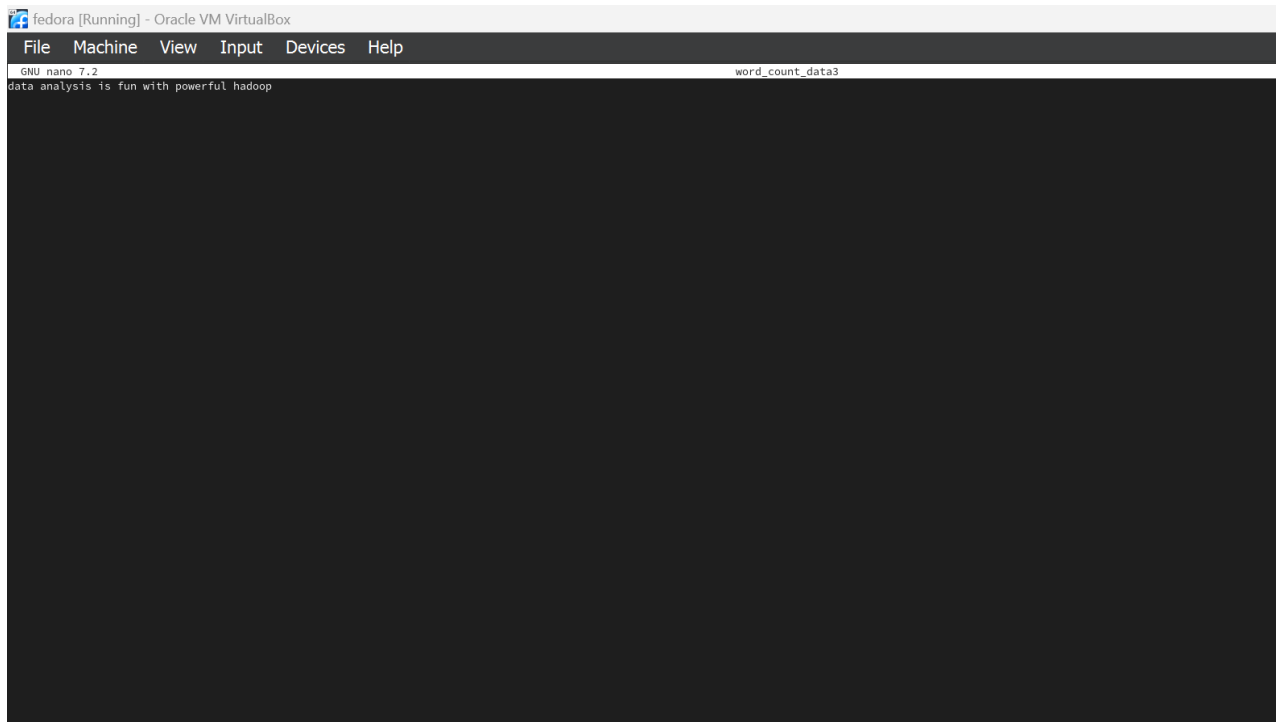


**Exp No: 2****Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm****Aim:**

To Run a basic Word Count MapReduce program to understand Map Reduce Paradigm.

**Procedure:****Step 1: Create Data File:**

Create a file named "word\_count\_data.txt" and populate it with text data that you wish to analyze. Login with your Hadoop user.

**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:
```

```
    nano word_count.txt 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 5: Make Python Files Executable:**

