Exp No: 2

Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm

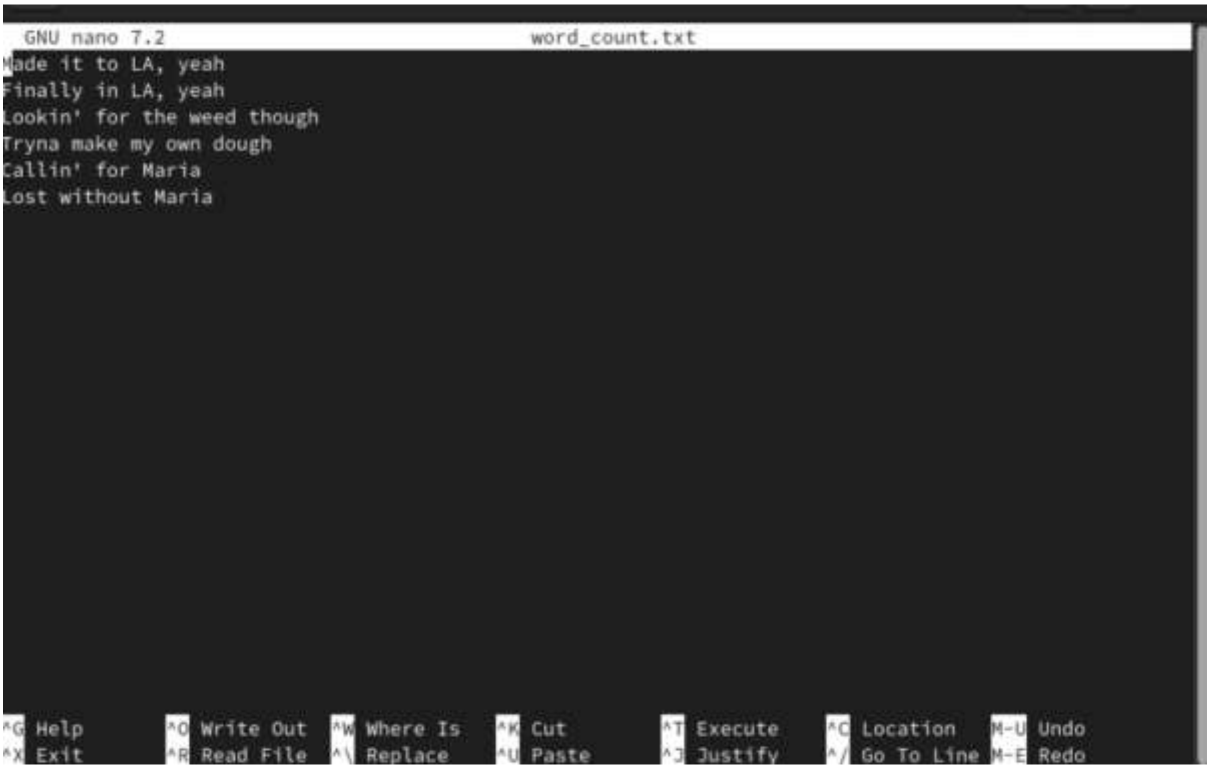
Aim:

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.



```
  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
```

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
```

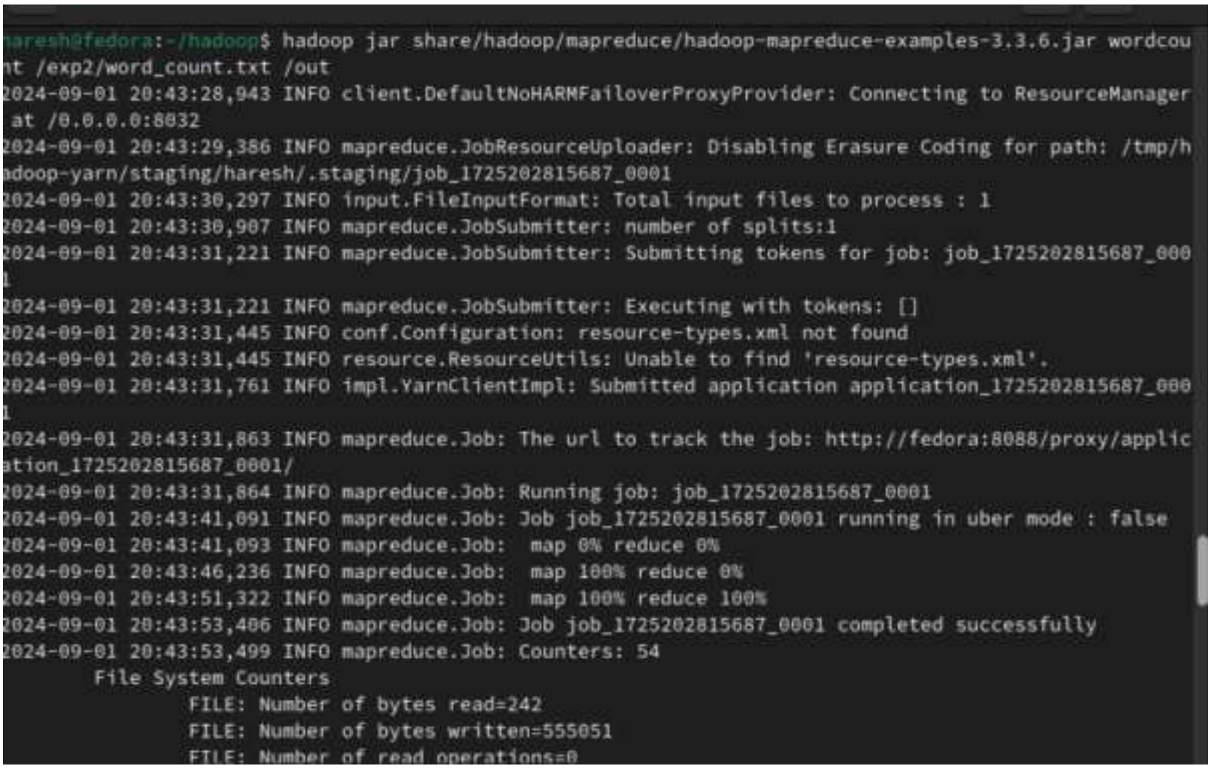
Step 6: Run Word Count using Hadoop Streaming:

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
```

