Roll no:215229109

Exercise 02a: Map Reduce applications for Word Counting

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:

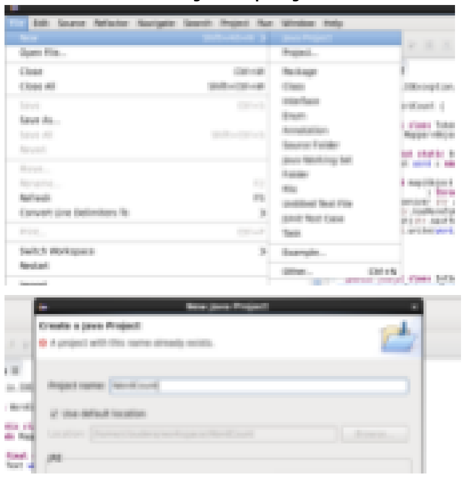
(input) `<k1, v1>` `->` **map** `->` `<k2, v2>` `->` **combine** `->` `<k2, v2>` `->` **reduce** `->` `<k3, v3>` (output)

Step 1

Compile `WordCount.java` and create a jar:

(i) Open Eclipse in Clouderea

(ii) Create 'WordCount' java project



New Java Project

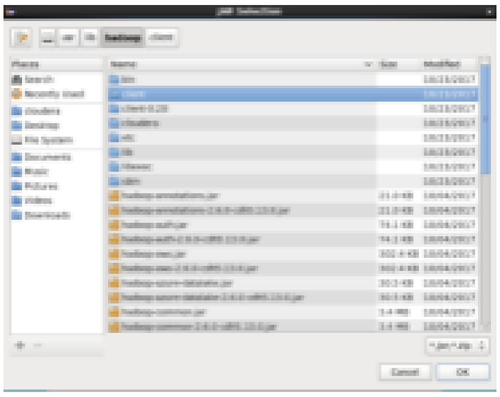
Create a Java Project

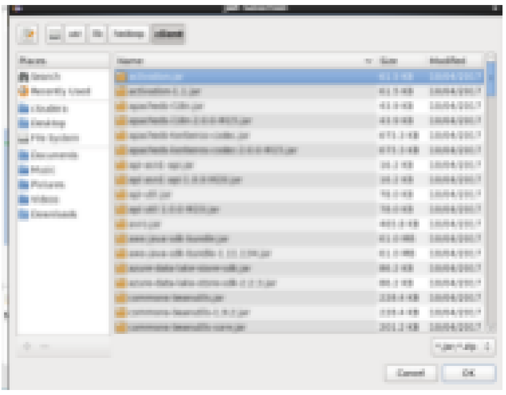
A project with this name already exists.

Project name: WordCount

☑ Use default location

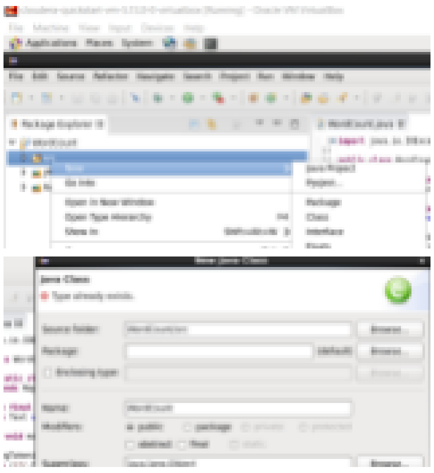
Import following Jar files

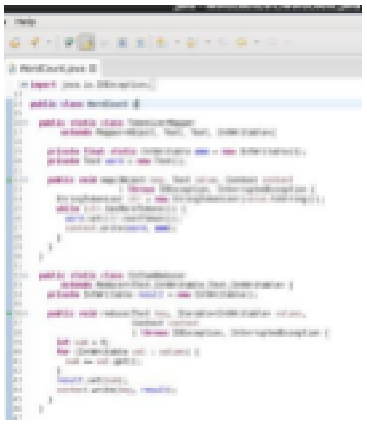




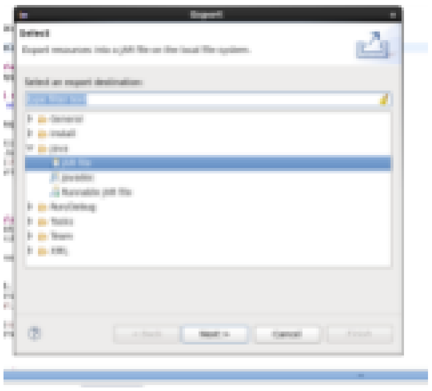
(iii)

