

Exercise 03a: Map Reduce applications for Word Counting

Name: Annapoornima .S

Roll no: 225229101

Previous exercise described how to save input file in to HDFS. This exercise train students to do MapReduce process using word counting application.

Prerequisites

Ensure that Hadoop is installed, configured and is running. More details:

Single Node Setup for first-time users.

Cluster Setup for large, distributed clusters.

MapReduce Overview

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

A MapReduce *job* usually splits the input data-set into independent chunks which are processed by the *map tasks* in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the *reduce tasks*. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

Typically the compute nodes and the storage nodes are the same, that is, the MapReduce framework and the Hadoop Distributed File System are running on the same set of nodes. This configuration allows the framework to effectively schedule tasks on the nodes where data is already present, resulting in very high aggregate bandwidth across the cluster.

The MapReduce framework consists of a single master `ResourceManager`, one worker `NodeManager` per cluster-node, and `MRAppMaster` per application.

Minimally, applications specify the input/output locations and supply *map* and *reduce* functions via implementations of appropriate interfaces and/or abstract-classes. These, and other job parameters, comprise the *job configuration*.

The Hadoop *job client* then submits the job (jar/executable etc.) and configuration to the `ResourceManager` which then assumes the responsibility of distributing the software/configuration to the workers, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.

Inputs and Outputs

The MapReduce framework operates exclusively on `<key, value>` pairs, that is, the framework views the input to the job as a set of `<key, value>` pairs and produces a set of `<key, value>` pairs as the output of the job, conceivably of different types.

The `key` and `value` classes have to be serializable by the framework and hence need to implement the `Writable` interface. Additionally, the `key` classes have to implement the `WritableComparable` interface to facilitate sorting by the framework.

Input and Output types of a MapReduce job:

(input) `<k1, v1>` -> **map** -> `<k2, v2>` -> **combine** -> `<k2, v2>` -> **reduce** -> `<k3, v3>` (output)

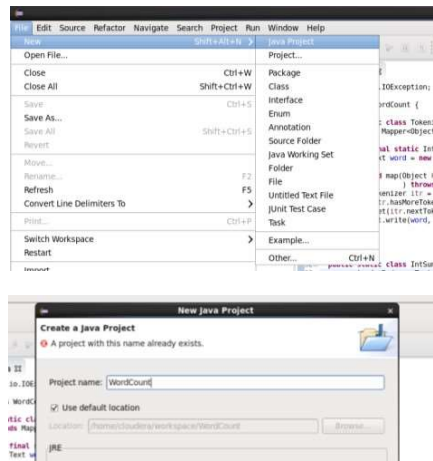
Step 1

Compile `WordCount.java` and create a jar:

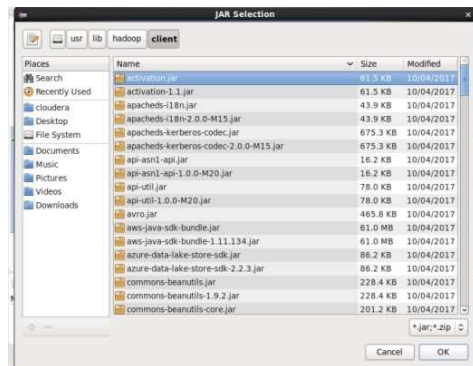
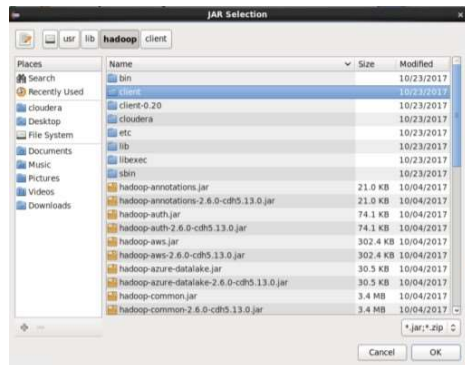
- (i) Open Eclipse in Cloudera



