

Exp.No: 3

Map Reduce program to process a weather dataset

AIM:

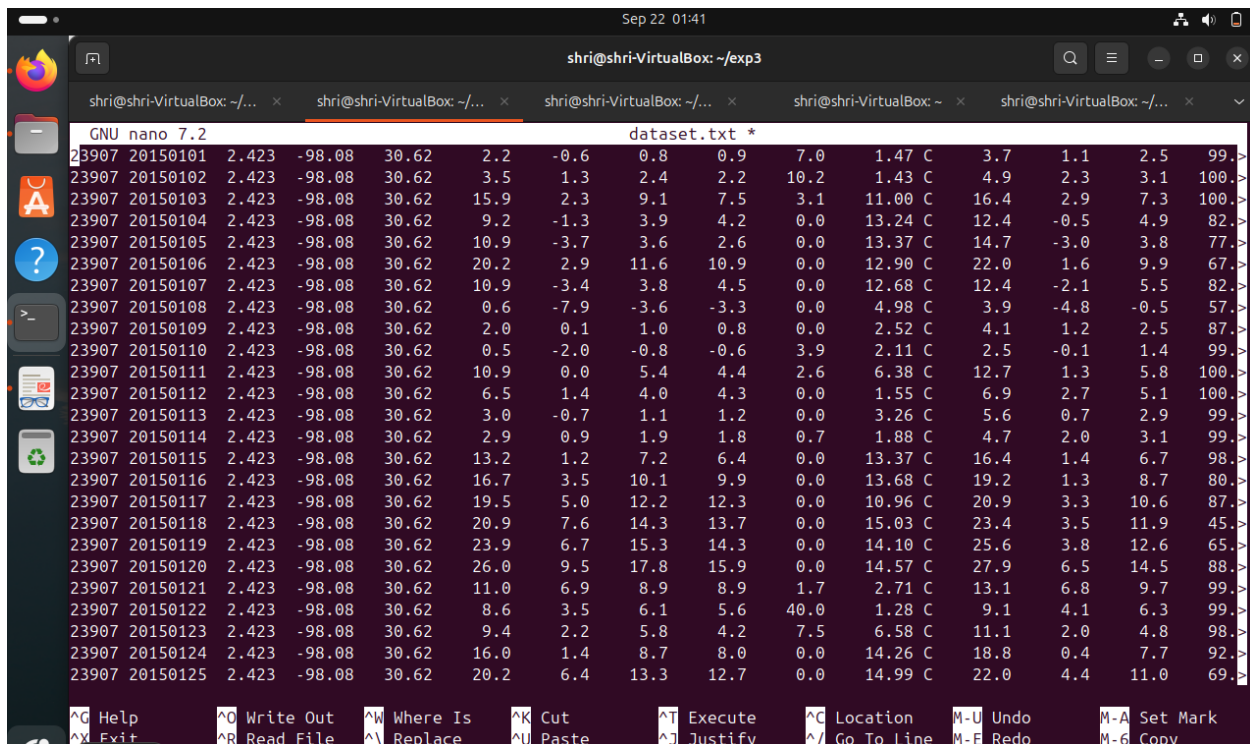
To implement MapReduce program to process a weather dataset.

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:

