HDFS and MapReduce

1.
Section 1: Access the cluster
ssh tdend2@cscluster.uis.edu

```
tdend2@node00:~
PS C:\Users\Swathi> ssh tdend2@cscluster.uis.edu
tdend2@cscluster.uis.edu's password:
Last login: Sat Aug 31 18:30:24 2024 from 192.168.23.148
```

Hadoop Distributed File System (HDFS)
a. Section 2.2.1

As data grows, a single physical machine gets saturated with its storage capacity. This growth drives the need to partition your data across separate machines. This type of file system that manages data storage across a network of machines is called a Distributed File System. Hadoop Distributed File Systems (HDFS) is a core component of Apache Hadoop and is designed to store large files with streaming data access patterns running on clusters of commodity hardware.

Checked the Hadoop version. The cluster has Cloudera Hadoop version 6.3.2 (CDH 6.3.2), which is based on Hadoop 3.0.

```
tdend2@node00:~
[tdend2@node00 ~]$  hadoop version
Hadoop 3.0.0-cdh6.3.2
Source code repository http://github.com/cloudera/hadoop -r 9aff20de3b5ecccf3c19d57f71b214fb4d37
Compiled by jenkins on 2019-11-08T13:49Z
Compiled with protoc 2.5.0
From source with checksum f539c87da37534aad732f2a7ddcc59
This command was run using /opt/cloudera/parcels/CDH-6.3.2-1.cdh6.3.2.p0.1605554/jars/hadoop-com
[tdend2@node00 ~]$
```

2.2 Loading data to HDFS
Loaded a simple dataset into HDFS using the command-line interface. It let's to perform tasks like creating directories, navigating file systems, and uploading files to HDFS. So created a directory and then loaded a file as follows:
into HDFS.
2.2.1 Create a folder in HDFS
We will create a 'WordCount' named directory in HDFS using the mkdir command. It creates the directory in HDFS if it does not already exist. Note: If the directory already exists in HDFS, we will get an error message that the file already exists.
Before creating a folder in HDFS, let's create a course directory in the local Linux filesystem:
mkdir CSC534BDA
Creating a folder in HDFS is similar to creating a folder in a Linux file system.
We can use *hadoop fs -mkdir /path/directoryname*. Just add 'hadoop fs -' in front of the Linux command.
hadoop fs -mkdir WordCount

Using the ls command, we can check for the directories in HDFS. The Hadoop fs shell command ls displays a list of the contents of a directory specified in the path provided by the user. It shows the name, permissions, owner, size, and modification date for each file or directory in the specified directory.

hadoop fs -ls

```
tdend2@node00:~
[tdend2@node00 ~]$  mkdir CSC534BDA
[tdend2@node00 ~]$ hadoop fs -mkdir WordCount
[tdend2@node00 ~]$  hadoop fs -ls
Found 1 items
drwxr-xr-x   - tdend2 hadoop          0 2024-09-07 12:50 WordCount
[tdend2@node00 ~]$
```

To list files/directories inside the WordCount directory: *hadoop fs -ls WordCount*

```
tdend2@node00:~
[tdend2@node00 ~]$  hadoop fs -ls WordCount
[tdend2@node00 ~]$
```

There is nothing in the directory.

b. Section 2.2.2 -Load the dataset to HDFS.

We are trying to load/copy a dataset of the local file system to the Hadoop filesystem.

2.2.2.1 Put command (=CopyFromLocal)

The Hadoop fs shell command put is similar to the copyFromLocal, which copies files or directories from the local filesystem to the destination in the Hadoop filesystem.

Let's **put/copy** a text file into the WordCount directory in HDFS.

hadoop fs -put /home/data/CSC534BDA/MapReduce/MaryHadALittleLamb.txt WordCount

And checked the file under WordCount in HDFS

```
tdend2@node00:~
[tdend2@node00 ~]$ hadoop fs -put /home/data/CSC534BDA/MapReduce/MaryHadALittleLamb.txt WordCount
[tdend2@node00 ~]$  hadoop fs -ls
Found 1 items
drwxr-xr-x   - tdend2 hadoop          0 2024-09-07 13:01 WordCount
[tdend2@node00 ~]$
```

<u>2.2.2.2 Get command (= CopyToLocal)</u>

In reverse, we are trying to copy the 'MaryHadALittleLamb.txt in HDFS to the local file system. The Hadoop fs shell command get copies the file or directory from the Hadoop file system to the local file system.

hadoop fs -get WordCount/MaryHadALittleLamb.txt CSC534BDA

ls CSC534BDA

```
tdend2@node00:~
[tdend2@node00 ~]$  hadoop fs -get WordCount/MaryHadALittleLamb.txt CSC534BDA
[tdend2@node00 ~]$ ls CSC534BDA
MaryHadALittleLamb.txt
[tdend2@node00 ~]$
```

Now we can see the text file under the CSC534BDA directory in your home directory in the local file system. So, you should have the text file in both your user directory in HDFS and your home directory in the local filesystem.

