# CS 6343: CLOUD COMPUTING

# MapReduce Assignment

# Rohini Yadav

# Net Id - rsy140030

Table of Contents

[Project Description 3](#_Toc434615148)

[Experiment 3](#_Toc434615149)

[1. Small input file 3](#_Toc434615150)

[a. Input File 3](#_Toc434615151)

[b. Block Distribution 3](#_Toc434615152)

[c. Output 4](#_Toc434615153)

[2. Large Number of Small Size Files 6](#_Toc434615154)

[a. Input 6](#_Toc434615155)

[b. Block Distribution 6](#_Toc434615156)

[c. Output 8](#_Toc434615157)

[3. Single Large File 9](#_Toc434615158)

[a. Input File 9](#_Toc434615159)

[b. Block Distribution 10](#_Toc434615160)

[c. Output 11](#_Toc434615161)

[Error Handling 13](#_Toc434615162)

[Observations 13](#_Toc434615163)

# 

# Project Description

Write a MapReduce program to compute the total crime incidents of each crime type in each region. Region definition − Crime location is defined on a coordinate system (East, North) − East and North are defined by a 5-digit numerical value .

Region definition 1: use the first digit of the coordinates only to define a region. (5xxxx, 7xxxx), (5xxxx, 3xxxx), (8xxxx, 6xxxx), each is one region. Supposedly there are 100 regions, but not all the numbers appear in the files − Region definition 2: use the first three digits of the coordinates to define a region.

Crime types include: Anti-social behavior, Burglary, Criminal damage and arson, Drugs, Other theft, Public disorder and weapons, Robbery, Shoplifting, Vehicle crime, Violent crime, Other crime.

# Experiment

The experiment was carried on a HDFS cluster with the following configuration:

* NameNode
* 3 DataNodes
* Replication factor = 2

## Small input file

### Input File

|  |  |  |
| --- | --- | --- |
| **File Name** | **File Size (MB)** | **Command** |
| 2012-02-thames-valley-street.csv | 2.41 | hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceInput/2012-02-thames-valley-street.csv  /MapreduceOutputRohini1 5 |

### Block Distribution

As the file size is less than 128 MB (block size of HDFS), the file is stored in just one block.