dataset.txt



```

GNU nano 7.2 dataset.txt *
23907 20150101 2.423 -98.08 30.62 2.2 -0.6 0.8 0.9 7.0 1.47 C 3.7 1.1 2.5 99.
23907 20150102 2.423 -98.08 30.62 3.5 1.3 2.4 2.2 10.2 1.43 C 4.9 2.3 3.1 100.
23907 20150103 2.423 -98.08 30.62 15.9 2.3 9.1 7.5 3.1 11.00 C 16.4 2.9 7.3 100.
23907 20150104 2.423 -98.08 30.62 9.2 -1.3 3.9 4.2 0.0 13.24 C 12.4 -0.5 4.9 82.
23907 20150105 2.423 -98.08 30.62 10.9 -3.7 3.6 2.6 0.0 13.37 C 14.7 -3.0 3.8 77.
23907 20150106 2.423 -98.08 30.62 20.2 2.9 11.6 10.9 0.0 12.90 C 22.0 1.6 9.9 67.
23907 20150107 2.423 -98.08 30.62 10.9 -3.4 3.8 4.5 0.0 12.68 C 12.4 -2.1 5.5 82.
23907 20150108 2.423 -98.08 30.62 0.6 -7.9 -3.6 -3.3 0.0 4.98 C 3.9 -4.8 -0.5 57.
23907 20150109 2.423 -98.08 30.62 2.0 0.1 1.0 0.8 0.0 2.52 C 4.1 1.2 2.5 87.
23907 20150110 2.423 -98.08 30.62 0.5 -2.0 -0.8 -0.6 3.9 2.11 C 2.5 -0.1 1.4 99.
23907 20150111 2.423 -98.08 30.62 10.9 0.0 5.4 4.4 2.6 6.38 C 12.7 1.3 5.8 100.
23907 20150112 2.423 -98.08 30.62 6.5 1.4 4.0 4.3 0.0 1.55 C 6.9 2.7 5.1 100.
23907 20150113 2.423 -98.08 30.62 3.0 -0.7 1.1 1.2 0.0 3.26 C 5.6 0.7 2.9 99.
23907 20150114 2.423 -98.08 30.62 2.9 0.9 1.9 1.8 0.7 1.88 C 4.7 2.0 3.1 99.
23907 20150115 2.423 -98.08 30.62 13.2 1.2 7.2 6.4 0.0 13.37 C 16.4 1.4 6.7 98.
23907 20150116 2.423 -98.08 30.62 16.7 3.5 10.1 9.9 0.0 13.68 C 19.2 1.3 8.7 80.
23907 20150117 2.423 -98.08 30.62 19.5 5.0 12.2 12.3 0.0 10.96 C 20.9 3.3 10.6 87.
23907 20150118 2.423 -98.08 30.62 20.9 7.6 14.3 13.7 0.0 15.03 C 23.4 3.5 11.9 45.
23907 20150119 2.423 -98.08 30.62 23.9 6.7 15.3 14.3 0.0 14.10 C 25.6 3.8 12.6 65.
23907 20150120 2.423 -98.08 30.62 26.0 9.5 17.8 15.9 0.0 14.57 C 27.9 6.5 14.5 88.
23907 20150121 2.423 -98.08 30.62 11.0 6.9 8.9 8.9 1.7 2.71 C 13.1 6.8 9.7 99.
23907 20150122 2.423 -98.08 30.62 8.6 3.5 6.1 5.6 40.0 1.28 C 9.1 4.1 6.3 99.
23907 20150123 2.423 -98.08 30.62 9.4 2.2 5.8 4.2 7.5 6.58 C 11.1 2.0 4.8 98.
23907 20150124 2.423 -98.08 30.62 16.0 1.4 8.7 8.0 0.0 14.26 C 18.8 0.4 7.7 92.
23907 20150125 2.423 -98.08 30.62 20.2 6.4 13.3 12.7 0.0 14.99 C 22.0 4.4 11.0 69.
  
```

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 7.2 mapper.py
#!/usr/bin/env python3
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()

    # Extract month and daily max temperature from the line
    # Assume the line format is fixed as per the example in the README
    month = line[10:12]
    daily_max = line[38:45].strip()

    # Output the results to STDOUT (standard output)
    # This output will be used as input for the reducer step
    print(f'{month}\t{daily_max}')

Read 19 lines
Help Exit Write Out Read File Where Is Replace Cut Paste Execute Justify Location Go To Line Undo Redo Set Mark Copy

```

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 7.2 reducer.py
#!/usr/bin/env python3
from operator import itemgetter
import sys

# Reducer will get input from STDIN, which will be a collection of key-value pairs (month, daily_max_temperature)
# Reducer logic: For each month, find the maximum daily max temperature

current_month = None
current_max = float('-inf') # Initialize with a very low value to handle all temperatures
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
    try:
        month, daily_max = line.split('\t', 1)
        daily_max = float(daily_max)
    except ValueError:
        # Skip lines with invalid data
        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 month == current_month:
        if daily_max > current_max:
            current_max = daily_max
    else:
        # New month, reset current_max
        current_max = float('-inf')
        current_month = month

    # Print the final output for each month
    if month != None:
        print(f'{month}\t{current_max}')

Read 39 lines
Help Exit Write Out Read File Where Is Replace Cut Paste Execute Justify Location Go To Line Undo Redo Set Mark Copy

```

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.