2.2.3 More commands

c. Section 2.2.3.1 cat

We are using the cat command to display the content of the file present in the WordCount directory of HDFS. The cat command reads the file in HDFS and displays the file's content on the console or stdout.

hadoop fs -cat WordCount/MaryHadALittleLamb.txt

```
tdend2@node00:~
[tdend2@node00 ~]$  hadoop fs -cat WordCount/MaryHadALittleLamb.txt
Mary had a little lamb
its fleece was white as snow
and everywhere that Mary went
the lamb was sure to go.
[tdend2@node00 ~]$
```

Below is the same command with the full path. HDFS' user directory name starts with /user, while the Linux home directory starts with '/home'. Note: This may give a hint, to know whether it's HDFS or Linux filesystem.

hadoop fs -cat /user/tdend2/WordCount/MaryHadALittleLamb.txt

```
Select tdend2@node00:~
[tdend2@node00 ~]$  hadoop fs -cat /user/tdend2/WordCount/MaryHadALittleLamb.txt
Mary had a little lamb
its fleece was white as snow
and everywhere that Mary went
the lamb was sure to go.
[tdend2@node00 ~]$
```

2.2.3.2 mv

The HDFS mv command moves the files or directories from the source to a destination within HDFS.

2.2.3.3 cp

The cp command copies a file from one directory to another directory within HDFS.

2.2.3.4 rm

hadoop fs -mv <src> <dest>

hadoop fs -cp <src> <dest>

hadoop fs –rm <path>

The rm command removes the file present in the specified path.

3 Programming model and an associated implementation for processing and generating large data sets – MapReduce

d. Section 3.2.3.2 – 3.2.3.5

"MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs and a reduce function that merges all intermediate values associated with the same intermediate key. Many real-world tasks are expressible in this model."

3.2 Counting words using MapReduce

Simple program to count words in a text file stored in HDFS using MapReduce.

3.2.1 Simple dataset

To write a program that counts unique words we will use the text file as input

data for your MapReduce program

cat /home/data/CSC534BDA/MapReduce/MaryHadALittleLamb.txt

```
Select tdend2@node00:~
[tdend2@node00 ~]$ cat /home/data/CSC534BDA/MapReduce/MaryHadALittleLamb.txt
Mary had a little lamb
its fleece was white as snow
and everywhere that Mary went
the lamb was sure to go.
[tdend2@node00 ~]$
```

3.2.2 Workflow

The data goes through the following phases

Input Splits:

An input to a MapReduce job is divided into fixed-size pieces called input splits. Input split is a chunk of the input consumed by a single map.

Mapping:

$ cat /home/data/CSC534BDA/MapReduce/MaryHadALittleLamb.txt

Mary had a little lamb

its fleece was white as snow

and everywhere that Mary went

the lamb was sure to go.

This is the very first phase in the execution of a map-reduce program. In this phase, data in each

split is passed to a mapping function to produce output values. In our example, a job of the mapping phase is to count the number of occurrences of each word from input splits (more details about input-split are given below) and prepare a list in the form of <word, frequency>

Shuffling:

This phase consumes the output of the Mapping phase. Its task is to consolidate the relevant records from the Mapping phase output. In our example, the same words are clubbed together with their respective frequency.

3.2.3 Word count program

3.2.3.1 Source code

Word count MapReduce can be found here:

https://hadoop.apache.org/docs/current/hadoopmapreduce-client/hadoop-mapreduce-clientcore/MapReduceTutorial.html#Example:_WordCount_v1.0.

