Exp.No.: 3 Map Reduce program to process a weather dataset

AIM:

To implement MapReduce program to process a weather dataset.

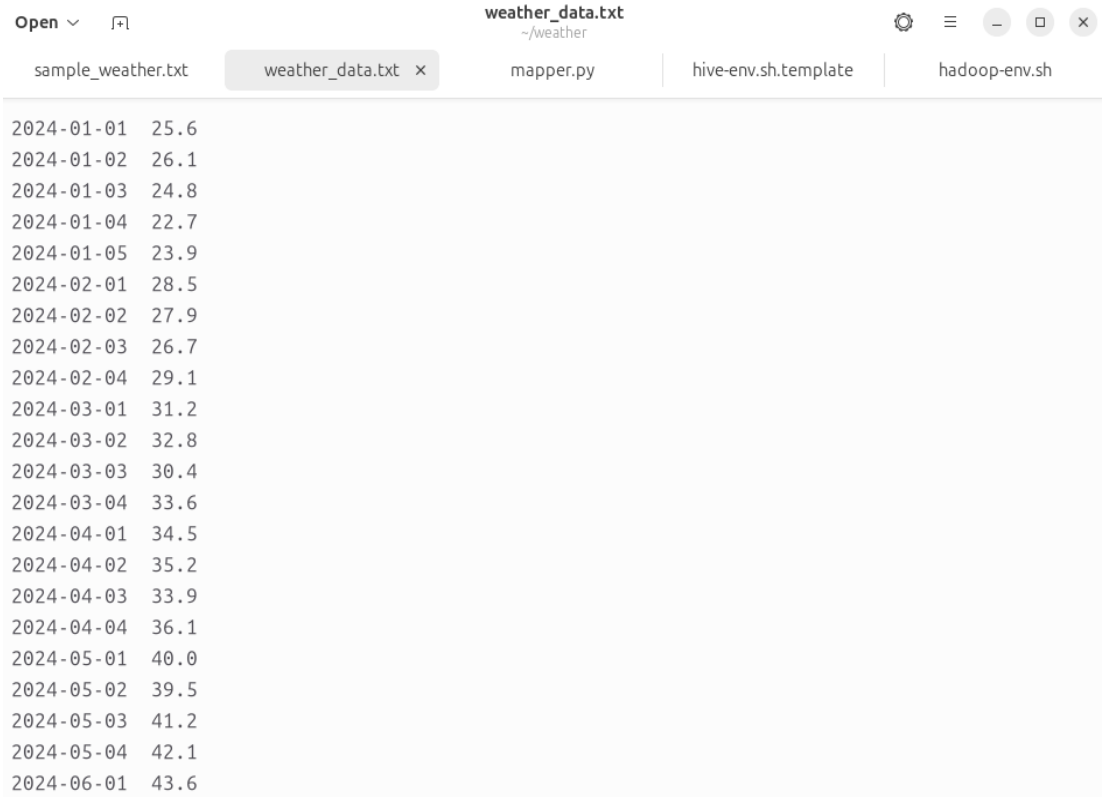
Procedure:

Step 1: Create Data File:

Create a file named "word_count_data.txt" and populate it with text data that you wish to analyse. Login with your hadoop user.

Download the dataset (weather data)

Output:



Step 2: Mapper Logic - mapper.py:

Create a file named "mapper.py" to implement the logic for the mapper. The mapper will read input data from STDIN, split lines into words, and output each word with its count.

```
nano mapper.py
# Copy and paste the mapper.py code
```

```
#!/usr/bin/env python

import sys

# input comes from STDIN (standard input)
# the mapper will get daily max temperature and group it by month. so output will be (month,dailymax_temperature)

for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()     # split the line into words     words = line.split()
    #See the README hosted on the weather website  which help us understand how each position represents a column     month = line[10:12]     daily_max = line[38:45]     daily_max = daily_max.strip()
    # increase counters     for word in words:
        # write the results to STDOUT (standard output);
        # what we output here will be go through the shuffle proess and then
        # be the input for the Reduce step, i.e. the input for reducer.py
        #
        # tab-delimited; month and daily max temperature as output
print ('%s\t%s' % (month ,daily_max))
```

.

Step 3: Reducer Logic - reducer.py:

Create a file named "reducer.py" to implement the logic for the reducer. The reducer will aggregate the occurrences of each word and generate the final output.

```
nano reducer.py
# Copy and paste the reducer.py code
```

reducer.py

```
#!/usr/bin/env python

from operator import itemgetter import sys
#reducer will get the input from stdid which will be a collection of key, value(Key=month , value= daily max temperature)
#reducer logic: will get all the daily max temperature for a month and find max temperature for the month
#shuffle will ensure that key are sorted(month)
```

```
current_month = None
current_max = 0 month =
None

# input comes from STDIN for
line in sys.stdin:
   # remove leading and trailing whitespace     line
= line.strip()
   # parse the input we got from mapper.py     month,
daily_max = line.split('\t', 1)

   # convert daily_max (currently a string) to float     try:
      daily_max = float(daily_max)     except
ValueError:
      # daily_max was not a number, so silently
      # ignore/discard this line
continue

   # this IF-switch only works because Hadoop shuffle process sorts map output
   # by key (here: month) before it is passed to the reducer
if current_month == month:         if daily_max > current_max:
current_max = daily_max     else:         if current_month:
          # write result to STDOUT
          print ('%s\t%s' % (current_month, current_max))
current_max = daily_max
      current_month = month

# output of the last month if current_month == month:
print ('%s\t%s' % (current_month, current_max))
```

