Week 6 assignment

Name: File ingestion and schema validation
Batch code: LISUM05
Submission date: 01/29/2022
Submitted to: Data Glacier

SUMARY OF THE FILE

Total number of rows: 9000 000

total number of columns: 14

file size: 2.94 GB

Dataset downloaded from Kaggle. (Seattle Library Collection Inventory)

url: https://www.kaggle.com/city-of-seattle/seattle-library-collection-inventory

File Name: dataset.csv

Size: 2.94GB (10.96 GB original data set size)

First at all, we only load the first 900000 rows from the original csv file, it represents us a csv file with 2.94 GB.

As first read method I utilize pandas and the magical function %lprun from “line_profiler” module with test purposes.

The results:

```
import pandas as pd
%load_ext line_profiler

def load_csv():
    dataset = pd.read_csv('dataset.csv')


%lprun -f load_csv load_csv()
```

```
Timer unit: 1e-07 s

Total time: 67.9148 s
File: <ipython-input-4-0241ecb78712>
Function: load_csv at line 4

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
     4                                           def load_csv():
     5         1  679148066.0 679148066.0    100.0      dataset = pd.read_csv('dataset.csv')
```

Fig. 1

So, with read_csv from pandas it takes 67.91 s this will be taken as a reference.

After, we tried with Dask:

```
# DASK TEST
import dask.dataframe as dd

%reload_ext line_profiler

def load_csv_dask():
    dataset = dd.read_csv('dataset.csv')


%lprun -f load_csv_dask load_csv_dask()
```

```
Timer unit: 1e-07 s

Total time: 0.0524728 s
File: <ipython-input-7-16e7e280647e>
Function: load_csv_dask at line 6

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
     6                                           def load_csv_dask():
     7         1     524728.0 524728.0    100.0      dataset = dd.read_csv('dataset.csv')
```

Fig. 2

It takes 52 ms to load the csv file.

Later, we tried Modin and Ray

```
Timer unit: 1e-07 s

Total time: 328.364 s
File: <ipython-input-2-2ff0c1bb4bd8>
Function: load_csv_modin at line 8

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
     8                                           def load_csv_modin():
     9         1 3283638768.0 3283638768.0    100.0      dataset = pd.read_csv('dataset.csv')
```

Fig. 3

It takes 328 s to load the csv file.

To perform basic validation on data columns : (remove special character , white spaces from the col name) we utilized the sample code given by Data Glacier (functions defined in testutility.py in the next python jupyter notebook cells:

```
%%writefile file.yaml
file_type: csv
dataset_name: library
file_name: dataset
inbound_delimiter: ","
outbound_delimiter: "|"
columns:
    - Unnamed_0
    - BibNum
    - Title
    - Author
    - ISBN
    - PublicationYear
    - Publisher
    - Subjects
    - ItemType
    - ItemCollection
    - FloatingItem
    - ItemLocation
    - ReportDate
    - ItemCount
```

```
# Read config file
import testutility as util
config_data = util.read_config_file("file.yaml")
# read the file using config file
file_type = config_data['file_type']
source_file = "./" + config_data['file_name'] + f'.{file_type}'
#print("",source_file)
df = pd.read_csv(source_file,config_data['inbound_delimiter'])
df.head()
```

So, from parameters in yaml file after read the csv file and clean it, validating after each step, at the end we indicated to write a new csv file with delimiter '|' and compressed to gz format.

```
import pandas as pd
# writing to csv compressed file
df.to_csv("output_dataset.gz",index=False, sep=config_data['outbound_delimiter'], compression='gzip')
```

The output file is created "output_dataset.gz" and the size was 992 MB. (one third from original)

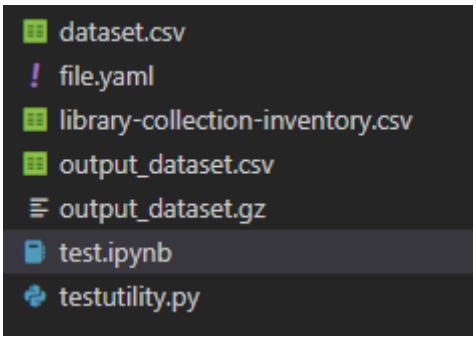

Fig. 4