It's written in Java using MapReduce. You also can find the java source in the below path.

cat /home/data/CSC534BDA/MapReduce/WordCount.java

tdend2@node00:~

```
[tdend2@node00 ~]$
[tdend2@node00 ~]$  cat /home/data/CSC534BDA/MapReduce/WordCount.java
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }

  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
```

```
tdend2@node00:~

public class WordCount {

  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }

  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
[tdend2@node00 ~]$
```

Navigated to course directory:

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 ~]$  cd CSC534BDA
[tdend2@node00 CSC534BDA]$
```

3.2.3.2 Set environment variables

To try to compile the source code, environment variables are set as follows:

export JAVA_HOME=/usr/java/jdk1.8.0_181-cloudera

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$ export JAVA_HOME=/usr/java/jdk1.8.0_181-cloudera
[tdend2@node00 CSC534BDA]$  export PATH=${JAVA_HOME}/bin:${PATH}
[tdend2@node00 CSC534BDA]$  export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar
[tdend2@node00 CSC534BDA]$ java -version
java version "1.8.0_181"
Java(TM) SE Runtime Environment (build 1.8.0_181-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode)
[tdend2@node00 CSC534BDA]$
```

3.2.3.3 Compile and create a Jar

Copied the WordCount.java to working directory and compiled the code.

hadoop com.sun.tools.javac.Main WordCount.java

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$
[tdend2@node00 CSC534BDA]$ cp /home/data/CSC534BDA/MapReduce/WordCount.java /home/tdend2/CSC534BDA/
[tdend2@node00 CSC534BDA]$ cd /home/tdend2/CSC534BDA
[tdend2@node00 CSC534BDA]$ ls
MaryHadALittleLamb.txt  WordCount.java
[tdend2@node00 CSC534BDA]$  hadoop com.sun.tools.javac.Main WordCount.java
[tdend2@node00 CSC534BDA]$
```

jar cf wc.jar WordCount.class*

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$ jar cf wc.jar WordCount*.class
[tdend2@node00 CSC534BDA]$ ls
MaryHadALittleLamb.txt  wc.jar  WordCount.class  WordCount$IntSumReducer.class  WordCount.java  WordCount$TokenizerMapper.class
[tdend2@node00 CSC534BDA]$
```

Finally, we have a word count program, wc.jar

3.2.3.4 Run WordCount program

Now running a MapReduce (parallel) program in a Hadoop cluster for the first time. Able to quickly run the word count program (wc.jar) by running the hadoop jar command with a class name, input directory/file, and output directory. We can also give a directory name, WordCount, instead of a filename as an input. Hadoop will read all files in the directory.

hadoop jar wc.jar WordCount WordCount/MaryHadALittleLamb.txt WordCount/output

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$  hadoop jar wc.jar WordCount WordCount/MaryHadALittleLamb.txt WordCour
WARNING: Use "yarn jar" to launch YARN applications.
24/09/07 14:19:41 INFO client.RMProxy: Connecting to ResourceManager at node00.sun/10.0.0.10:8032
24/09/07 14:19:42 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not perf
.
24/09/07 14:19:42 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/to
24/09/07 14:19:42 INFO input.FileInputFormat: Total input files to process : 1
24/09/07 14:19:42 INFO mapreduce.JobSubmitter: number of splits:1
24/09/07 14:19:42 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.e
24/09/07 14:19:42 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1722897143033_0027
24/09/07 14:19:42 INFO mapreduce.JobSubmitter: Executing with tokens: []
24/09/07 14:19:43 INFO conf.Configuration: resource-types.xml not found
24/09/07 14:19:43 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
24/09/07 14:19:43 INFO impl.YarnClientImpl: Submitted application application_1722897143033_0027
24/09/07 14:19:43 INFO mapreduce.Job: The url to track the job: http://node00.sun:8088/proxy/appl
24/09/07 14:19:43 INFO mapreduce.Job: Running job: job_1722897143033_0027
24/09/07 14:19:49 INFO mapreduce.Job: Job job_1722897143033_0027 running in uber mode : false
24/09/07 14:19:49 INFO mapreduce.Job:  map 0% reduce 0%
24/09/07 14:19:54 INFO mapreduce.Job:  map 100% reduce 0%
24/09/07 14:20:00 INFO mapreduce.Job:  map 100% reduce 59%
24/09/07 14:20:01 INFO mapreduce.Job:  map 100% reduce 80%
24/09/07 14:20:03 INFO mapreduce.Job:  map 100% reduce 82%
24/09/07 14:20:04 INFO mapreduce.Job:  map 100% reduce 95%
24/09/07 14:20:05 INFO mapreduce.Job:  map 100% reduce 100%
24/09/07 14:20:05 INFO mapreduce.Job: Job job_1722897143033_0027 completed successfully
24/09/07 14:20:05 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=1087
                FILE: Number of bytes written=9894074
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=241
                HDFS: Number of bytes written=131
                HDFS: Number of read operations=223
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=88
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=44
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=2778
                Total time spent by all reduces in occupied slots (ms)=146372
                Total time spent by all map tasks (ms)=2778
                Total time spent by all reduce tasks (ms)=146372
                Total vcore-milliseconds taken by all map tasks=2778
                Total vcore-milliseconds taken by all reduce tasks=146372
                Total megabyte-milliseconds taken by all map tasks=2844672
                Total megabyte-milliseconds taken by all reduce tasks=149884928
```

