Exercise 03: 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.

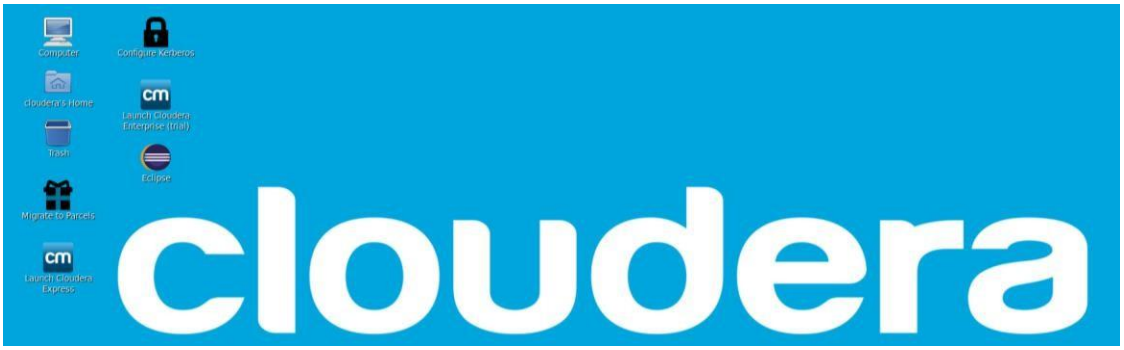
Input and Output types of a MapReduce job:

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

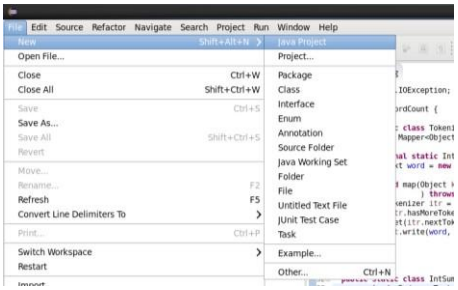
Step 1

Compile `WordCount.java` and create a jar:

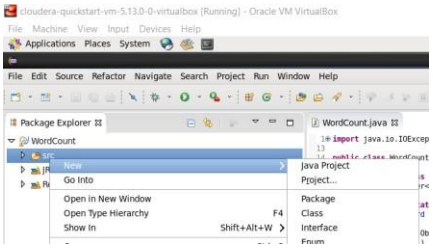
(i) Open Eclipse in Clouderea

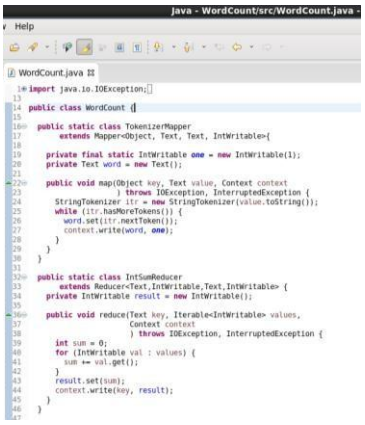


(ii) Create 'WordCount' java project

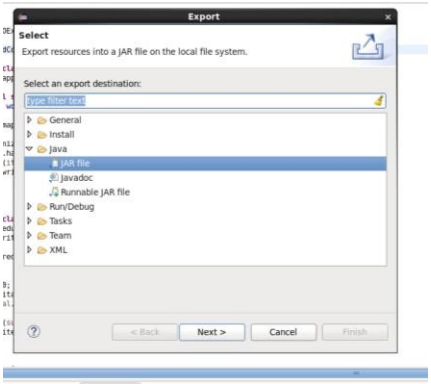


(iii) Create 'WordCount.java' in src folder





(iv) Create WordCount.jar file



Step 2

Create following folders in HDFS:

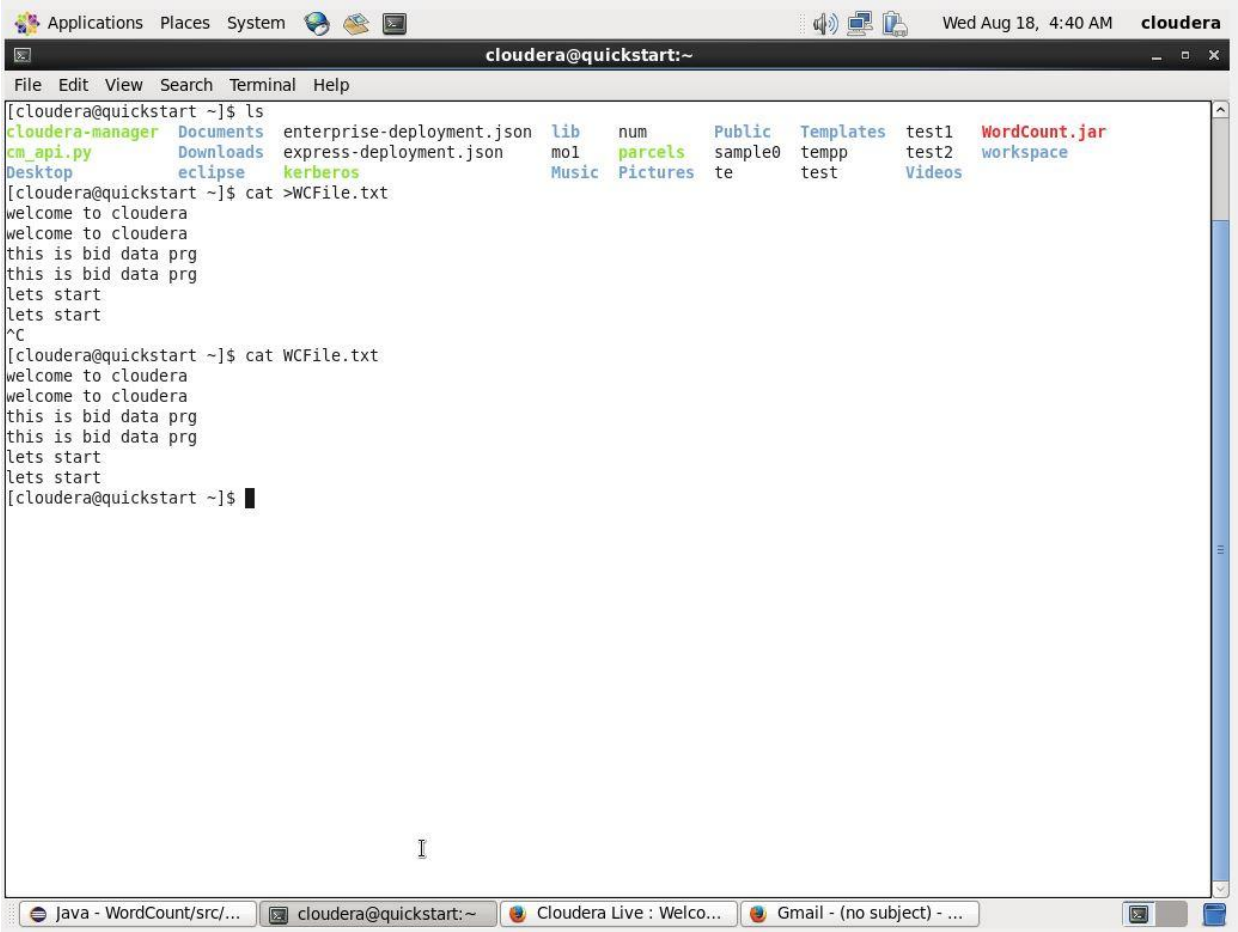
- `/input` - input directory in HDFS
- `/output` - output directory in HDFS

```
[cloudera@quickstart ~]$ hdfs dfs -mkdir /in00
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 15 items
drwxrwxrwx   - hdfs     supergroup          0 2017-10-23 09:15 /benchmarks
drwxr-xr-x   - cloudera supergroup          0 2021-08-17 08:18 /demo
drwxr-xr-x   - hbase    supergroup          0 2021-08-15 07:21 /hbase
drwxr-xr-x   - cloudera supergroup          0 2021-08-18 04:44 /in00
drwxr-xr-x   - cloudera supergroup          0 2021-08-17 00:30 /page
-rw-r--r--   1 cloudera supergroup          0 2021-08-16 23:20 /sample0
-rw-r--r--   1 cloudera supergroup          0 2021-08-16 23:17 /sample00
drwxr-xr-x   - solr     solr                0 2017-10-23 09:18 /solr
drwxr-xr-x   - cloudera supergroup          0 2021-08-16 23:59 /temp
drwxr-xr-x   - cloudera supergroup          0 2021-08-17 00:23 /temp1
drwxr-xr-x   - cloudera supergroup          0 2021-08-17 00:19 /tempe
-rw-r--r--   1 cloudera supergroup         61 2021-08-16 09:24 /tempp
drwxrwxrwt   - hdfs     supergroup          0 2021-08-12 01:56 /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 ~]$
```

Java - WordCount/src/... | cloudera@quickstart:~ | Cloudera Live : Welco... | Gmail - (no subject) - ...

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

```
Applications  Places  System                                   Wed Aug 18, 4:56 AM   cloudera
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
                Total time spent by all map tasks (ms)=29530
                Total time spent by all reduce tasks (ms)=22990
                Total vcore-milliseconds taken by all map tasks=29530
                Total vcore-milliseconds taken by all reduce tasks=22990
                Total megabyte-milliseconds taken by all map tasks=30238720
                Total megabyte-milliseconds taken by all reduce tasks=23541760
        Map-Reduce Framework
                Map input records=6
                Map output records=20
                Map output bytes=184
                Map output materialized bytes=118
                Input split bytes=112
                Combine input records=20
                Combine output records=10
                Reduce input groups=10
                Reduce shuffle bytes=118
                Reduce input records=10
                Reduce output records=10
                Spilled Records=20
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=698
                CPU time spent (ms)=2430
                Physical memory (bytes) snapshot=334016512
                Virtual memory (bytes) snapshot=3015933952
                Total committed heap usage (bytes)=226365440
        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=104
        File Output Format Counters
                Bytes Written=72
[cloudera@quickstart ~]$
Java - WordCount/src/...  cloudera@quickstart:~  Cloudera Live : Welco...  Gmail - (no subject) - ...
```

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

Step 5

```
[cloudera@quickstart ~]$ hdfs dfs -ls /out00
Found 2 items
-rw-r--r--   1 cloudera supergroup          0 2021-08-18 04:55 /out00/_SUCCESS
-rw-r--r--   1 cloudera supergroup         72 2021-08-18 04:54 /out00/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat /out00/part-r-00000
bid     2
cloudera        2
data    2
is      2
lets    2
prg     2
start   2
this    2
to      2
welcome 2
[cloudera@quickstart ~]$
```

Java - WordCount/src/... | cloudera@quickstart:~ | Cloudera Live : Welco... | Gmail - (no subject) - ...