Project #4 (Apache Spark)

CSC 261/461 (Database Systems), Spring 2018,
University of Rochester
Due Date: 05/03/2018 (11:59 pm)
This is an optional project. Go for it if you want to learn Spark

Introduction

Apache Spark™ is a fast and general engine for large-scale (big) data processing. Spark is faster and easier than Hadoop and it goes beyond typical batch processing (map-reduce) to supprot multiple types of computations (e.g. text processing, SQL queries, and machine learning). For this project, we will use python 2 (spark) kernel via Jupyter notebooks provided and hosted by CIRC.

Data

All the required data files are located at `/public/tbiswas2/csc261/spark` folder. You do not need to copy these files. This folder contains a txt file (`bronte-jane-eyre.txt`) and a directory `wegmans` which further contains four other files storing data related to real transactions in Wegmans between April 2013 and April 2014.

Task 1: RDD operations

Create a new Python 2 (Spark) notebook `proj4task1.ipynb` on bluehive. Perform the follwoing operations (mostly based on MapReduce model) on the file `bronte-jane-eyre.txt` .
This file contains the famous novel Jane Eyre by Charlotte Bronte in plain text format.

1. Count the occurrences of each character (a-z ignore-case). Produce a histogram from this data.

2. Find all the anagrams (a word, phrase, or name formed by rearranging the letters of another, such as cinema, formed from iceman) present in the file.
   (Coding tip: You need to store each word as a key-value pair. The key would be a string produced from the sorted letters present in the word).
   Display the key and all the anagrams sorted by the key.

   Sample answer:

   ```
   [(u'aadeemqrsu', {u'masquerade', u'squaremade'}),
   (u'aadins', {u'dianas', u'naiads'}),
   (u'aaginnostt', {u'antagonist', u'stagnation'}), ...
   (u'abest', {u'bates', u'beast', u'beats'}), ...]
   ```

3. Write a function that takes a word as input and returns a list of tuples containing the (line number, beginning character location) where line number and location gives the exact location of the word searched.

   For example, if you search for `Rochester` , the first entry in the list should be: `(4504,34)` .
   Allow any word starting with the search string. i.e,. your query for Rochester should match “Rochester’s”, “Rochester?” etc. The search should be case sensitive.

Task 2: Wegmans Data (Queries)

Create another Python 2 (Spark) notebook `proj4task2.ipynb` on bluehive. Perform the following queires. All the queries must be performed on the dataframes (i.e., do not use createOrReplaceTempView function)

You can find the details of the fields here: `http://www.cs.rochester.edu/courses/261/spring2018/projects/proj4/file_description.pdf`

1. How many stores are there in the sample?
2. How many distinct departments are there in the sample?
3. How many stores in the sample are in New York state (store_state)?
4. What is the range of POS transaction dates in the sample?
5. How many transaction entries are there from the WEGMANS MARKETPLACE store?
6. What is the average number of items in each transaction?
7. How many items falls into the WHOLE MILK class(class_name)?
8. Find the House-hold number (hshld_acct) which bought WHOLE MILK the most number of times. Display all the customer fields.
9. Find the heaviest (item_unt_qty) item(s) whose unit of measure (size_unit_desc) is LB. Display all the fields.
10. Find the top three (3) transactions with the most number of items sold (summation of unit_count for all the items present in that transaction). Display the transaction number and count
11. Find the average house hold size (hh_size) of customers for every store (where those customers made transactions). Display store name and average house hold size.
12. Find the top three (3) stores with the highest number of customers. Display the name of the stores and the number of customers.
13. Find the top three (3) stores with the highest sale (summation of net_sales) on any day. Display the store name and sale amount.
14. Find the top three (3) transactions with the highest net sales (net_sales). Display the transaction numbers and items bought (comma separated)

Submission instructions

For `proj4task1.ipynb`, have a markdown cell with content # Part <Part Number> immediately before the cell which produces the final output.

Similarly, for `proj4task2.ipynb`, have a markdown cell with content # Query <Query Number> immediately before the cell producing the final output.

There is no restriction on the number of cells you can use. But try to keep the notebooks concise and do not print or display any intermediate output.

To submit the project, first copy proj4task1.ipynb and proj4task2.ipynb files in the folder `proj4`.

Now, archive the file (with .tar extension).

```
cd ..
tar -cvf proj4.tar proj4
ls
```

(Note: `cd ..` takes you to the parent directory and `ls` command should display all the files int the directory including the tar file.)

You can submit your work on bluehive itself. submit the tar file using the command:

```
/public/hwen5/submit p4 proj4.tar
```

You should get a feedback whether the submission is successful or not.

You may resubmit as many times as you like; however, only the latest submission (along with the timestamp) will be saved, and we will use your latest submission for grading your work. Submissions via email will not be accepted! No late submission is allowed.

Note: This project is only for those students who love challenges. TAs will set up a demo session for you to explain your code or may ask you to reproduce it. If you can't explain/reproduce the outcome, we may treat that as violation of academic honesty policy.

How much does the project worth?: 3 pts (out of overall 100 pts). These points will be awarded after the grade assignment to all the students (which may/may not involve curving) to avoid negative impact on others grade. In other words, not submitting this project won't hurt your grade.