```
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=44
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=2778
                Total time spent by all reduces in occupied slots (ms)=146372
                Total time spent by all map tasks (ms)=2778
                Total time spent by all reduce tasks (ms)=146372
                Total vcore-milliseconds taken by all map tasks=2778
                Total vcore-milliseconds taken by all reduce tasks=146372
                Total megabyte-milliseconds taken by all map tasks=2844672
                Total megabyte-milliseconds taken by all reduce tasks=149884928
        Map-Reduce Framework
                Map input records=4
                Map output records=22
                Map output bytes=195
                Map output materialized bytes=911
                Input split bytes=132
                Combine input records=22
                Combine output records=19
                Reduce input groups=19
                Reduce shuffle bytes=911
                Reduce input records=19
                Reduce output records=19
                Spilled Records=38
                Shuffled Maps =44
                Failed Shuffles=0
                Merged Map outputs=44
                GC time elapsed (ms)=2942
                CPU time spent (ms)=56020
                Physical memory (bytes) snapshot=11559112704
                Virtual memory (bytes) snapshot=119219339264
                Total committed heap usage (bytes)=13634633728
                Peak Map Physical memory (bytes)=535937024
                Peak Map Virtual memory (bytes)=2638290944
                Peak Reduce Physical memory (bytes)=287535104
                Peak Reduce Virtual memory (bytes)=2652594176
        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=109
        File Output Format Counters
                Bytes Written=131
[tdend2@node00 CSC534BDA]$
```

3.2.3.5 Output

Output files are shown in the WordCount/output directory as you specified in the previous command. We see output files in the directory as follows.

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$  hadoop fs -ls WordCount/output
Found 45 items
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:20 WordCount/output/_SUCCESS
-rw-r--r--   3 tdend2 hadoop          7 2024-09-07 14:19 WordCount/output/part-r-00000
-rw-r--r--   3 tdend2 hadoop         13 2024-09-07 14:19 WordCount/output/part-r-00001
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00002
-rw-r--r--   3 tdend2 hadoop          7 2024-09-07 14:19 WordCount/output/part-r-00003
-rw-r--r--   3 tdend2 hadoop         11 2024-09-07 14:19 WordCount/output/part-r-00004
-rw-r--r--   3 tdend2 hadoop          9 2024-09-07 14:19 WordCount/output/part-r-00005
-rw-r--r--   3 tdend2 hadoop          6 2024-09-07 14:19 WordCount/output/part-r-00006
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00007
-rw-r--r--   3 tdend2 hadoop          6 2024-09-07 14:19 WordCount/output/part-r-00008
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00009
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00010
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00011
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00012
-rw-r--r--   3 tdend2 hadoop          7 2024-09-07 14:19 WordCount/output/part-r-00013
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00014
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00015
-rw-r--r--   3 tdend2 hadoop          7 2024-09-07 14:19 WordCount/output/part-r-00016
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00017
-rw-r--r--   3 tdend2 hadoop          6 2024-09-07 14:19 WordCount/output/part-r-00018
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00019
-rw-r--r--   3 tdend2 hadoop         15 2024-09-07 14:19 WordCount/output/part-r-00020
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00021
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00022
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00023
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00024
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00025
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00026
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00027
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00028
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00029
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00030
-rw-r--r--   3 tdend2 hadoop          6 2024-09-07 14:19 WordCount/output/part-r-00031
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00032
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00033
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:19 WordCount/output/part-r-00034
-rw-r--r--   3 tdend2 hadoop          5 2024-09-07 14:20 WordCount/output/part-r-00035
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:20 WordCount/output/part-r-00036
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:20 WordCount/output/part-r-00037
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:20 WordCount/output/part-r-00038
-rw-r--r--   3 tdend2 hadoop          9 2024-09-07 14:20 WordCount/output/part-r-00039
-rw-r--r--   3 tdend2 hadoop          4 2024-09-07 14:20 WordCount/output/part-r-00040
-rw-r--r--   3 tdend2 hadoop          6 2024-09-07 14:20 WordCount/output/part-r-00041
-rw-r--r--   3 tdend2 hadoop          7 2024-09-07 14:20 WordCount/output/part-r-00042
-rw-r--r--   3 tdend2 hadoop          0 2024-09-07 14:20 WordCount/output/part-r-00043
[tdend2@node00 CSC534BDA]$
```

