

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:**

The screenshot shows a code editor window with the title 'weather_data.txt' and a subtitle '~/.weather'. The editor contains a list of dates and temperatures, with the first tab labeled 'weather_data.txt' and a close button 'x'.

Date	Temperature
2024-01-01	25.6
2024-01-02	26.1
2024-01-03	24.8
2024-01-04	22.7
2024-01-05	23.9
2024-02-01	28.5
2024-02-02	27.9
2024-02-03	26.7
2024-02-04	29.1
2024-03-01	31.2
2024-03-02	32.8
2024-03-03	30.4
2024-03-04	33.6
2024-04-01	34.5
2024-04-02	35.2
2024-04-03	33.9
2024-04-04	36.1
2024-05-01	40.0
2024-05-02	39.5
2024-05-03	41.2
2024-05-04	42.1
2024-06-01	43.6

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,daily_max_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 process 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:

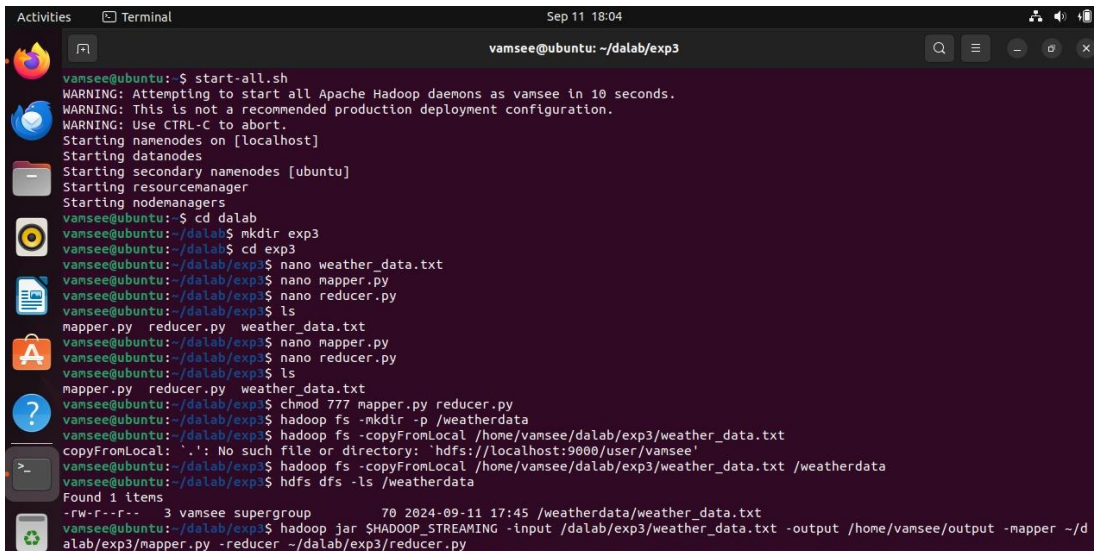
Start the Hadoop daemons and create a directory in HDFS to store your data.

```
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
```



```

vnmsee@ubuntu: ~/dalab/exp3
vnmsee@ubuntu:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as vnmsee in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [ubuntu]
Starting resourcemanager
Starting nodemanagers
vnmsee@ubuntu:~$ cd dalab
vnmsee@ubuntu:~/dalab$ mkdir exp3
vnmsee@ubuntu:~/dalab$ cd exp3
vnmsee@ubuntu:~/dalab/exp3$ nano weather_data.txt
vnmsee@ubuntu:~/dalab/exp3$ nano mapper.py
vnmsee@ubuntu:~/dalab/exp3$ nano reducer.py
vnmsee@ubuntu:~/dalab/exp3$ ls
mapper.py  reducer.py  weather_data.txt
vnmsee@ubuntu:~/dalab/exp3$ nano mapper.py
vnmsee@ubuntu:~/dalab/exp3$ nano reducer.py
vnmsee@ubuntu:~/dalab/exp3$ ls
mapper.py  reducer.py  weather_data.txt
vnmsee@ubuntu:~/dalab/exp3$ chmod 777 mapper.py reducer.py
vnmsee@ubuntu:~/dalab/exp3$ hadoop fs -mkdir -p /weatherdata
vnmsee@ubuntu:~/dalab/exp3$ hadoop fs -copyFromLocal /home/vnmsee/dalab/exp3/weather_data.txt
copyFromLocal: '.': No such file or directory: 'hdfs://localhost:9000/user/vnmsee'
vnmsee@ubuntu:~/dalab/exp3$ hadoop fs -copyFromLocal /home/vnmsee/dalab/exp3/weather_data.txt /weatherdata
vnmsee@ubuntu:~/dalab/exp3$ hdfs dfs -ls /weatherdata
Found 1 items
-rw-r--r-- 3 vnmsee supergroup 70 2024-09-11 17:45 /weatherdata/weather_data.txt
vnmsee@ubuntu:~/dalab/exp3$ hadoop jar $HADOOP_STREAMING -input /dalab/exp3/weather_data.txt -output /home/vnmsee/output -mapper ~/dalab/exp3/mapper.py -reducer ~/dalab/exp3/reducer.py

```

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
```

```

vamsee@ubuntu:~/dalab/exp3$ hadoop jar $HADOOP_STREAMING -input /dalab/exp3/weather_data.txt -output /user/vamsee/output -mapper ~/dalab/exp3/mapper.py -reducer ~/dalab/exp3/reducer.py
packageJobJar: [/tmp/hadoop-unjar8010354082854767480/] [] /tmp/streamjob4019098913476748699.jar tmpDir=null
2024-09-11 18:02:04,656 INFO client.DefaultNoHARMFaloverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-11 18:02:04,885 INFO client.DefaultNoHARMFaloverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-11 18:02:05,211 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/vamsee/.staging/job_1726056172778_0007
2024-09-11 18:02:05,649 INFO mapred.FileInputFormat: Total input files to process : 1
2024-09-11 18:02:06,853 INFO mapreduce.JobSubmitter: number of splits:2
2024-09-11 18:02:07,116 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1726056172778_0007
2024-09-11 18:02:07,116 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-09-11 18:02:07,417 INFO conf.Configuration: resource-types.xml not found
2024-09-11 18:02:07,417 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-09-11 18:02:09,874 INFO impl.YarnClientImpl: Submitted application application_1726056172778_0007
2024-09-11 18:02:09,956 INFO mapreduce.Job: The url to track the job: http://ubuntu.myguest.virtualbox.org:8088/proxy/application_1726056172778_0007/
2024-09-11 18:02:09,959 INFO mapreduce.Job: Running job: job_1726056172778_0007
2024-09-11 18:02:26,945 INFO mapreduce.Job: Job job_1726056172778_0007 running in uber mode : false
2024-09-11 18:02:26,984 INFO mapreduce.Job: map 0% reduce 0%
2024-09-11 18:02:35,652 INFO mapreduce.Job: map 100% reduce 0%
2024-09-11 18:02:42,755 INFO mapreduce.Job: map 100% reduce 100%
2024-09-11 18:02:43,806 INFO mapreduce.Job: Job job_1726056172778_0007 completed successfully
2024-09-11 18:02:44,669 INFO mapreduce.Job: Counters: 54

```

Step 8: Check Output:

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

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

```

      Bytes Read=105
      File Output Format Counters
      Bytes Written=8
2024-09-11 18:02:44,671 INFO streaming.StreamJob: Output directory: /user/vamsee/output
vamsee@ubuntu:~/dalab/exp3$ hdfs dfs -cat /user/vamsee/output/part-* | more
2021      33
vamsee@ubuntu:~/dalab/exp3$ █

```

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.