EXP 2: Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm.

AIM:

To run a basic Word Count MapReduce program.

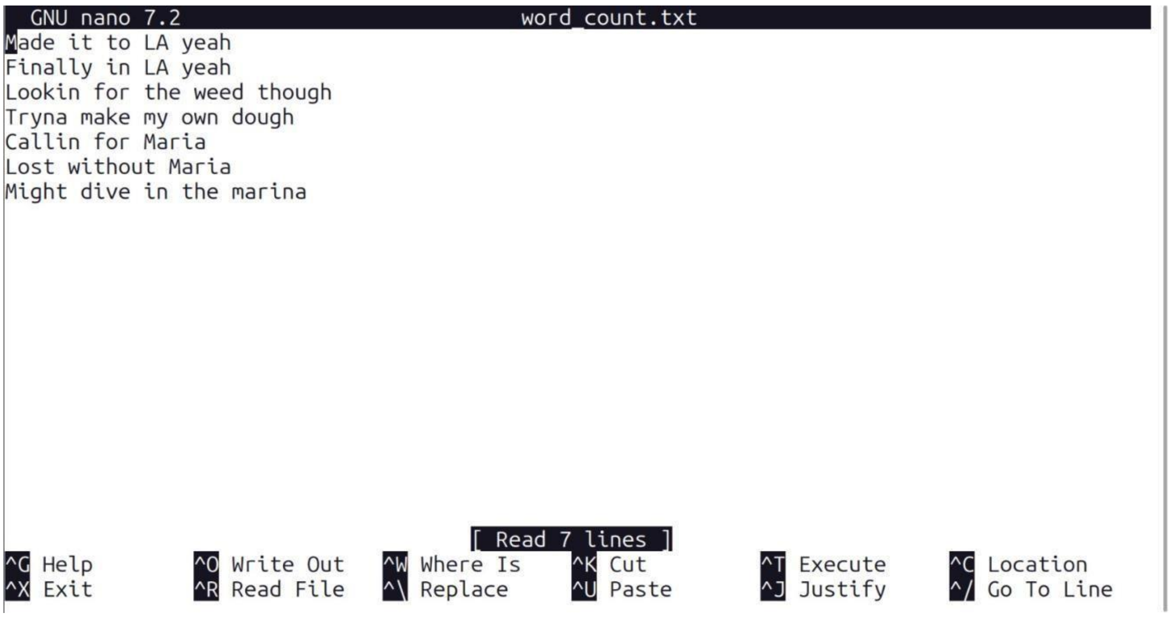
Procedure:

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.

```
nano word_count.txt
```

Output: Type the below content in word_count.txt



```
  GNU nano 7.2                         word_count.txt
Made it to LA yeah
Finally in LA yeah
Lookin for the weed though
Tryna make my own dough
Callin for Maria
Lost without Maria
Might dive in the marina

                                   [ Read 7 lines ]
^G Help          ^O Write Out    ^W Where Is     ^K Cut          ^T Execute      ^C Location
^X Exit          ^R Read File    ^\ Replace      ^U Paste        ^J Justify      ^/ Go To Line
```

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 python3
# import sys because we need to read and write data to STDIN and STDOUT #!/usr/bin/python3
import sys
for line in sys.stdin:
    line = line.strip()     # remove leading and trailing whitespace
    words = line.split() # split the line into words
    for word in words:
        print( '%s\t%s' % (word, 1))
         .
```

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/python3
from operator import itemgetter
import sys
current_word = None
current_count = 0
word = None
for line in sys.stdin:
    line = line.strip()
    word, count = line.split('\t', 1)
    try:
        count = int(count) except
    ValueError:
        continue
    if current_word == word:
        current_count += count
    else:
        if current_word:
            print( '%s\t%s' % (current_word, current_count))
        current_count = count
        current_word = word if
current_word == word:
    print( '%s\t%s' % (current_word, current_count))
```

Step 4: Prepare Hadoop Environment:

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

```
start-all.sh
hdfsdfs -mkdir /word_count_in_python
hdfsdfs -copyFromLocal /path/to/word_count.txt/word_count_in_python
```

Step 6: Make Python Files Executable:

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

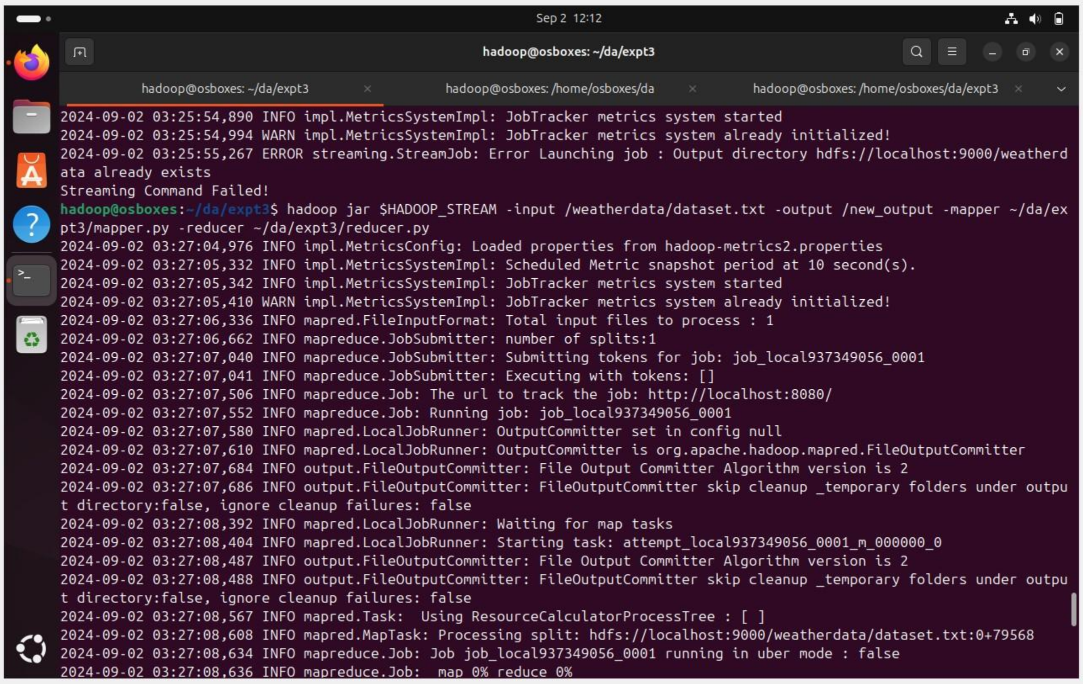
```
chmod 777 mapper.py reducer.py
```

Step 7: Run Word Count using Hadoop Streaming:

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

Then run the Word Count program using Hadoop Streaming.

```
hadoop jar /path/to/hadoop-streaming-3.3.6.jar \
    -input /word_count_in_python/word_count_data.txt \
    -output /word_count_in_python/new_output \
    -mapper /path/to/mapper.py \
    -reducer /path/to/reducer.py
```


Sep 2 12:12
hadoop@osboxes: ~/da/expt3
hadoop@osboxes: ~/da/expt3
hadoop@osboxes: /home/osboxes/da
hadoop@osboxes: /home/osboxes/da/expt3