Step 4: Prepare Hadoop Environment:

Start the Hadoop daemons and create a directory in HDFS to store your data.

```
start-all.sh
```

Step 6: Make Python Files Executable:

Give executable permissions to your mapper.py and reducer.py files.

```
chmod 777 mapper.py reducer.py
```

```
swathi@swathi-VirtualBox:~/da2$ hadoop jar $HADOOP_STREAMING -input /weatherdata/weather_data.txt -output /user/swathi/outputte -map
per ~/da2/mapper.py -reducer ~/da2/reducer.py
packageJobJar: [/tmp/hadoop-unjar8997754304788571910/] [] /tmp/streamjob523954984575776188.jar tmpDir=null
2024-09-19 18:25:55,425 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-19 18:25:55,649 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-19 18:25:56,096 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/swathi/.stag
ing/job_1726748824473_0007
2024-09-19 18:25:56,492 INFO mapred.FileInputFormat: Total input files to process : 1
2024-09-19 18:25:56,998 INFO mapreduce.JobSubmitter: number of splits:2
2024-09-19 18:25:57,191 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1726748824473_0007
2024-09-19 18:25:57,191 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-09-19 18:25:57,442 INFO conf.Configuration: resource-types.xml not found
2024-09-19 18:25:57,443 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-09-19 18:25:57,576 INFO impl.YarnClientImpl: Submitted application application_1726748824473_0007
2024-09-19 18:25:57,654 INFO mapreduce.Job: The url to track the job: http://swathi-VirtualBox:8088/proxy/application_1726748824473_
0007/
2024-09-19 18:25:57,663 INFO mapreduce.Job: Running job: job_1726748824473_0007
2024-09-19 18:26:05,883 INFO mapreduce.Job: Job job_1726748824473_0007 running in uber mode : false
2024-09-19 18:26:05,885 INFO mapreduce.Job:  map 0% reduce 0%
2024-09-19 18:26:15,099 INFO mapreduce.Job:  map 100% reduce 0%
2024-09-19 18:26:21,160 INFO mapreduce.Job:  map 100% reduce 100%
2024-09-19 18:26:22,178 INFO mapreduce.Job: Job job_1726748824473_0007 completed successfully
2024-09-19 18:26:22,336 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=56
                FILE: Number of bytes written=934326
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=309
                HDFS: Number of bytes written=8
                HDFS: Number of read operations=11
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
```

Step 7: Run the program using Hadoop Streaming:

Download the latest hadoop-streaming jar file and place it in a location you can easily access.

Then run the program using Hadoop Streaming.

```
hadoop fs -mkdir -p /weatherdata

hadoop fs -copyFromLocal /home/sx/Downloads/dataset.txt /weatherdata

hdfs dfs -ls /weatherdata

hadoop jar /home/sx/hadoop-3.2.3/share/hadoop/tools/lib/hadoop-streaming-3.2.3.jar \
  -input /weatherdata/dataset.txt \
  -output /weatherdata/output \
  -file "/home/sx/Downloads/mapper.py" \
  -mapper "python3 mapper.py" \
  -file "/home/sx/Downloads/reducer.py" \
  -reducer "python3 reducer.py"

hdfs dfs -text /weatherdata/output/* > /home/sx/Downloads/outputfile.txt
```

Step 8: Check Output:

```
swathi@swathi-VirtualBox:~/dalab/exp3$ hdfs dfs -cat /home/swathi/ex1output/part-00000
2021    33
2021    32
2021    28
2021    30
2021    25
swathi@swathi-VirtualBox:~/dalab/exp3$ hdfs dfs -rm -r /home/swathi/outputkl
Deleted /home/swathi/outputkl
swathi@swathi-VirtualBox:~/dalab/exp3$ cd ..
swathi@swathi-VirtualBox:~/dalab$ ls
exp2  exp3  exp4  exp5  exp6
```

Trash

Check the output of the program in the specified HDFS output directory.

hdfs dfs -text /weatherdata/output/* > /home/sx/Downloads/output/ /part-00000

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	swathi	supergroup	0 B	Sep 19 18:26	1	128 MB	_SUCCESS
-rw-r--r--	swathi	supergroup	8 B	Sep 19 18:26	1	128 MB	part-00000

Showing 1 to 2 of 2 entries

Previous 1 Next

After copy and paste the above output in your local file give the below command to remove the directory from hdfs : hadoop fs -rm -r /weatherdata/output

Result:

Thus, the program for weather dataset using Map Reduce has been executed successfully.