**Command**: hdfs fsck /MapReduceInput/2012-02-thames-valley-street.csv  -files –blocks -locations

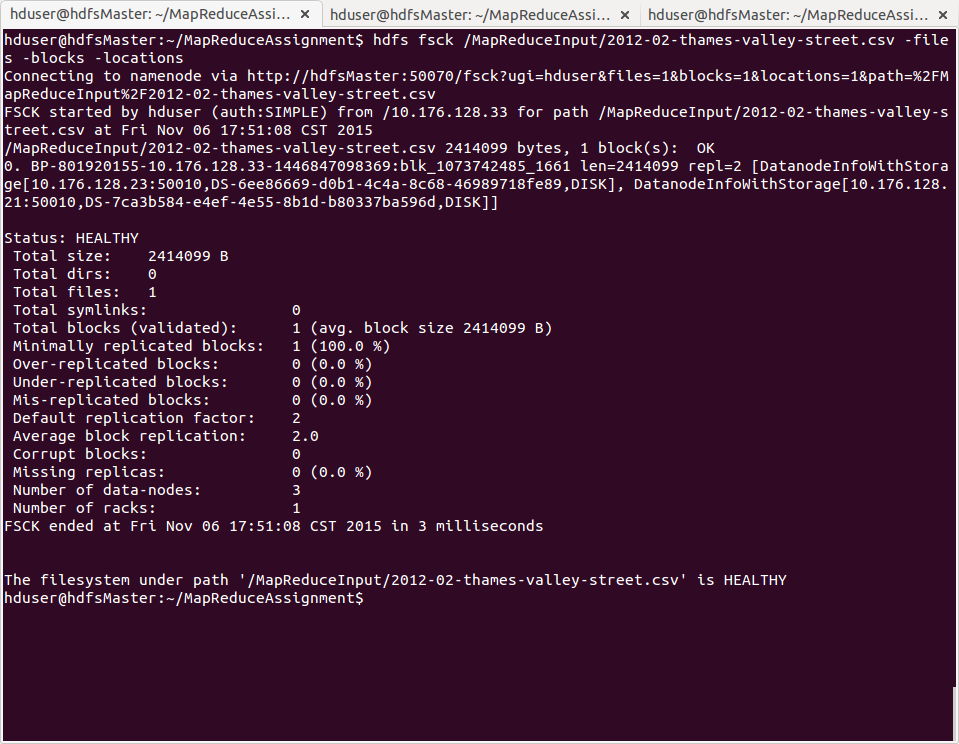


Figure Block Distribution of small file

### Output

**Mapper and Reducer Counts**

|  |  |
| --- | --- |
| **Mapper Count** | 1 |
| **Reducer count** | 1 |

**Time required for region digits**

|  |  |
| --- | --- |
| **Region digits** | **Total time (sec.)** |
| 1 | 6 |
| 3 | 7 |
| 5 | 8 |

**Output after running the command:**

*File System Counters  
        FILE: Number of bytes read=918254  
        FILE: Number of bytes written=1922928  
        FILE: Number of read operations=0  
        FILE: Number of large read operations=0  
        FILE: Number of write operations=0  
        HDFS: Number of bytes read=4828198  
        HDFS: Number of bytes written=250879  
        HDFS: Number of read operations=13  
        HDFS: Number of large read operations=0  
        HDFS: Number of write operations=4  
    Map-Reduce Framework  
        Map input records=13987  
        Map output records=13986  
        Map output bytes=425650  
        Map output materialized bytes=453628  
        Input split bytes=136  
        Combine input records=0  
        Combine output records=0  
        Reduce input groups=8908  
        Reduce shuffle bytes=453628  
        Reduce input records=13986  
        Reduce output records=8908  
        Spilled Records=27972  
        Shuffled Maps =1  
        Failed Shuffles=0  
        Merged Map outputs=1  
        GC time elapsed (ms)=10  
        Total committed heap usage (bytes)=352976896  
    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=2414099  
    File Output Format Counters   
        Bytes Written=250879*

## Large Number of Small Size Files

### Input

|  |  |  |  |
| --- | --- | --- | --- |
| **File Count** | **Approx. File Size (MB)** | **Total File Size** | **Command** |
| 1341 | 1 MB | 2.2 GB | $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceInput /MapreduceOutputMultipleRohini 5 |