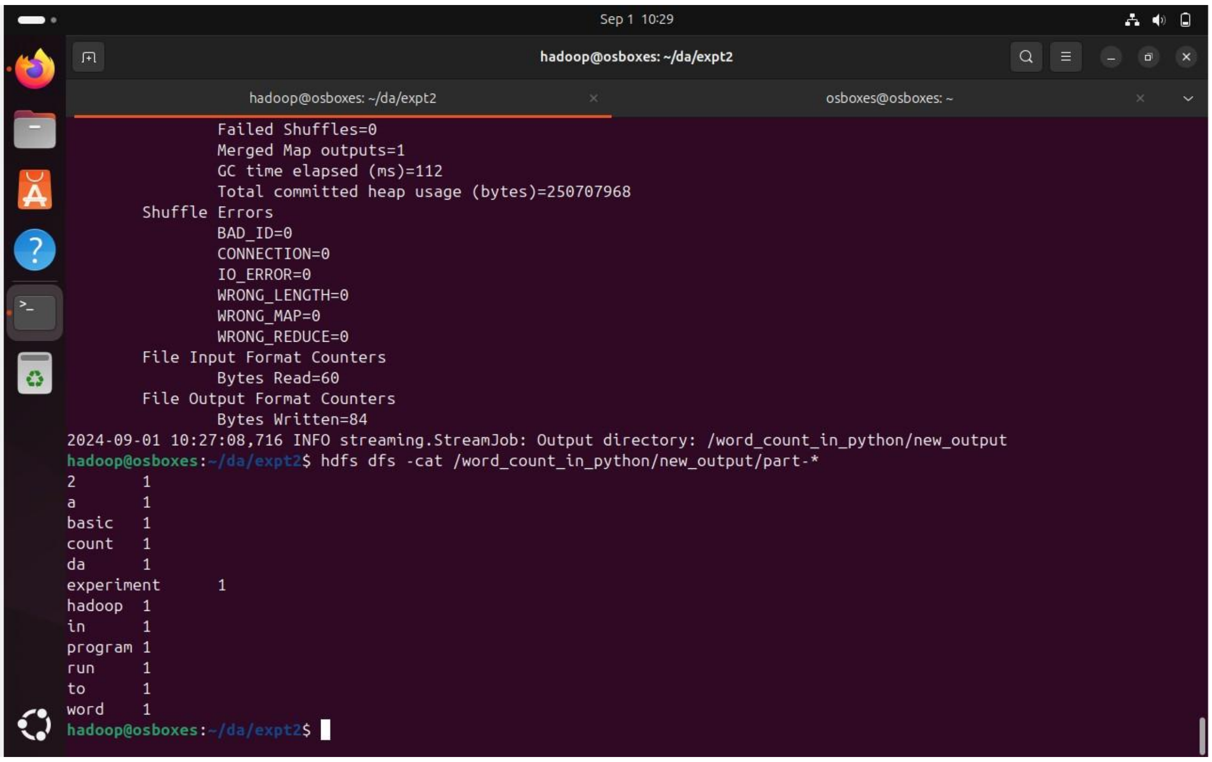
```
2024-09-02 03:25:54,890 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-09-02 03:25:54,994 WARN impl.MetricsSystemImpl: JobTracker metrics system already initialized!
2024-09-02 03:25:55,267 ERROR streaming.StreamJob: Error Launching job : Output directory hdfs://localhost:9000/weatherd
ata already exists
Streaming Command Failed!
hadoop@osboxes:~/da/expt3$ hadoop jar $HADOOP_STREAM -input /weatherdata/dataset.txt -output /new_output -mapper ~/da/ex
pt3/mapper.py -reducer ~/da/expt3/reducer.py
2024-09-02 03:27:04,976 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-09-02 03:27:05,332 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-09-02 03:27:05,342 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-09-02 03:27:05,410 WARN impl.MetricsSystemImpl: JobTracker metrics system already initialized!
2024-09-02 03:27:06,336 INFO mapred.FileInputFormat: Total input files to process : 1
2024-09-02 03:27:06,662 INFO mapreduce.JobSubmitter: number of splits:1
2024-09-02 03:27:07,040 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local937349056_0001
2024-09-02 03:27:07,041 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-09-02 03:27:07,506 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2024-09-02 03:27:07,552 INFO mapreduce.Job: Running job: job_local937349056_0001
2024-09-02 03:27:07,580 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2024-09-02 03:27:07,610 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter
2024-09-02 03:27:07,684 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-09-02 03:27:07,686 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under outpu
t directory:false, ignore cleanup failures: false
2024-09-02 03:27:08,392 INFO mapred.LocalJobRunner: Waiting for map tasks
2024-09-02 03:27:08,404 INFO mapred.LocalJobRunner: Starting task: attempt_local937349056_0001_m_000000_0
2024-09-02 03:27:08,487 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-09-02 03:27:08,488 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under outpu
t directory:false, ignore cleanup failures: false
2024-09-02 03:27:08,567 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2024-09-02 03:27:08,608 INFO mapred.MapTask: Processing split: hdfs://localhost:9000/weatherdata/dataset.txt:0+79568
2024-09-02 03:27:08,634 INFO mapreduce.Job: Job job_local937349056_0001 running in uber mode : false
2024-09-02 03:27:08,636 INFO mapreduce.Job:  map 0% reduce 0%
```

Step 8: Check Output:

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

```
hdfs dfs -cat /word_count_in_python/new_output/part-00000
```

[Type here]


Sep 1 10:29

hadoop@osboxes: ~/da/expt2

hadoop@osboxes: ~/da/expt2

osboxes@osboxes: ~

```
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=112
                Total committed heap usage (bytes)=250707968
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=60
        File Output Format Counters
                Bytes Written=84
2024-09-01 10:27:08,716 INFO streaming.StreamJob: Output directory: /word_count_in_python/new_output
hadoop@osboxes:~/da/expt2$ hdfs dfs -cat /word_count_in_python/new_output/part-*
2       1
a       1
basic   1
count   1
da      1
experiment      1
hadoop  1
in      1
program 1
run     1
to      1
word    1
hadoop@osboxes:~/da/expt2$ 
```