Design Laboratory (CS69202)

Spring Semester 2024

Assignment 5.1: NoSQL - Mapper, Reducer

Assignment date: January 31, 2024 (In Class Assignment)

Important Instructions:

1. You need to write Mapper, Reducer codes for the Queries. You have to fill the folders for Queries. You also need to write Make files so that on executing 'make' command in the terminal, the file does the needful and gives the necessary outputs.
2. Write Python code for the above Queries.
3. You must NOT use any libraries apart from sys.
4. You can use 1D lists as and when required based on Queries. You CANNOT use any other data structures for the assignment.
5. Not adhering to these instructions can incur a penalty (worst case being 0 marks).
6. You can write a readme file to provide any particular instructions related to program execution steps, input format, or anything that you might think is useful for the evaluator while evaluating the assignment.
7. Not writing Makefiles for the Queries will incur 50% of the marks against corresponding Queries
8. Plagiarism in any form is not allowed. Students found copying/sharing code will be awarded 0 marks. Any discussions among students are NOT allowed for the assignment.
9. All errors should be handled properly.
10. Also submit a README file.
11. Save this in a folder named in the format: <Roll No.>_DesLab_A5_1. Compress this folder to zip format, creating a compressed file <Roll No.>_DesLab_A5_1.zip. Upload this compressed file to moodle. Example: If your roll no. is 22CS60R05, the folder should be 22CS60R05_DesLab_A5_1, and the compressed file should be 22CS60R05_DesLab_A5_1.zip.

Problem Statement:

Phoenix, a Mathematician, wants to perform some analysis on a 1000 faced hypercube. The hypercube contains numbers from 1 to 1000. She generates random numbers by rolling the hypercube and stores it in data_0.txt. Also, to make her life easier, she divides the random numbers generated into 5 files and distributes the numbers randomly and names it data_1.txt, data_2.txt,...,data_5.txt. An example of data_i.txt where i = {0,1,2,3,4,5} is as follows:

data_i.txt

```
253
792
123
432
179
444
…….
```

Now, Phoenix wants your help in getting some of her queries resolved using the concept of Mapper and Reducer and helping her in her research. She provides you with the Query folders and the dataset data_i.txt in a folder named 'Assignment', compressed in a zip file so as to make your life easy. The queries are as follows :

Query 1:

Write Mapper, Reducer codes for finding the largest number in each dataset data_i.txt. The results are stored in the same folder in the form result_i.txt. You also need to create a result text result_6.txt which contains largest number in combined data_j.txt (j={1,2,3,4,5}), by using mapper function on each of data_j.txt by running it through a mapper function, and passing the output of each mapper code (combined) through reducer code. Note : use sort command after mapper. Also create make file for the same.

Output in respective files:

Largest Number = <Largest Number in the Dataset>

Query 2:

Write Mapper, Reducer codes for finding the smallest number in each dataset data_i.txt. The results are stored in the same folder in the form result_i.txt. You also need to create a result text result_6.txt which contains smallest number in combined data_j.txt (j={1,2,3,4,5}), by using mapper function on each of data_j.txt by running it through a mapper function, and passing the output of each mapper code (combined) through reducer code. Note : use sort command after mapper. Also create make file for the same.

Output in respective files:

Smallest Number = <Smallest Number in the Dataset>

Query 3:

Using data_j.txt, j = {1,2,3,4,5}, write Mapper codes to map data of each data file data_j.txt and pass the combined output, sort it and send it to reducer subroutine. Store the result in result_partial.txt of the form <key, value> where key = the number of the hypercube and value = the frequency of the number. Note : use sort command after mapper. You may use 1D list for reducer.py for storing and combining values. Mapper codes should NOT contain any data structures. Also create make file for the same.

Result_combine.txt contains :

<smallest value present in combined dataset, it's frequency>
<2nd smallest value present in combined dataset, it's frequency>
…
…
<2nd largest value present in combined dataset, it's frequency>
<largest value present in combined dataset, it's frequency>

Query 4:

Write Mapper, Reducer codes for finding the mean of combined data file result_combine.txt generated in Query 3 by processing each data file in mapper, sorting it and passing the output of the data files to reducer. The result is stored in the same folder in the form result_query1.txt. Note : use sort command after mapper. Also create make file for the same.

result_query1.txt contains : Mean = <Mean_of_the_dataset>

Query 5:

Write Mapper, Reducer codes for finding the mode list of combined data file result_combine.txt generated in Query 3 by processing data file in mapper, sorting it and passing the output of the data file to reducer. The result is stored in the same folder in the form result_query2.txt. Note : use sort command after mapper. Also create make file for the same. Mode list is the list of numbers whose frequency in the dataset is the highest.

result_query2.txt contains : Mode List = [<Mode_list_Values>]

Query 6:

Write Mapper, Reducer codes for finding the median of combined data file result_combine.txt generated in Query 3 by processing data file in mapper, sorting it and passing the output of the data file to reducer. The result is stored in the same folder in the form result_query3.txt. Do not USE sort function or any sorting algorithm to find the median.

Note : use sort command after mapper. Also create make file for the same. You may use 1D list for finding the median. For finding the number of random numbers generated (say n) in the dataset, you should not assume the value of n, rather calculate it and use it for inference. Handle the cases when n is odd/even. Failure to do so will incur a penalty.

result_query3.txt contains : Median = <Median_of_the_dataset>

Query 7:

Now, Phoenix wants to use the dataset information and create a custom probability distribution by using the numbers generated by the hypercube. Write Mapper, Reducer codes for producing the probability distribution of combined data file result_combine.txt generated in Query 3 by processing each data file in mapper, sorting it and passing the output of the 5 data files to reducer. The result is stored in the same folder in the form result_query4.txt. Also, consider a variable num_of_nums = 1000 i.e. the total number of numbers that can be generated from the hypercube is 1000.

Note : use sort command after mapper. Also create make file for the same. You may use 1D list for the same.

result_query4.txt contains :

<1, probability of 1 if dataset values are considered as a probability distribution>
<2, probability of 2 if dataset values are considered as a probability distribution>
…
…
<998, probability of 998 if dataset values are considered as a probability distribution>
<999, probability of 999 if dataset values are considered as a probability distribution>
<1000, probability of 1000 if dataset values are considered as a probability distribution>

Deliverables :

For Queries 1-7, write Mapper, Reducer, Makefile codes i.e. 7 Mapper.py, 7 Reducer.py and 7 Makefile. You will be provided with the zip file.

All you need to do is extract files from zip file, write mapper, reducer and makefile codes in each of the folders Query<i> where i belongs to one of the 7 queries given in the assignment.

After writing the codes, zip the same file under the name <RollNo>_*DesLab*_A5_1.zip. For more see last point of the Instructions given in the first page itself.

Hint : Mapper codes for Queries 1-3 are the same.
Mapper codes for Queries 4-7 are the same.
Reducer codes are the same for all. You have to manipulate codes based on your query.