### Block Distribution

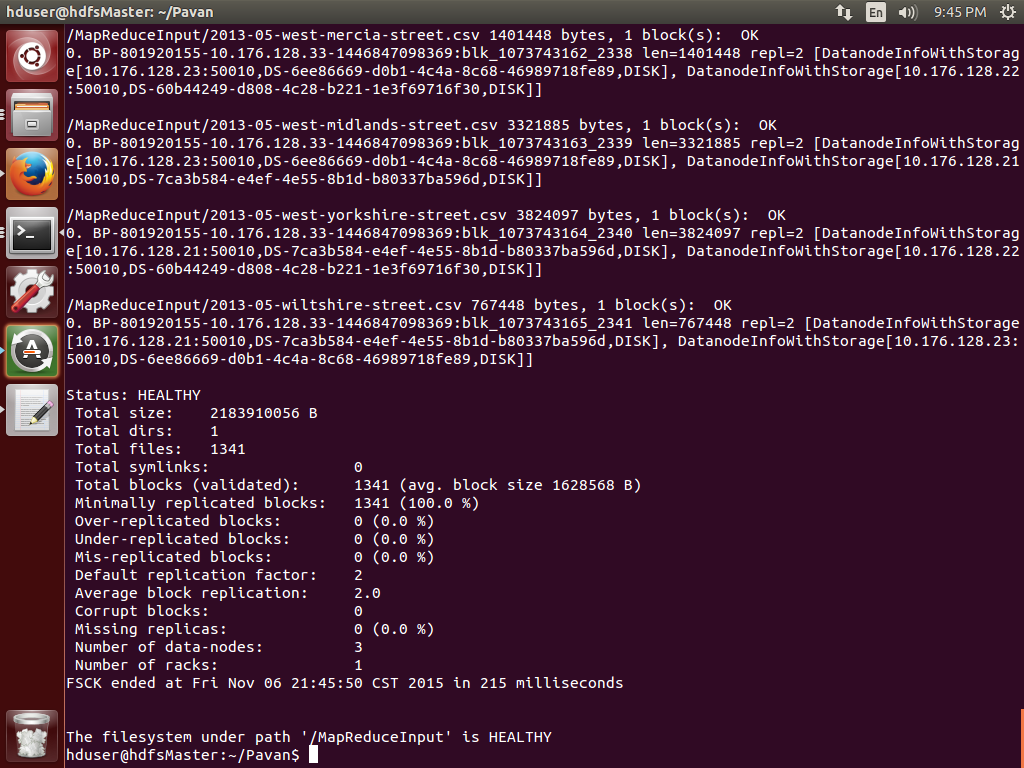


Figure Block Distribution of 1341 files with small size

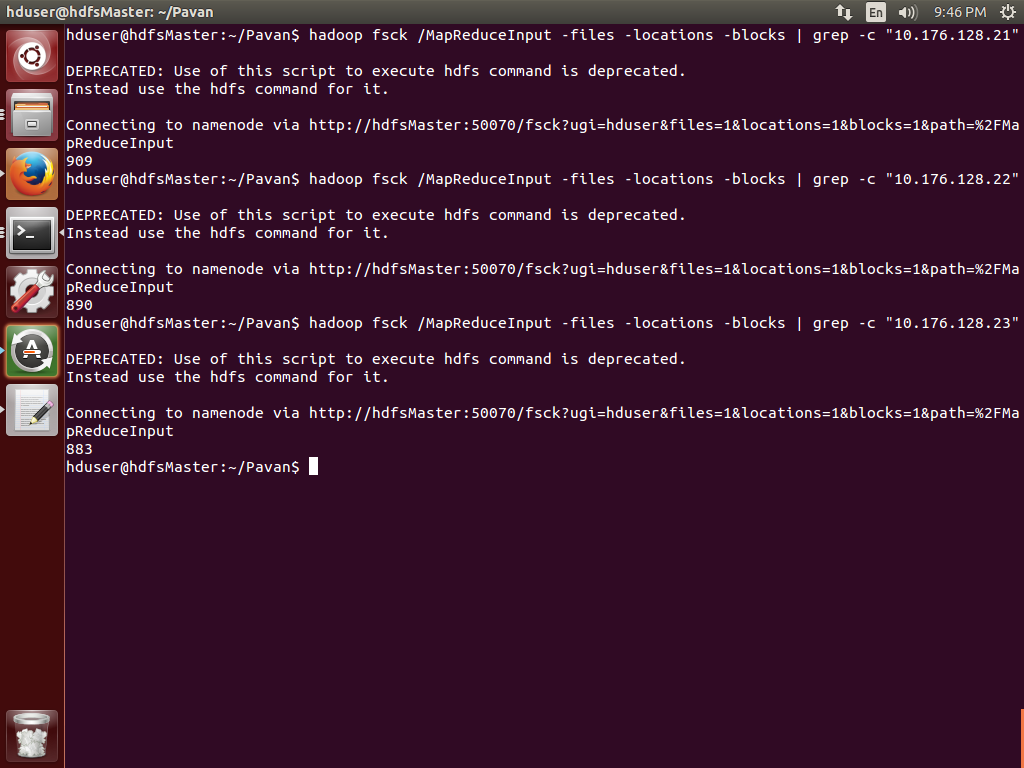


Figure Block Distribution for 1341 files

* Each file is less than 128 MB hence the blocks that are distributed over the datanodes are the number of files: 1341.
* The blocks of these 1341 files distributed over the 3 datanodes in the following way:

|  |  |  |
| --- | --- | --- |
| **No.** | **DataNode IP Address** | **No. of blocks** |
| 1 | 10.176.128.21 | 909 |
| 2 | 10.176.128.22 | 890 |
| 3 | 10.176.128.23 | 883 |