Create ‘WordCount.java’ in src folder





(iv) Create WordCount.jar file

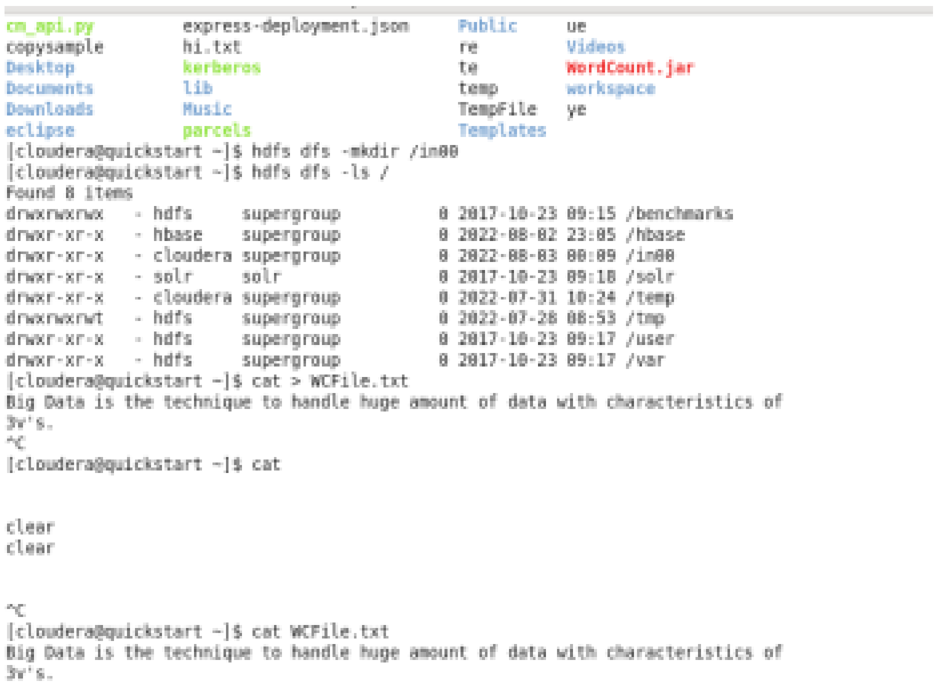


Export resources into a JAR file on the local file system.

Step 2

Create following folders in HDFS:

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



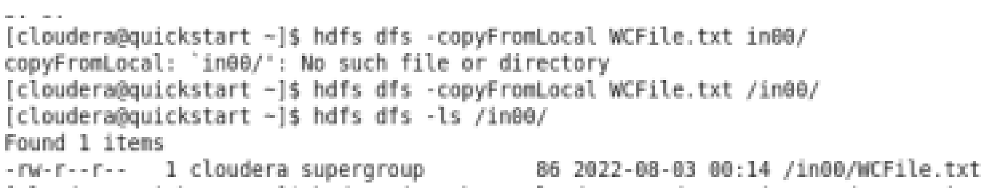
```
cm_api.py            express-deployment.json     Public      ue
copysample           hi.txt                      re          Videos
Desktop              kerberos                    te          WordCount.jar
Documents            lib                         temp        workspace
Downloads            Music                       TempFile    ye
eclipse              parcels                     Templates
[cloudera@quickstart ~]$ hdfs dfs -mkdir /in00
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 8 items
drwxrwxrwx   - hdfs     supergroup          0 2017-10-23 09:15 /benchmarks
drwxr-xr-x   - hbase    supergroup          0 2022-08-02 23:05 /hbase
drwxr-xr-x   - cloudera supergroup          0 2022-08-03 00:09 /in00
drwxr-xr-x   - solr     solr                0 2017-10-23 09:18 /solr
drwxr-xr-x   - cloudera supergroup          0 2022-07-31 10:24 /temp
drwxrwxrwt   - hdfs     supergroup          0 2022-07-28 08:53 /tmp
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 09:17 /user
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 09:17 /var
[cloudera@quickstart ~]$ cat > WCFile.txt
Big Data is the technique to handle huge amount of data with characteristics of
3v's.
^C
[cloudera@quickstart ~]$ cat


clear
clear


^C
[cloudera@quickstart ~]$ cat WCFile.txt
Big Data is the technique to handle huge amount of data with characteristics of
3v's.
```

