

*Our project is the parsing of the sites on Python*

*This is a method for extracting information from websites. This method is primarily aimed at converting unstructured data - in HTML format - on the Internet into structured data: databases or spreadsheets. Site parsing involves accessing the Internet directly via HTTP or via a web browser.*

***Action sequencing***

*1)Get the url of the page we want to fetch data from.*

*2)Copy or download the HTML content of the page.*

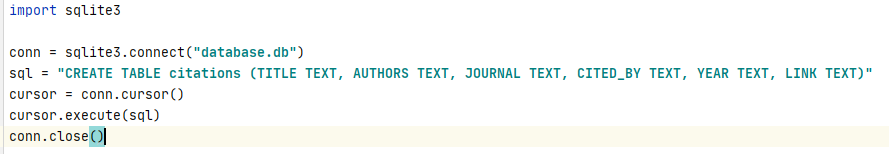
*3)Parse the HTML content and get the data you need.*

*This sequence will help you navigate to the URL of the desired page, retrieve the HTML content, and parse the required data. But sometimes you need to first enter a site and then go to a specific address to get the data. In this case, another step is added to enter the site.*

*To analyze HTML content and get the necessary data, use the "Beautiful Soup" library. To log in to a website, go to the desired URL in a single session, and download HTML content, we will use the Selenium library. Selenium Python helps with button clicks, content input, and other manipulations.*

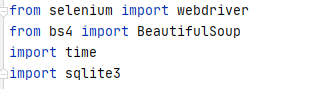
***Explanation of the code***

*Initially, we should create a database in sql format by running this code. The code creates sql file that already contains necessary columns.*

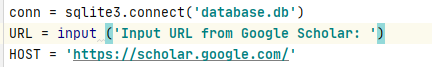
**

*Now, let’s move to the main code*

*First, we import the libraries that we will use:*



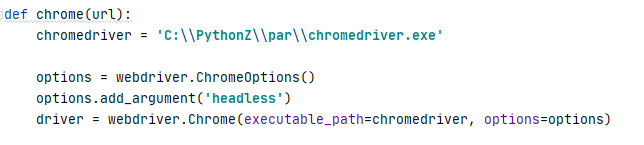
*Then, we connect sql file we created earlier with this code by conn object. Also, the code should input a page from which it will parse the information. We need HOST library in order to get working links to every article from the page.*

**

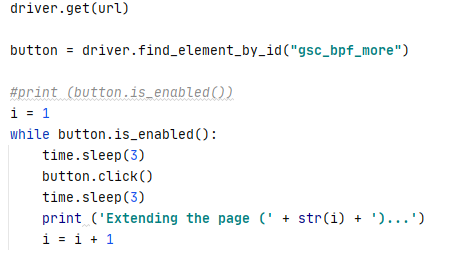
*At the same time, create articles dictionary, which will contain all the data about articles.*

Articles = []

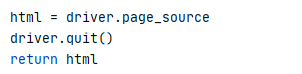
*After that, we create ‘chrome’ function, where we tell the browser driver the path to "Selenium" to launch our web browser. To avoid displaying the browser's graphical interface, add the "headless" option to Selenium.*



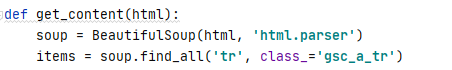
*Next, we pass the link to our driver, find “Show More” button and assign it to ‘button’ object. In ‘while’ loop, we click the button until it becomes disabled*

**

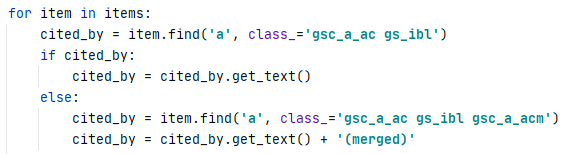
*Finally, the function copies the html code of the extended page, exit from browser and returns the code.*

**

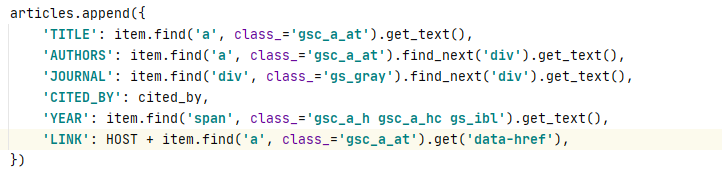
*In get\_content function, we start to parse the transmitted html code by retrieving to ‘items’ object only chunks of code under ‘gsc\_a\_tr’ class and ‘tr’ tags.*

**

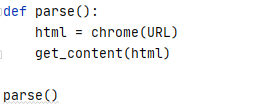
*In the loop that goes through every item, we are looking for text about an article. Classes of text that represents the number of citation of an article can have different classes, so we have to write conditions for them.*

**

*We add to ‘articles’ dictionary the rest data about an article, which is under other tags and classes. (To get a correct link to the article, we must add to the cut link under ‘gsc\_a\_at’ class and ‘a’ tag HOST variable we created earlier)*

**

*In ‘parse’, the main function, we just transfer given URL to ‘chrome’ function and then pass html code we have got from ‘chrome’ function to ‘get\_content’ function.*

**

*After that, we pass all the data from ‘articles’ dictionary to the database we created earlier.*

**