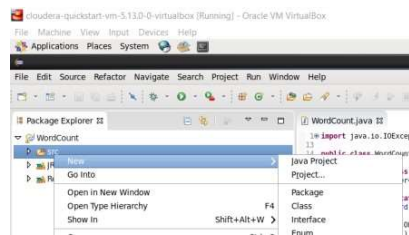
- (ii) Create 'WordCount' java project

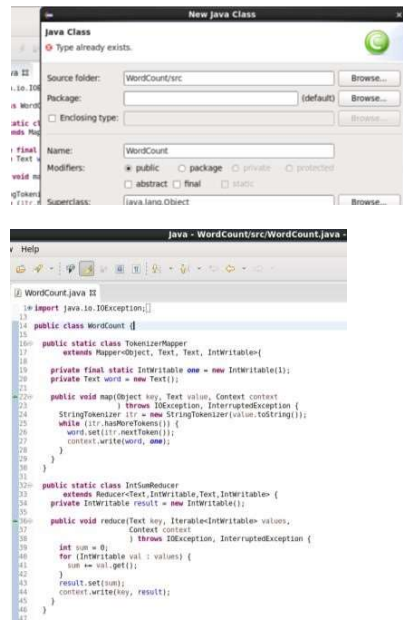


Import following Jar files

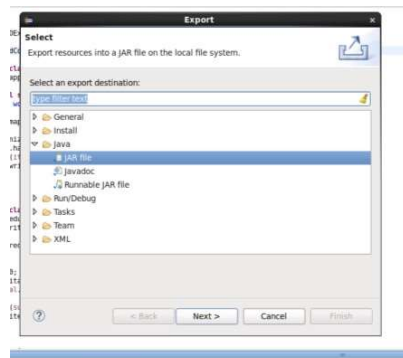


(iii) Create 'WordCount.java' in src folder





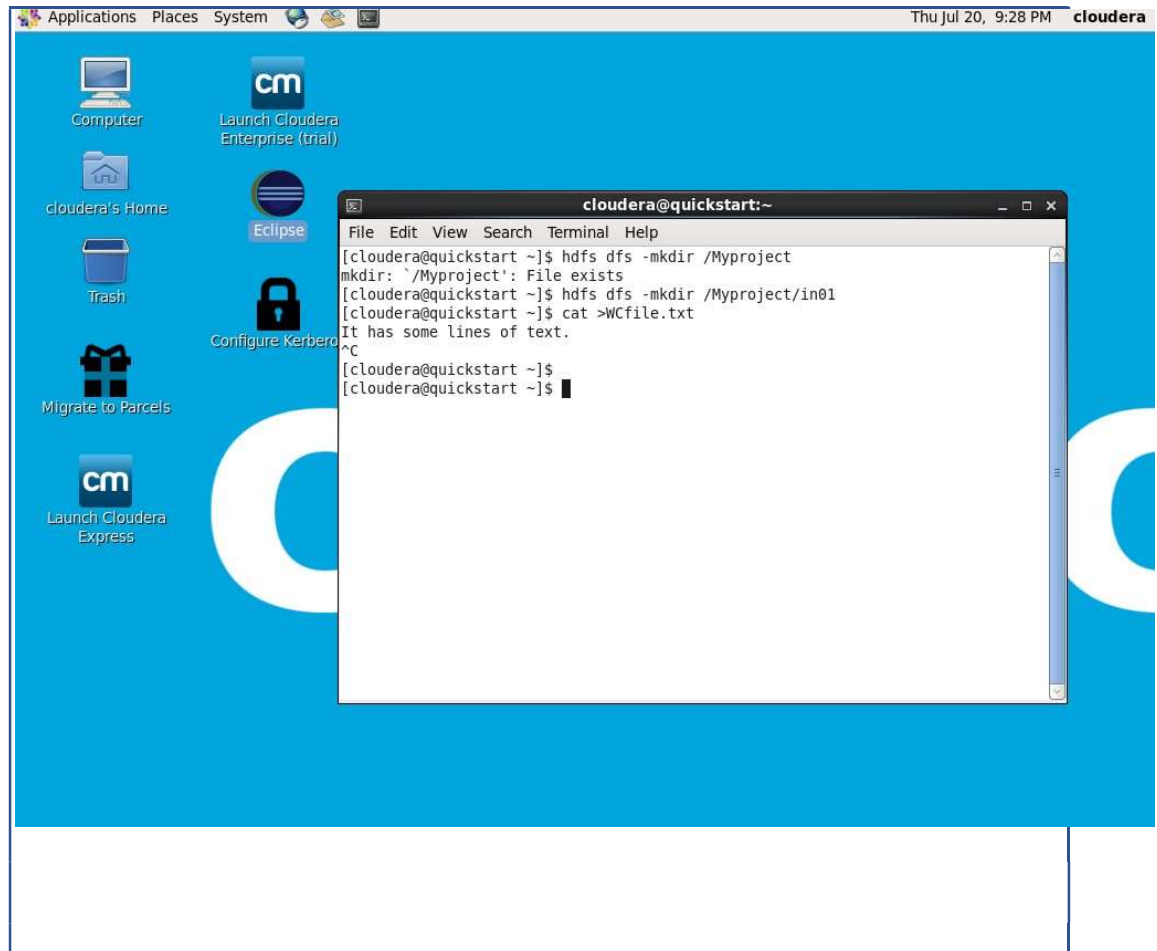
(iv) Create WordCount.jar file



Step 2

Create following folders in HDFS:

- /input - input directory in HDFS
- /output - output directory in HDFS



Step 3

Create and copy sample text-files into input folder:

```
[cloudera@quickstart ~]$ hdfs dfs -ls /in00/
```

Found 1 items

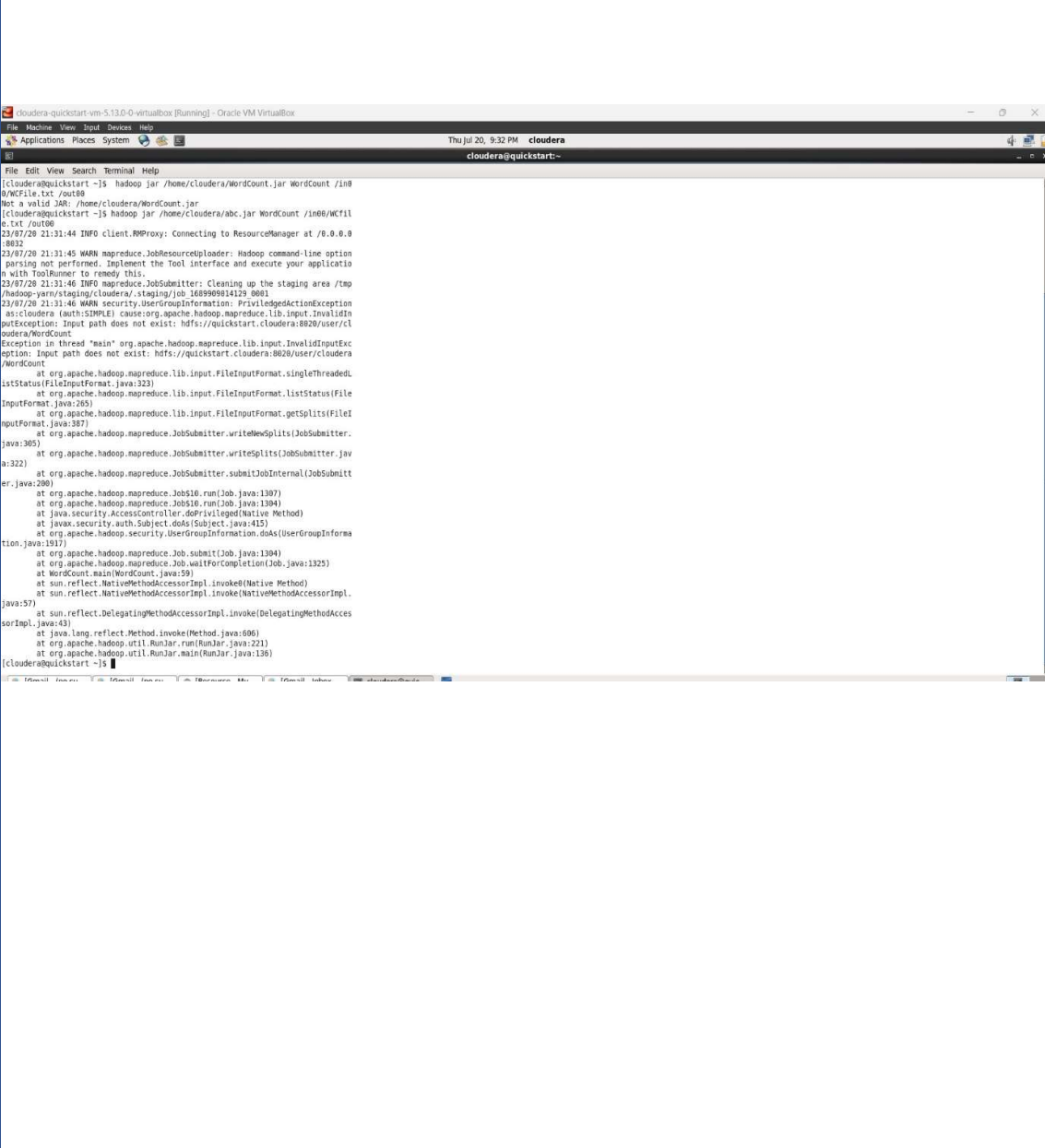
```
-rw-r--r-- 1 cloudera supergroup 158 2021-08-15 04:32 /in00/WCFile.txt
```