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

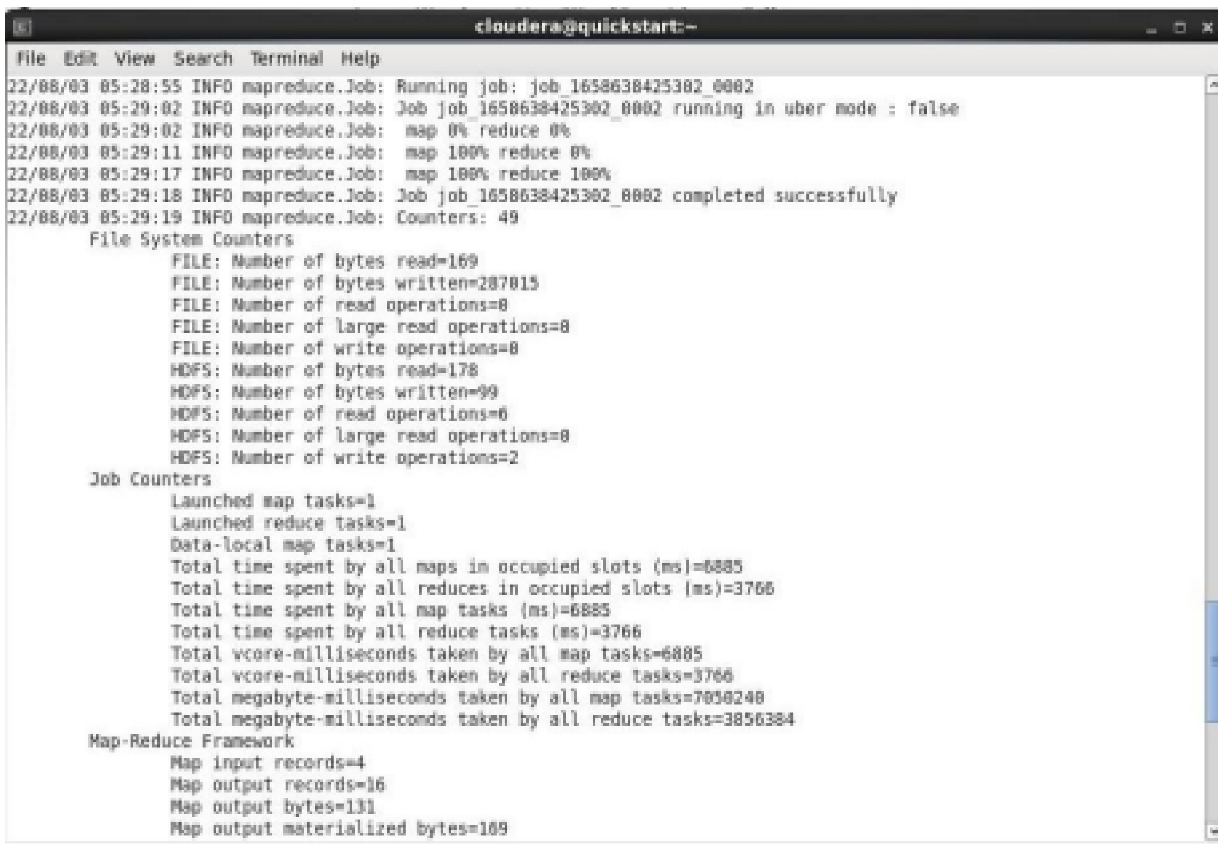


```
[cloudera@quickstart ~]$ hdfs dfs -copyFromLocal WCFile.txt in00/
copyFromLocal: `in00/': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -copyFromLocal WCFile.txt /in00/
[cloudera@quickstart ~]$ hdfs dfs -ls /in00/
Found 1 items
-rw-r--r--   1 cloudera supergroup         86 2022-08-03 00:14 /in00/WCFile.txt
```

Step 4

Run the MapReduce application:
hadoop jar /home/cloudera/WordCount.jar WordCount /in00/WCFile.txt

/out00 Show MapReduce Framework



```
22/08/03 05:28:55 INFO mapreduce.Job: Running job: job_1658638425302_0002
22/08/03 05:29:02 INFO mapreduce.Job: Job job_1658638425302_0002 running in uber mode : false
22/08/03 05:29:02 INFO mapreduce.Job:  map 0% reduce 0%
22/08/03 05:29:11 INFO mapreduce.Job:  map 100% reduce 0%
22/08/03 05:29:17 INFO mapreduce.Job:  map 100% reduce 100%
22/08/03 05:29:18 INFO mapreduce.Job: Job job_1658638425302_0002 completed successfully
22/08/03 05:29:19 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=169
                FILE: Number of bytes written=287015
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=178
                HDFS: Number of bytes written=99
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=6885
                Total time spent by all reduces in occupied slots (ms)=3766
                Total time spent by all map tasks (ms)=6885
                Total time spent by all reduce tasks (ms)=3766
                Total vcore-milliseconds taken by all map tasks=6885
                Total vcore-milliseconds taken by all reduce tasks=3766
                Total megabyte-milliseconds taken by all map tasks=7050240
                Total megabyte-milliseconds taken by all reduce tasks=3856384
        Map-Reduce Framework
                Map input records=4
                Map output records=16
                Map output bytes=131
                Map output materialized bytes=169
```