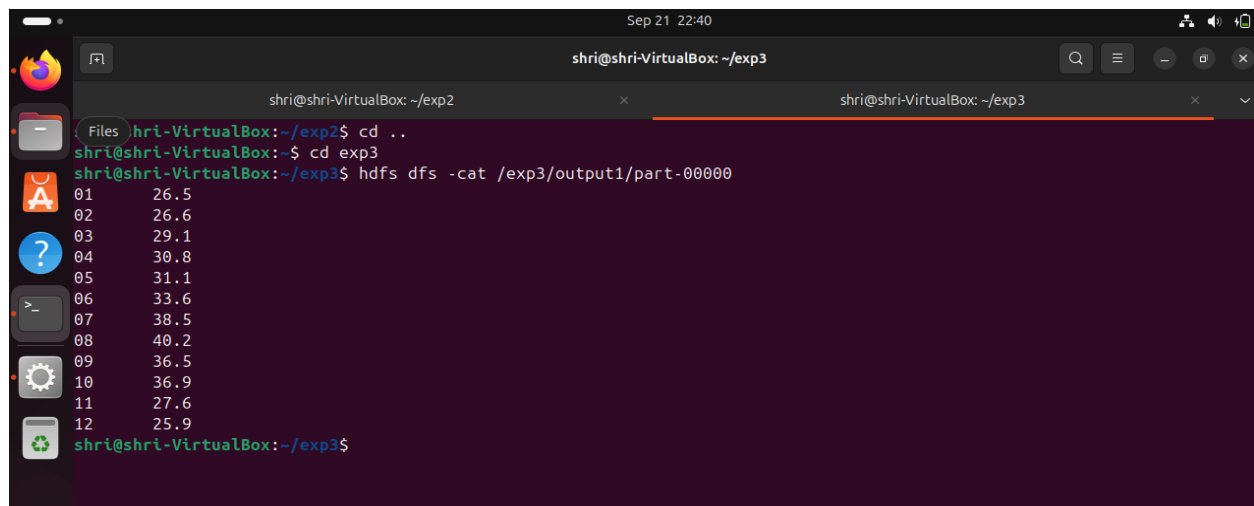
Then run the program using Hadoop Streaming.

```
hadoop fs -mkdir -p /weatherdata hadoop fs -copyFromLocal  
/home/shri@shri/Downloads/dataset.txt /weatherdata
```

```
hdfs dfs -ls /weatherdata hadoop jar /home/shri@shri/hadoop-  
3.2.3/share/hadoop/tools/lib/hadoop-streaming-3.2.3.jar \ -input  
/weatherdata/dataset.txt \ -output /weatherdata/output \ -file "/home/shri@shri  
/Downloads/mapper.py" \ -mapper "python3 mapper.py" \ -file "/home/shri@shri  
/Downloads/reducer.py" \ -reducer "python3 reducer.py"
```

```
hdfs dfs -text /weatherdata/output/* > /home/shri@shri/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 'shri@shri-VirtualBox: ~/exp3'. The user has navigated to the 'exp3' directory and executed the command 'hdfs dfs -cat /exp3/output1/part-00000'. The output displays a list of 12 lines, each containing an index (01 to 12) and a temperature value (26.5 to 25.9). The terminal window also shows the file manager and application dock on the left side.

```
shri@shri-VirtualBox: ~/exp2  
shri@shri-VirtualBox: ~/exp3  
shri@shri-VirtualBox: ~/exp3$ hdfs dfs -cat /exp3/output1/part-00000  
01      26.5  
02      26.6  
03      29.1  
04      30.8  
05      31.1  
06      33.6  
07      38.5  
08      40.2  
09      36.5  
10      36.9  
11      27.6  
12      25.9  
shri@shri-VirtualBox: ~/exp3$
```

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#/exp3/output1`. The main content area is titled "Block information -- Block 0" and displays the following details:

- Block ID: 1073741828
- Block Pool ID: BP-1260272768-127.0.1.1-1725100397044
- Generation Stamp: 1004
- Size: 96
- Availability:
 - shri-VirtualBox

Below the block information, the "File contents" section displays a list of 12 rows of data:

01	26.5
02	26.6
03	29.1
04	30.8
05	31.1
06	33.6
07	38.5
08	40.2
09	36.5
10	36.9
11	27.6
12	25.9

RESULT:

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