Data Loading

March 22, 2018

0.1 Load data into memory

Data is often stored in file format such as CSV format or Excel workbook. This section covers how to load data into memory

1) first we need to know the directory of the data file
2) load the data file into memory using Python
3) Do data analysis and store the results into file

```
In [13]: # Import `os`
         import os

         # Retrieve current working directory (`cwd`)
         cwd = os.getcwd()
         cwd

         # Change directory
         os.chdir("C:\\Users\\yliu3\\Documents\\Spring2018\\MATH303_503\\Lecture")
         cwd = os.getcwd()
         cwd
         # Import pandas
         import pandas as pd

         # Load csv
         df = pd.read_csv("hour.csv")
         df.head()
         #get the number of rows and columns
         df.shape
         #gives number of row count
         Count_Row=df.shape[0]
         #gives number of col count
         Count_Col=df.shape[1]

         #create a new column
         df['sum'] = df['registered'] + df['cnt']
         df.head()
         #save the dataframe into a csv file
         #When you are storing a DataFrame object into a csv file using the to_csv method,
         #it also store the preceding indices of each row of the DataFrame object
         df.to_csv("newhour.csv") #notice: the output has indices in the 1st column

         #to ingnore the indices of each row of the DataFrame object, set index prameter to be F
         df.to_csv("newhour2.csv", index=False)
```

```
In [21]: import pandas
         #Load data into memory
         df = pandas.read_excel(open('day.xlsx','rb'), sheetname='mydata')
         df.shape
         # or using sheet index starting 0
         df = pandas.read_excel(open('day.xlsx','rb'), sheetname=0)
         df.head()
         #add new column
         df['ratio'] = df['registered']/df['cnt']
         df.head()

         #save Data frame to Excel file
         # DF TO EXCEL

         writer = pd.ExcelWriter('daynew.xlsx')
         df.to_excel(writer,'mydaysheet', index=False)
         writer.save()
```

SQLite is a C library that provides a lightweight disk-based database that doesn't require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language. Some applications can use SQLite for internal data storage. It's also possible to prototype an application using SQLite and then port the code to a larger database such as PostgreSQL or Oracle.

```
In [32]: import sqlite3

         conn = sqlite3.connect('emaildb.sqlite')
         cur = conn.cursor()

         cur.execute('DROP TABLE IF EXISTS Counts')

         cur.execute('''
         CREATE TABLE Counts (email TEXT, count INTEGER)''')

         fname = input('Enter file name: ')
         if (len(fname) < 1): fname = 'mbox.txt'
         fh = open(fname)
         for line in fh:
             if not line.startswith('From: '): continue
             pieces = line.split()
```

```
    email = pieces[1]
    cur.execute('SELECT count FROM Counts WHERE email = ? ', (email,))
    row = cur.fetchone()
    if row is None:
        cur.execute('''INSERT INTO Counts (email, count)
                VALUES (?, 1)''', (email,))
    else:
        cur.execute('UPDATE Counts SET count = count + 1 WHERE email = ?',
                    (email,))
    conn.commit()

# https://www.sqlite.org/lang_select.html
sqlstr = 'SELECT email, count FROM Counts ORDER BY count DESC LIMIT 10'

for row in cur.execute(sqlstr):
    print(str(row[0]), row[1])

cur.close()
```

```
Enter file name:
zqian@umich.edu 195
mmmay@indiana.edu 161
cwen@iupui.edu 158
chmaurer@iupui.edu 111
aaronz@vt.edu 110
ian@caret.cam.ac.uk 96
jimeng@umich.edu 93
rjlowe@iupui.edu 90
dlhaines@umich.edu 84
david.horwitz@uct.ac.za 67
```