Chapter 4: Storing Scraped Data to a Database

DR. LINDA MAHMOUDI

”يرفعِ اللهُ الذين أمنوا منكم والذين أوتوا العلم درجات“

In this chapter, we'll learn how to:

- Storing scraped data to **CSV File,**
- Storing scraped data into a **MySQL Database,**
- Scrape **Multiple Web Pages** Using Python.

Storing data to CSV File

- CSV, or comma-separated values, is one of the most popular file formats in which to store spreadsheet data.
- It is supported by Microsoft Excel and many other applications because of its simplicity.
- To write to a CSV file in Python, we need to use the CSV module with these 4 steps:

1. Open a CSV file in the write mode:

This happens using the open () function. Give it the path of the file as the first argument. Specify the mode as the second argument (‘r’ for read and ‘w’ for write).

2. Create a CSV writer object:

To do this, create a csv module’s writer () object, and pass the opened file as its argument.

3. Write data to the CSV file:

Use the writer object’s writerow () function to write data into the CSV file.

4. Close the CSV file:

Using the close () method of a file.

Here is a practical example:

- Modifying a CSV file, or even creating one entirely from scratch, is extremely easy with Python's csv library:

```
import csv
csvFile = open('test.csv', 'w+')
```

  - reminder: If *test.csv* does not already exist, Python will create the file (but not the directory) automatically. If it already exists, Python will overwrite *test.csv* with the new data.

To support writing non-ASCII values to a CSV file, specify the character encoding in the open () call as the third argument.

```
csvFile = open('test.csv', 'w', encoding="UTF8")
```

Write in the CSV file#

```
writer = csv.writer(csvFile)
writer.writerow(('number', 'number plus 2', 'number times 2'))
for i in range(10):
writer.writerow( (i, i+2, i*2))
```

In Python, we can use the CSV writer's **writerows ()** function to write multiple rows into CSV file on the same go.

```
writer.writerows(('number', 'number plus 2', 'number times 2'), (10, 12, 20))
```

Close the CSV file#

```
csvFile.close()
```

After running, we should see a CSV file:

```
number,number plus 2,number times 2
0,2,0
1,3,2
2,4,4
...
```

- One common web scraping task is to retrieve an HTML table and write it as a CSV file.
- Wikipedia's Comparison of Text Editors provides a fairly complex HTML table, complete with color coding, links, sorting, and other HTML garbage that needs to be discarded before it can be written to CSV.
- Using BeautifulSoup and the get_text() function copiously, we can do that in fewer than 20 lines:

```
import csv
from urllib.request import urlopen
from bs4 import BeautifulSoup
```

```
html = urlopen('http://en.wikipedia.org/wiki/Comparison_of_text_editors')
bs = BeautifulSoup(html, 'html.parser')
#The main comparison table is currently the first table on the page
table = bs.findAll('table',{'class':'wikitable'})[0]
rows = table.findAll('tr')
csvFile = open('editors.csv', 'w+')

writer = csv.writer(csvFile)
for row in rows:
        csvRow = []
        for cell in row.findAll(['td', 'th']):
                csvRow.append(cell.get_text())
        writer.writerow(csvRow)
csvFile.close()
```

The result should be a well-formatted CSV file saved locally, under *../files/editors.csv*.

Storing data into a MySQL Database

- Unfortunately, Python support for MySQL is not built in. However, many open source libraries, both with Python 2.x and Python 3.x, allow you to interact with a MySQL database.
- One of the most popular of these is PyMySQL, which can be installed using pip:

```
$ pip install PyMySQL
```

❖ To insert data into a MySQL table using Python, we follow these 4 steps:

1. Connect to the MySQL database server
2. Initiate a MySQL Cursor object
3. Execute the Insert statement to insert data into the table
4. Close the database connection.

Here is a practical example:

In this example, the database is already prepared to accept a wide variety of all that Wikipedia can throw at it

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import datetime
import random
import pymysql
import re
```

#Establishing the connection

```
conn = pymysql.connect(host='127.0.0.1', unix_socket='/tmp/mysql.sock', user='root',
password=None, db='mysql', charset='utf8')
```

#create cursor, used to execute commands

```
cur = conn.cursor()
cur.execute("USE scraping")
random.seed(datetime.datetime.now())
```

```
#define and execute sql statement
def store(title, content):
cur.execute('INSERT INTO pages (title, content) VALUES ("%s", "%s")', (title, content))
#commit the changes in the database
cur.connection.commit()
def getLinks(articleUrl):
html = urlopen('http://en.wikipedia.org'+articleUrl)
bs = BeautifulSoup(html, 'html.parser')
title = bs.find('h1').get_text()
content = bs.find('div', {'id':'mw-content-text'}).find('p').get_text()
store(title, content)
```

```
return bs.find('div', {'id':'bodyContent'}).findAll('a',
href=re.compile('^(/wiki/)((?!:).)*$'))
links = getLinks('/wiki/Kevin_Bacon')
while len(links) > 0:
newArticle = links[random.randint(0, len(links)-1)].attrs['href']
print(newArticle)
links = getLinks(newArticle)
#close cursor and connection to database
cur.close()
conn.close()
```