Step 5

Output:

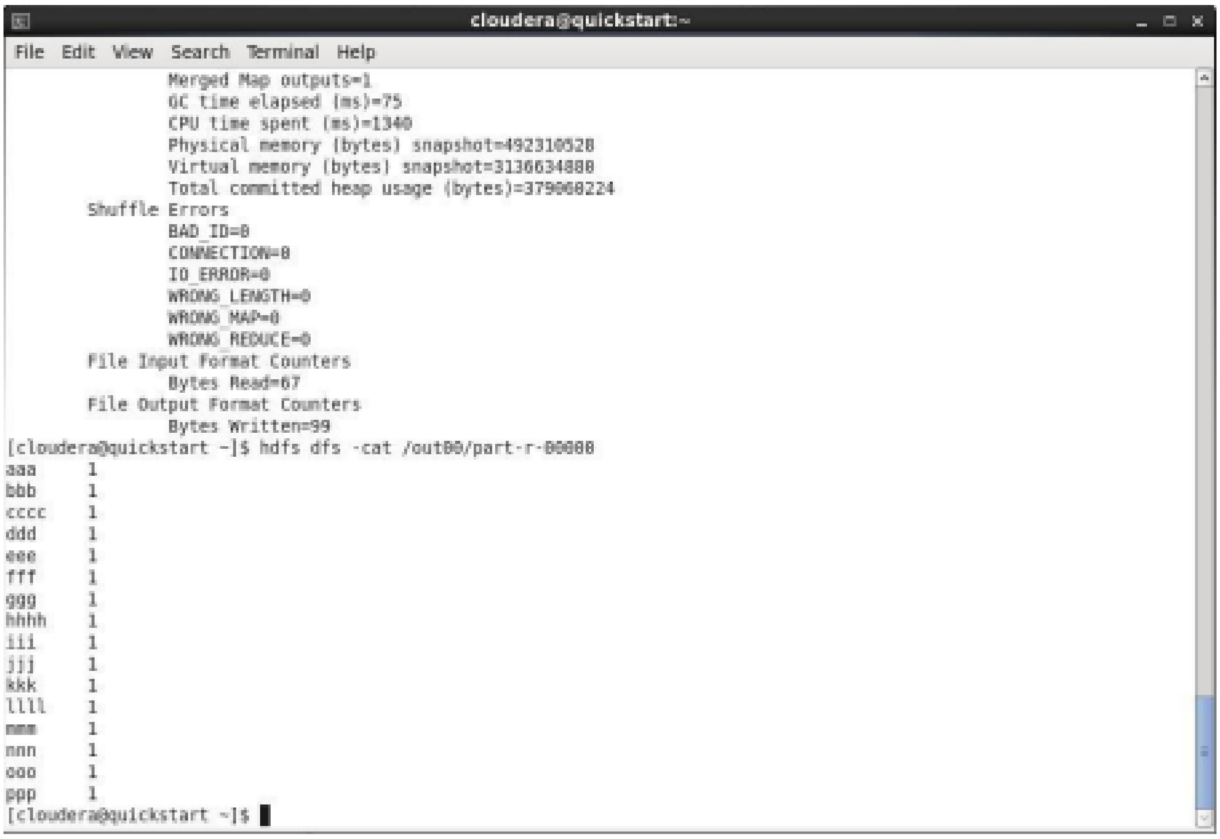
[cloudera@quickstart ~]$ hdfs dfs -ls /out00/

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



```
                Merged Map outputs=1
                GC time elapsed (ms)=75
                CPU time spent (ms)=1340
                Physical memory (bytes) snapshot=492310528
                Virtual memory (bytes) snapshot=3136634880
                Total committed heap usage (bytes)=379060224
        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=67
        File Output Format Counters
                Bytes Written=99
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/part-r-00000
aaa     1
bbb     1
cccc    1
ddd     1
eee     1
fff     1
ggg     1
hhhh    1
iii     1
jjj     1
kkk     1
llll    1
mmm     1
nnn     1
ooo     1
ppp     1
[cloudera@quickstart ~]$ 
```