

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.

Output:

```

GNU nano 6.2 dataset.txt
26494 20240101 2.424 -147.51 64.97 -8.6 -15.7 -12.2 -13.5 0.0 0.00 C -13.2 -22.0 -19.5 -9999.0 -9999.0 -9999.0
26494 20240102 2.424 -147.51 64.97 -6.9 -10.1 -8.5 -8.3 0.0 0.01 C -10.9 -16.6 -12.8 -9999.0 -9999.0 -9999.0
26494 20240103 2.424 -147.51 64.97 -10.0 -15.7 -12.9 -13.5 0.0 0.00 C -14.4 -20.1 -17.4 -9999.0 -9999.0 -9999.0
26494 20240104 2.424 -147.51 64.97 -13.6 -16.4 -15.0 -15.6 0.2 0.03 C -14.7 -19.5 -16.4 -9999.0 -9999.0 -9999.0
26494 20240105 2.424 -147.51 64.97 -13.5 -20.5 -17.0 -17.3 0.0 0.00 C -16.7 -25.2 -22.0 -9999.0 -9999.0 -9999.0
26494 20240106 2.424 -147.51 64.97 -10.4 -21.7 -16.0 -17.7 0.0 0.00 C -16.1 -25.7 -22.9 -9999.0 -9999.0 -9999.0
26494 20240107 2.424 -147.51 64.97 -2.9 -10.7 -6.8 -5.8 0.0 0.01 C -6.2 -17.8 -9.2 -9999.0 -9999.0 -9999.0
26494 20240108 2.424 -147.51 64.97 -4.8 -12.5 -8.6 -8.0 5.5 0.00 C -6.2 -12.0 -8.0 -9999.0 -9999.0 -9999.0
26494 20240109 2.424 -147.51 64.97 -12.5 -20.0 -16.2 -15.8 0.2 0.00 C -12.0 -27.6 -16.9 -9999.0 -9999.0 -9999.0
26494 20240110 2.424 -147.51 64.97 -15.2 -20.9 -18.1 -17.6 0.0 0.00 C -16.6 -31.2 -22.3 -9999.0 -9999.0 -9999.0
26494 20240111 2.424 -147.51 64.97 -10.2 -17.1 -13.7 -13.6 0.0 0.04 C -15.7 -25.2 -19.9 -9999.0 -9999.0 -9999.0
26494 20240112 2.424 -147.51 64.97 -14.5 -19.3 -16.9 -17.2 0.0 0.00 C -18.4 -23.9 -21.8 -9999.0 -9999.0 -9999.0
26494 20240113 2.424 -147.51 64.97 -15.9 -20.9 -18.4 -18.5 0.0 0.01 C -16.5 -28.2 -23.4 -9999.0 -9999.0 -9999.0
26494 20240114 2.424 -147.51 64.97 -14.9 -18.8 -16.8 -17.2 3.1 0.02 C -14.3 -19.5 -16.1 -9999.0 -9999.0 -9999.0
26494 20240115 2.424 -147.51 64.97 -7.6 -15.2 -11.4 -11.8 2.6 0.02 C -7.8 -18.4 -12.2 -9999.0 -9999.0 -9999.0
26494 20240116 2.424 -147.51 64.97 -3.8 -9.8 -6.8 -7.4 1.8 0.00 C -4.9 -16.3 -8.2 -9999.0 -9999.0 -9999.0
26494 20240117 2.424 -147.51 64.97 -8.7 -16.2 -12.5 -12.5 0.0 0.13 C -14.0 -24.4 -17.5 -9999.0 -9999.0 -9999.0
26494 20240118 2.424 -147.51 64.97 -12.2 -17.3 -14.7 -14.7 0.0 0.18 C -11.8 -25.9 -19.4 -9999.0 -9999.0 -9999.0
26494 20240119 2.424 -147.51 64.97 -11.6 -17.4 -14.5 -15.3 0.0 0.01 C -11.7 -24.7 -20.5 -9999.0 -9999.0 -9999.0
26494 20240120 2.424 -147.51 64.97 -14.0 -21.1 -17.5 -18.6 0.0 0.03 C -18.4 -26.8 -23.4 -9999.0 -9999.0 -9999.0
26494 20240121 2.424 -147.51 64.97 -19.1 -27.0 -23.1 -22.2 0.0 0.05 C -22.8 -34.6 -27.2 -9999.0 -9999.0 -9999.0
26494 20240122 2.424 -147.51 64.97 -23.5 -29.0 -26.3 -26.4 0.0 0.05 C -30.6 -35.8 -33.5 -9999.0 -9999.0 -9999.0
26494 20240123 2.424 -147.51 64.97 -23.5 -31.7 -27.6 -26.9 0.0 0.22 C -28.1 -36.6 -31.7 -9999.0 -9999.0 -9999.0
26494 20240124 2.424 -147.51 64.97 -23.9 -33.6 -28.7 -28.1 0.0 0.23 C -24.4 -37.4 -30.8 -9999.0 -9999.0 -9999.0
26494 20240125 2.424 -147.51 64.97 -25.9 -31.0 -28.5 -28.7 0.0 0.07 C -30.8 -37.6 -34.8 -9999.0 -9999.0 -9999.0
26494 20240126 2.424 -147.51 64.97 -28.0 -36.9 -32.4 -33.2 0.0 0.03 C -34.7 -42.2 -39.2 -9999.0 -9999.0 -9999.0
26494 20240127 2.424 -147.51 64.97 -36.2 -40.4 -38.3 -38.5 0.0 0.26 C -38.7 -43.7 -41.9 -9999.0 -9999.0 -9999.0
26494 20240128 2.424 -147.51 64.97 -31.7 -38.3 -35.0 -34.9 1.4 0.12 C -28.5 -43.3 -34.4 -9999.0 -9999.0 -9999.0
26494 20240129 2.424 -147.51 64.97 -29.8 -33.7 -31.7 -31.9 0.7 0.00 C -26.9 -32.5 -28.9 -9999.0 -9999.0 -9999.0
26494 20240130 2.424 -147.51 64.97 -27.6 -32.6 -30.1 -29.5 0.0 0.08 C -25.4 -37.5 -30.6 -9999.0 -9999.0 -9999.0
26494 20240131 2.424 -147.51 64.97 -27.7 -32.6 -30.1 -30.2 0.0 0.15 C -29.2 -40.1 -37.0 -9999.0 -9999.0 -9999.0
26494 20240201 2.424 -147.51 64.97 -31.4 -36.6 -34.0 -34.1 0.0 0.24 C -36.4 -41.8 -39.8 -9999.0 -9999.0 -9999.0
26494 20240202 2.424 -147.51 64.97 -33.8 -38.8 -36.3 -36.1 0.0 0.33 C -36.9 -45.1 -42.2 -9999.0 -9999.0 -9999.0
26494 20240203 2.424 -147.51 64.97 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0 0.0 U -9999.0 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
26494 20240204 2.424 -147.51 64.97 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0 0.0 U -9999.0 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
26494 20240205 2.424 -147.51 64.97 -16.1 -20.2 -18.1 -17.9 0.0 0.65 C -18.2 -25.1 -22.2 -9999.0 -9999.0 -9999.0