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

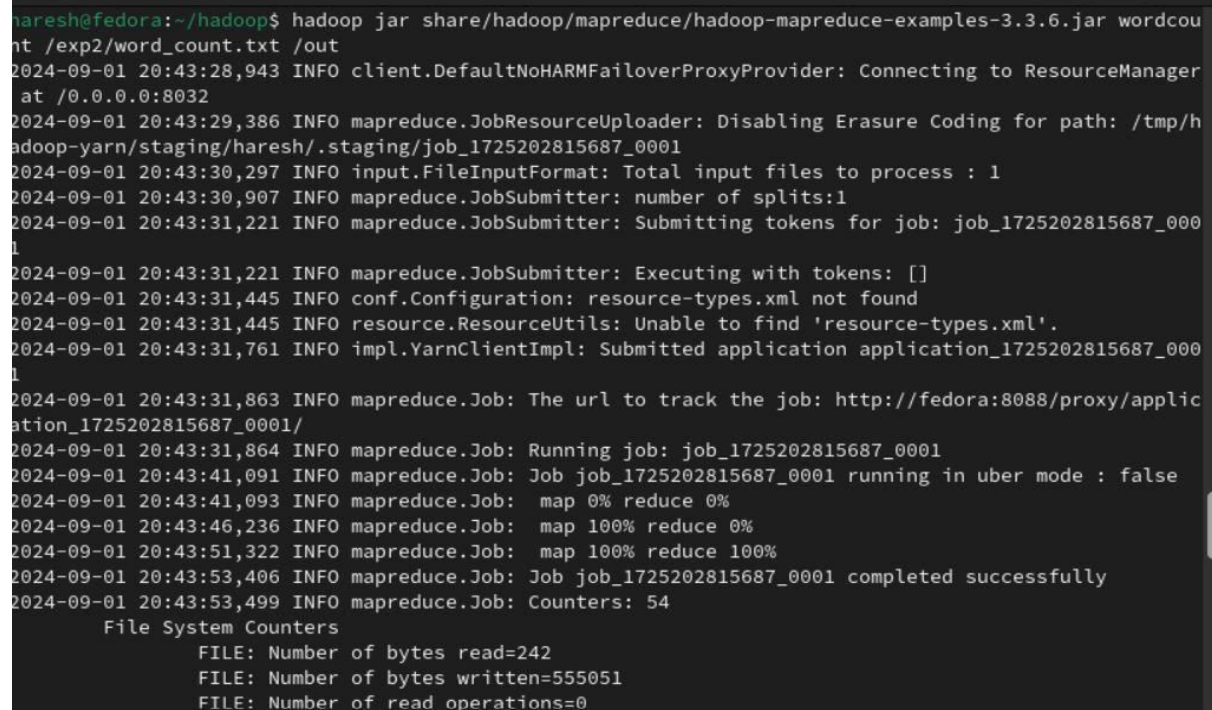
```
chmod 777 mapper.py reducer.py
```

**Step 6: 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
```



```
haresh@fedora:~/hadoop$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar wordcount /exp2/word_count.txt /out
2024-09-01 20:43:28,943 INFO client.DefaultNoHARMAFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-01 20:43:29,386 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/haresh/.staging/job_1725202815687_0001
2024-09-01 20:43:30,297 INFO input.FileInputFormat: Total input files to process : 1
2024-09-01 20:43:30,907 INFO mapreduce.JobSubmitter: number of splits:1
2024-09-01 20:43:31,221 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1725202815687_0001
2024-09-01 20:43:31,221 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-09-01 20:43:31,445 INFO conf.Configuration: resource-types.xml not found
2024-09-01 20:43:31,445 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-09-01 20:43:31,761 INFO impl.YarnClientImpl: Submitted application application_1725202815687_0001
2024-09-01 20:43:31,863 INFO mapreduce.Job: The url to track the job: http://fedora:8088/proxy/application_1725202815687_0001/
2024-09-01 20:43:31,864 INFO mapreduce.Job: Running job: job_1725202815687_0001
2024-09-01 20:43:41,091 INFO mapreduce.Job: Job job_1725202815687_0001 running in uber mode : false
2024-09-01 20:43:41,093 INFO mapreduce.Job: map 0% reduce 0%
2024-09-01 20:43:46,236 INFO mapreduce.Job: map 100% reduce 0%
2024-09-01 20:43:51,322 INFO mapreduce.Job: map 100% reduce 100%
2024-09-01 20:43:53,406 INFO mapreduce.Job: Job job_1725202815687_0001 completed successfully
2024-09-01 20:43:53,499 INFO mapreduce.Job: Counters: 54
    File System Counters
        FILE: Number of bytes read=242
        FILE: Number of bytes written=555051
        FILE: Number of read operations=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
```

```

Sep 12 4:59 AM
jananiarumugam@fedora:~/hadoop/sbin

Reduce input groups=7
Reduce shuffle bytes=111
Reduce input records=10
Reduce output records=7
Spilled Records=20
Shuffled Maps=2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=352
CPU time spent (ms)=2120
Physical memory (bytes) snapshot=815423468
Virtual memory (bytes) snapshot=7075645052
Total committed heap usage (bytes)=637809920
Peak Map Physical memory (bytes)=332197888
Peak Map Virtual memory (bytes)=2568524288
Peak Reduce Physical memory (bytes)=188485632
Peak Reduce Virtual memory (bytes)=2558735488

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=89

File Output Format Counters
Bytes Written=96

2024-09-12 04:56:25,585 INFO streaming.StreamJob: Output directory: /word_count_in_python/new_output_2024
jananiarumugam@fedora:~/hadoop/sbin$ hdfs dfs -ls /word_count_in_python/new_output_2024
Found 2 items
-rw-r--r-- 1 jananiarumugam supergroup          0 2024-09-12 04:56 /word_count_in_python/new_output_2024/_SUCCESS
-rw-r--r-- 1 jananiarumugam supergroup    56 2024-09-12 04:56 /word_count_in_python/new_output_2024/part-00000
jananiarumugam@fedora:~/hadoop/sbin$ hdfs dfs -cat /word_count_in_python/new_output_2024/part-0
Data      1
hadoop   1
analysis 1
fun       1
is        2
powerful 1
with      1
jananiarumugam@fedora:~/hadoop/sbin$

```

## Result:

Thus, the program for basic Word Count Map Reduce has been executed successfully.