---

2. Ran the word count program with a real dataset,screenshots of running your code/program and its outputs/results with some titles/explanations is shown here.

a. Coronavirus tweets dataset (sample).

i. Dataset location (Linux filesystem): /home/data/CSC534BDA/datasets/COVID19/

ii. Dataset filename: coronavirus-text-only-1000.txt

iii. It contains 1000 tweets (text message only) with the keyword ‘coronavirus’ in its text message.

b. Load the dataset into your user directory in HDFS

i. Created a new directory, COVID19, in HDFS,
e.g., /user/tdend2/COVID19

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 ~]$ hadoop fs -mkdir /user/tdend2/COVID19
```

Copied the dataset coronavirus-text-only-1000.txt from the Linux filesystem to the COVID19 directory in HDFS

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 ~]$ hadoop fs -put /home/data/CSC534BDA/datasets/COVID19/coronavirus-text-only-1000.txt /user/tdend2/COVID19/
put: `/user/tdend2/COVID19/coronavirus-text-only-1000.txt': File exists
[tdend2@node00 ~]$ hadoop fs -ls /user/tdend2/COVID19/
Found 1 items
-rw-r--r--   3 tdend2 hadoop     149569 2024-09-08 12:00 /user/tdend2/COVID19/coronavirus-text-only-1000.txt
```

ii. Note: did not copy the dataset into your local Linux filesystem

c. Run the word count program with the dataset

Run the word count program (wc.jar) by running the hadoop jar command with a class name, input directory/file, and output directory. We can also give a directory name, WordCount, instead of a filename as an input. Hadoop will read all files in the directory.

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$ hadoop jar wc.jar WordCount /user/tdend2/COVID19/coronavirus-text-only-1000.txt /user/tdend2/COVID19/output-text
WARNING: Use "yarn jar" to launch YARN applications.
24/09/08 12:17:23 INFO client.RMProxy: Connecting to ResourceManager at node00.sun/10.0.0.10:8032
24/09/08 12:17:23 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and exe
.
24/09/08 12:17:24 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/tdend2/.staging/job_1722897143033_0045
24/09/08 12:17:24 INFO input.FileInputFormat: Total input files to process : 1
24/09/08 12:17:24 INFO mapreduce.JobSubmitter: number of splits:1
24/09/08 12:17:24 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.sys
24/09/08 12:17:24 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1722897143033_0045
24/09/08 12:17:24 INFO mapreduce.JobSubmitter: Executing with tokens: []
24/09/08 12:17:24 INFO conf.Configuration: resource-types.xml not found
24/09/08 12:17:24 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
24/09/08 12:17:24 INFO impl.YarnClientImpl: Submitted application application_1722897143033_0045
24/09/08 12:17:24 INFO mapreduce.Job: The url to track the job: http://node00.sun:8088/proxy/application_1722897143033_0045/
24/09/08 12:17:24 INFO mapreduce.Job: Running job: job_1722897143033_0045
24/09/08 12:17:31 INFO mapreduce.Job: Job job_1722897143033_0045 running in uber mode : false
24/09/08 12:17:31 INFO mapreduce.Job:  map 0% reduce 0%
24/09/08 12:17:36 INFO mapreduce.Job:  map 100% reduce 0%
24/09/08 12:17:43 INFO mapreduce.Job:  map 100% reduce 70%
24/09/08 12:17:44 INFO mapreduce.Job:  map 100% reduce 80%
24/09/08 12:17:47 INFO mapreduce.Job:  map 100% reduce 100%
24/09/08 12:17:47 INFO mapreduce.Job: Job job_1722897143033_0045 completed successfully
24/09/08 12:17:48 INFO mapreduce.Job: Counters: 54
```

Last half of it is below:

