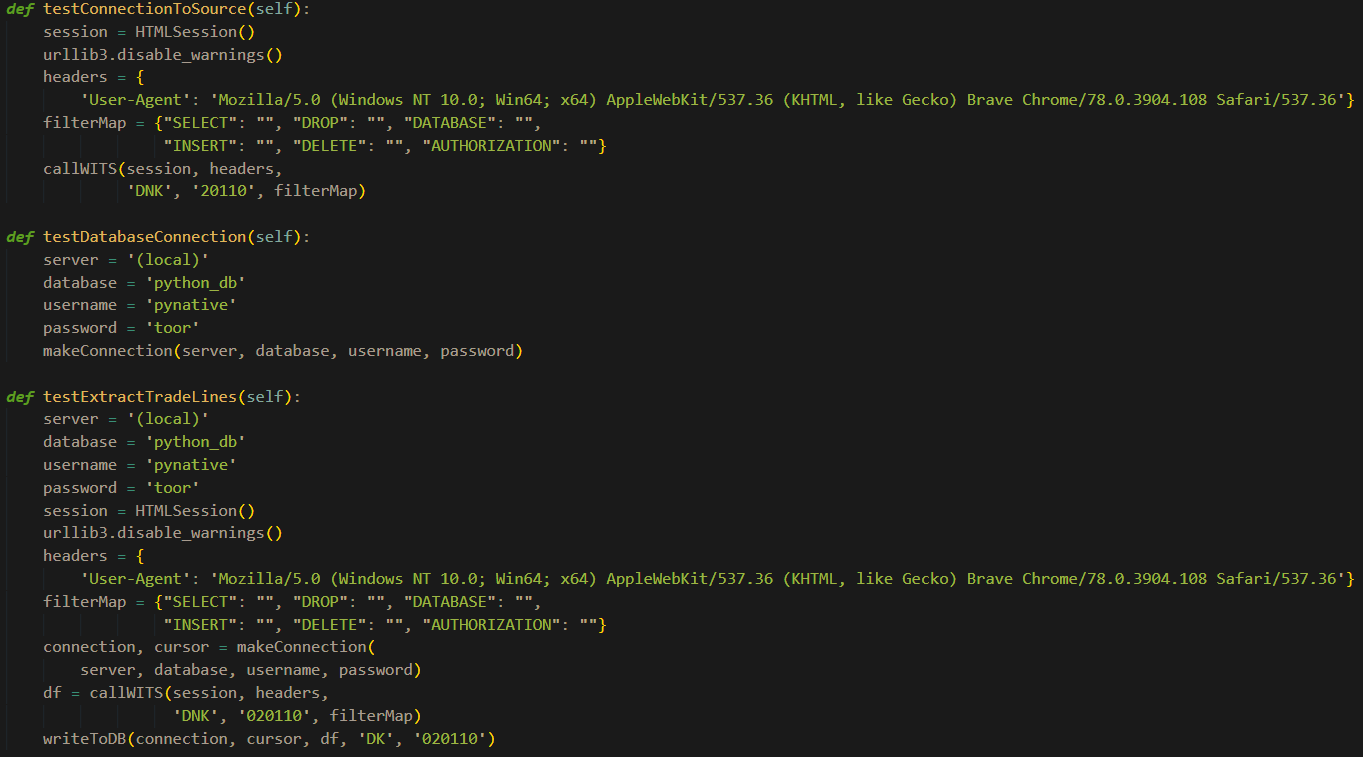
This concludes the implementation and will be followed by the testing phase.

## Testing

After the implementation, the testing will consist of unit tests based on test cases. The test cases are derived from the use case.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nr | Description | Actor | Precondition | Expected Result | Steps | Result |
| 1 | Extract Tradeline to Database | System | None | Successfully added new Tradeline to Database | 1. Connect to Source  2. Connect to Database  3. Extract Tradeline based on importer, exporter, product and year | New Tradeline added to Database |

Based on this table. Each of the steps have been tested using unit tests.



The tests are asserting if any error is being thrown with the demo data. There are 3 tests, one for each step of the sunny scenario. The first test involves setting up a connection to the source. The second one checks a default example for connecting to the database and the third one is extracting tradeline from to the source to the database.