```

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

```

GNU nano 6.2 mapper.py
#!/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 weathr 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

```

GNU nano 6.2 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

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.

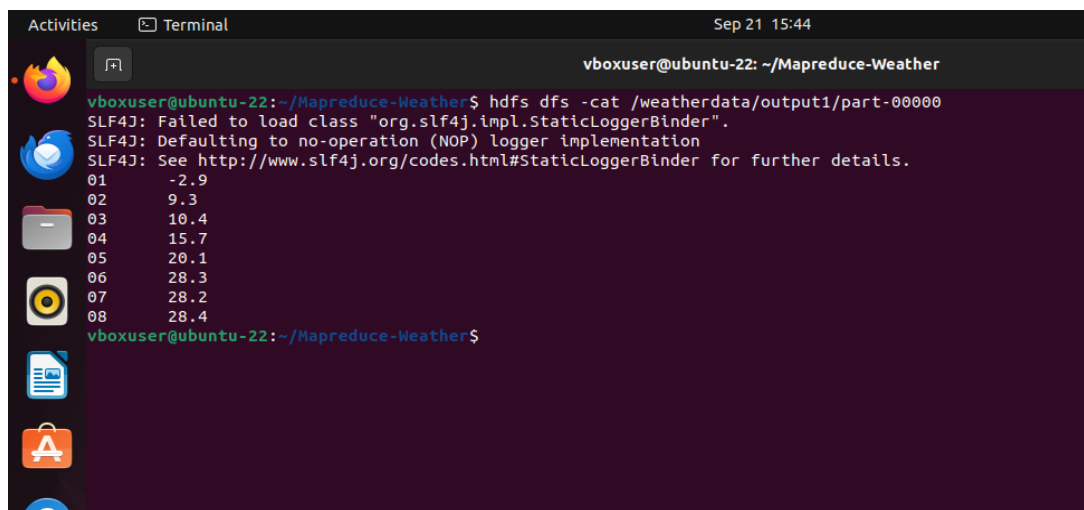
Then run the program using Hadoop Streaming.

```
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/vboxuser/Downloads/outputfile.txt
```

Step 8: Check Output: Check the output of the program in the specified HDFS output directory.



The screenshot shows a terminal window titled 'Terminal' with the date 'Sep 21 15:44'. The user is 'vboxuser@ubuntu-22: ~/Mapreduce-Weather'. The terminal displays the following commands and output:

```
vboxuser@ubuntu-22:~/Mapreduce-Weather$ hdfs dfs -cat /weatherdata/output1/part-00000  
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".  
SLF4J: Defaulting to no-operation (NOP) logger implementation  
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.  
01      -2.9  
02      9.3  
03      10.4  
04      15.7  
05      20.1  
06      28.3  
07      28.2  
08      28.4  
vboxuser@ubuntu-22:~/Mapreduce-Weather$
```

Step 9: The result in the browser is as follows:

The screenshot shows a web browser window with the address bar displaying 'localhost:9870/explorer.html#/weatherdata/output1'. The main content area is titled 'File information - part-00000'. It features a green header bar with 'Block information --' and a dropdown menu showing 'Block 0'. Below this, the following details are listed: Block ID: 1073741917, Block Pool ID: BP-305058674-127.0.1.1-1726117174333, Generation Stamp: 1093, Size: 63, and Availability: ubuntu-22.4.myguest.virtualbox.org. A 'File contents' section is also visible, displaying a list of data points.

File contents
01 -2.9
02 9.3
03 10.4
04 15.7
05 20.1
06 28.3
07 28.2
08 28.4

RESULT:

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