```
tdend2@node00:~/CSC534BDA
                HDFS: Number of write operations=88
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=44
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=3246
                Total time spent by all reduces in occupied slots (ms)=150775
                Total time spent by all map tasks (ms)=3246
                Total time spent by all reduce tasks (ms)=150775
                Total vcore-milliseconds taken by all map tasks=3246
                Total vcore-milliseconds taken by all reduce tasks=150775
                Total megabyte-milliseconds taken by all map tasks=3323904
                Total megabyte-milliseconds taken by all reduce tasks=154393600
        Map-Reduce Framework
                Map input records=1000
                Map output records=20576
                Map output bytes=230426
                Map output materialized bytes=74390
                Input split bytes=138
                Combine input records=20576
                Combine output records=5720
                Reduce input groups=5720
                Reduce shuffle bytes=74390
                Reduce input records=5720
                Reduce output records=5720
                Spilled Records=11440
                Shuffled Maps =44
                Failed Shuffles=0
                Merged Map outputs=44
                GC time elapsed (ms)=3046
                CPU time spent (ms)=67590
                Physical memory (bytes) snapshot=11876556800
                Virtual memory (bytes) snapshot=119235338240
                Total committed heap usage (bytes)=14067695616
                Peak Map Physical memory (bytes)=562348032
                Peak Map Virtual memory (bytes)=2644017152
                Peak Reduce Physical memory (bytes)=286818304
                Peak Reduce Virtual memory (bytes)=2652573696
        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=149569
        File Output Format Counters
                Bytes Written=70409
```

i. Show the first 10 lines of the outputs/results.

We can see outputs by running the below command.

$ hadoop fs -cat COVID19/output-text/part-r-00000 | head

Head - Returns the first 10 lines of the output

Each file has a part of the output due to parallel processing as shown below

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$ hadoop fs -cat /user/tdend2/COVID19/output-text/part-r-00000 | head
"Eso    1
#Covid_19,      1
#ExclusivaLatinus.      4
#sarandi        1
10,000  1
132k    1
23rd,   1
3031    1
99%     3
@FaceTheNation  1
[tdend2@node00 CSC534BDA]$
```

First half of output for part o without using head:

```
tdend2@node00:~/CSC534BDA
[tdend2@node00 CSC534BDA]$ hadoop fs -cat /user/tdend2/COVID19/output-text/part-r-00000
"Eso    1
#Covid_19,      1
#ExclusivaLatinus.      4
#sarandi        1
10,000  1
132k    1
23rd,   1
3031    1
99%     3
@FaceTheNation  1
@Mary_Js_ART:   1
@MegafonoPopular:       1
@NYPost_Mets    1
@NewsHour:      1
@RepJayapal:    1
@VictOcampo:    1
@charliekirk11: 58
@hotmart        2
@mirandayaver:  4
@pagseguro      2
@porttada:      1
@thejtlewis     1
Arizonans       2
CV      1
CoronavV≠rus:   1
Daily   3
Depende 1
Employees       4
Fall    1
Florida 32
GimV@nez:       2
Half    2
Idaho   1
Ill     1
Laurepinpon     1
Malaysia        3
Neanderthals    1
Nebraska        1
OBESITY,        1
On      4
Proclamation    1
Remember:       2
Sky     1
South   14
Stephen 7
WH.     1
above   1
alguV@m 2
amooAAAAAAA     1
```

Last half of part0 output without head:

```
habV≠a  3
help.   1
heritage        2
hicieron        1
highs   1
https://t.co/2hZTD9c9LH 1
https://t.co/9tOoWpYLw3 1
https://t.co/DdlYJ4IpcL 3
https://t.co/H1AUlB4BCy 1
https://t.co/UQMDXZCDTE 1
https://t.co/XAXtOkW4LO"        1
https://t.co/ebIFtlmrPJ.        1
https://t.co/ycQGAgcueD 1
ideal   1
iss     1
leurs   1
lists   1
lock    3
l,Ä¶"   2
mutaciv≥n       1
nadie   2
news    7
nv∫meros        1
per     5
perks   1
really  6
rosy    1
sacrifice       1
sells   1
soon    1
strangers,      1
system  1
tercer  1
today,  9
toenails        1
took    2
touts   1
travv®s 2
update: 1
virus   15
volver,Ä¶       1
wil,Ä¶  1
wrong   2
,ÄúNo   1
,Äúyou  1
‡§ú‡•ç‡§Ø‡§æ‡§¶‡§æ      1
▯üè™    1
▯üíú    1
▯üî¥▯üá°▯üá±    1
[tdend2@node00 CSC534BDA]$
```

++++++++++++++++++++++++++++++++THE END++++++++++++++++++++++++++++++++