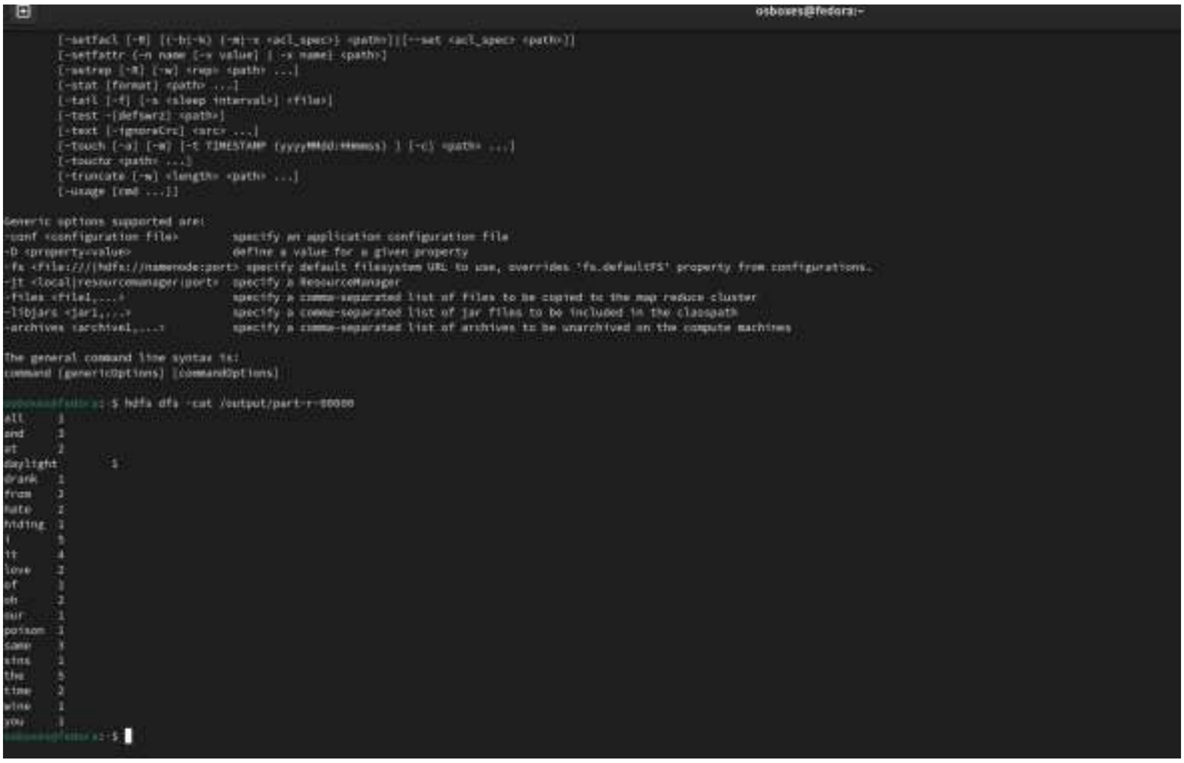


```
haresh@fedora:~/hadoop$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar wordcou
nt /exp2/word_count.txt /out
2024-09-01 20:43:28,943 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
 at /0.0.0.0:8032
2024-09-01 20:43:29,386 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/h
adoop-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_000
1
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_000
1
2024-09-01 20:43:31,863 INFO mapreduce.Job: The url to track the job: http://fedora:8088/proxy/applic
ation_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
```



```
        [-setfacl [-R] [{-b|-k} {-m|-x <acl_spec>} <path>]|[--set <acl_spec> <path>]]
        [-setfattr {-n name [-v value] | -x name} <path>]
        [-setrep [-R] [-w] <rep> <path> ...]
        [-stat [format] <path> ...]
        [-tail [-f] [-s <sleep interval>] <file>]
        [-test -[defswrz] <path>]
        [-text [-ignoreCrc] <src> ...]
        [-touch [-a] [-m] [-t TIMESTAMP (yyyyMMdd:HHmmss) ] [-c] <path> ...]
        [-touchz <path> ...]
        [-truncate [-w] <length> <path> ...]
        [-usage [cmd ...]]

Generic options supported are:
-conf <configuration file>        specify an application configuration file
-D <property=value>               define a value for a given property
-fs <file:///|hdfs://namenode:port> specify default filesystem URL to use, overrides 'fs.defaultFS' property from configurations.
-jt <local|resourcemanager:port>  specify a ResourceManager
-files <file1,...>                specify a comma-separated list of files to be copied to the map reduce cluster
-libjars <jar1,...>               specify a comma-separated list of jar files to be included in the classpath
-archives <archive1,...>          specify a comma-separated list of archives to be unarchived on the compute machines

The general command line syntax is:
command [genericOptions] [commandOptions]

osboxes@fedora:~$ hdfs dfs -cat /output/part-r-00000
all     1
and     3
at      2
daylight        1
drank   1
from    3
hate    2
hiding  1
i       5
it      4
love    2
of      1
oh      2
our     1
poison  1
same    3
sins    1
the     5
time    2
wine    1
you     1
osboxes@fedora:~$ 
```

Result:

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