Step 4

Run the MapReduce application: `hadoop jar /home/cloudera/WordCount.jar`

`WordCount /in00/WCFile.txt /out00`

Show MapReduce Framework



```
cloudera@quickstart:~$ hadoop jar /home/cloudera/WordCount.jar WordCount /in00/WCFile.txt /out00
Not a valid JAR: /home/cloudera/WordCount.jar
cloudera@quickstart:~$ hadoop jar /home/cloudera/abc.jar WordCount /in00/WCFile.txt /out00
23/07/20 21:31:44 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
23/07/20 21:31:45 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
23/07/20 21:31:46 INFO mapreduce.JobSubmitter: Cleaning up the staging area /tmp/hadoop-yarn/staging/cloudera/staging/job_1689909814129_0001
23/07/20 21:31:46 WARN security.UserGroupInformation: PrivilegedActionException as: cloudera (auth:SIMPLE) cause: org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input path does not exist: hdfs://quickstart.cloudera:8020/user/cloudera/WordCount
Exception in thread "main" org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input path does not exist: hdfs://quickstart.cloudera:8020/user/cloudera/WordCount
    at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:323)
    at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.listStatus(FileInputFormat.java:265)
    at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.getSplits(FileInputFormat.java:387)
    at org.apache.hadoop.mapreduce.JobSubmitter.writeNewSplits(JobSubmitter.java:365)
    at org.apache.hadoop.mapreduce.JobSubmitter.writeSplits(JobSubmitter.java:322)
    at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:200)
    at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1307)
    at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1304)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:415)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1917)
    at org.apache.hadoop.mapreduce.Job.submit(Job.java:1304)
    at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1325)
    at WordCount.main(WordCount.java:59)
    at sun.reflect.NativeMethodAccessorImpl.invoke(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    at org.apache.hadoop.util.RunJar.run(RunJar.java:221)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:136)
cloudera@quickstart:~$
```

Step 5

Output:

`[cloudera@quickstart ~]$ hdfs dfs -ls /out00/`

Found 2 items

```
-rw-r--r--  1 cloudera supergroup      0 2021-08-15 04:41 /out00/_SUCCESS
-rw-r--r--  1 cloudera supergroup 113 2021-08-15 04:41 /out00/part-r-00000
```

```
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/part-r-00000
```

```
[cloudera@quickstart ~]$ hdfs dfs -ls /out00/
ls: '/out00/': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/part-r-00000
cat: '/out00/part-r-00000': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -mkdir /out00
[cloudera@quickstart ~]$ hdfs dfs -ls /out00/
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/part-r-00000
cat: '/out00/part-r-00000': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -cat hdfs://localhost:8020/out00/part-r-00000
cat: 'hdfs://localhost:8020/out00/part-r-00000': No such file or directory
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/part-r-00000
cat: '/out00/part-r-00000': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/
cat: '/out00': Is a directory
[cloudera@quickstart ~]$ hdfs dfs -ls /Myproject/out82
ls: '/Myproject/out82': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -ls /Myproject/out00
ls: '/Myproject/out00': No such file or directory
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ hdfs dfs -ls /WCfile.txt/out00
ls: '/WCfile.txt/out00': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -put WCfile.txt /Myproject/inol
put: '/Myproject/inol/WCfile.txt': File exists
[cloudera@quickstart ~]$ hdfs dfs -put WCfile.txt /Myproject/out00
[cloudera@quickstart ~]$ hdfs dfs -ls /Myproject/out00
-rw-r--r-- 1 cloudera supergroup 27 2023-07-20 21:43 /Myproject/out00
[cloudera@quickstart ~]$ hdfs dfs -ls /Myproject/out00/part-r-00000
ls: '/Myproject/out00/part-r-00000': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -cat/Myproject/out00/part-r-00000
cat/Myproject/out00/part-r-00000: Unknown command
[cloudera@quickstart ~]$ hdfs dfs -cat /Myproject/out00/part-r-00000
cat: '/Myproject/out00/part-r-00000': No such file or directory
[cloudera@quickstart ~]$
```

[Gmail - (no su... [Gmail - (no su... [Resource - My... [Gmail - Inbox -... cloudera@quic...