### Output

**Mapper and Reducer Count**

|  |  |
| --- | --- |
| **Mapper Count** | 1314 |
| **Reducer count** | 1 |

**Time required for region digits**

|  |  |  |
| --- | --- | --- |
| **Command** | **Region digits** | **Total time (sec.)** |
| $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceMultipleRohini /MapreduceOutputMultipleRohini 1 | 1 | 445 |
| $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceMultipleRohini /MapreduceOutputMultipleRohini 3 | 3 | 451 |
| $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceMultipleRohini  /MapreduceOutputMultipleRohini 5 | 5 | 461 |

**Output after running the command:**

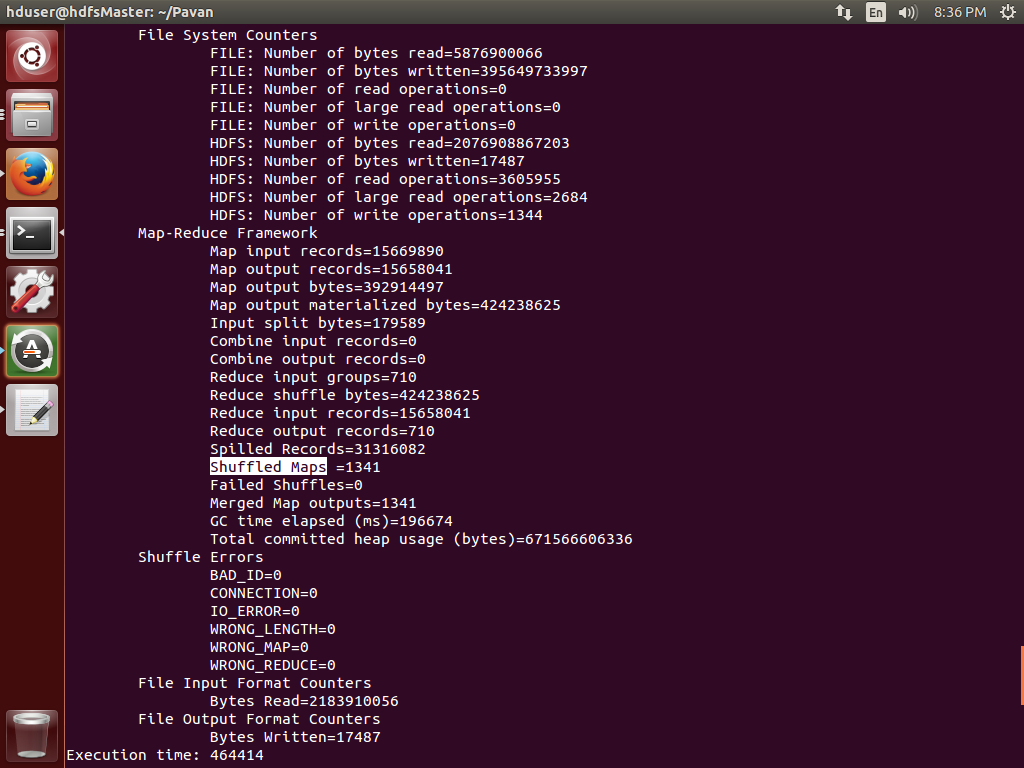
****

Figure Output of 1341 files

## Single Large File

### Input File

|  |  |  |
| --- | --- | --- |
| **File Name** | **File Size** | **Command** |
| largeFile.csv | 2 GB | $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceMultipleRohini/largeFile.csv  /MapreduceOutputMultipleRohini 5 |

### Block Distribution

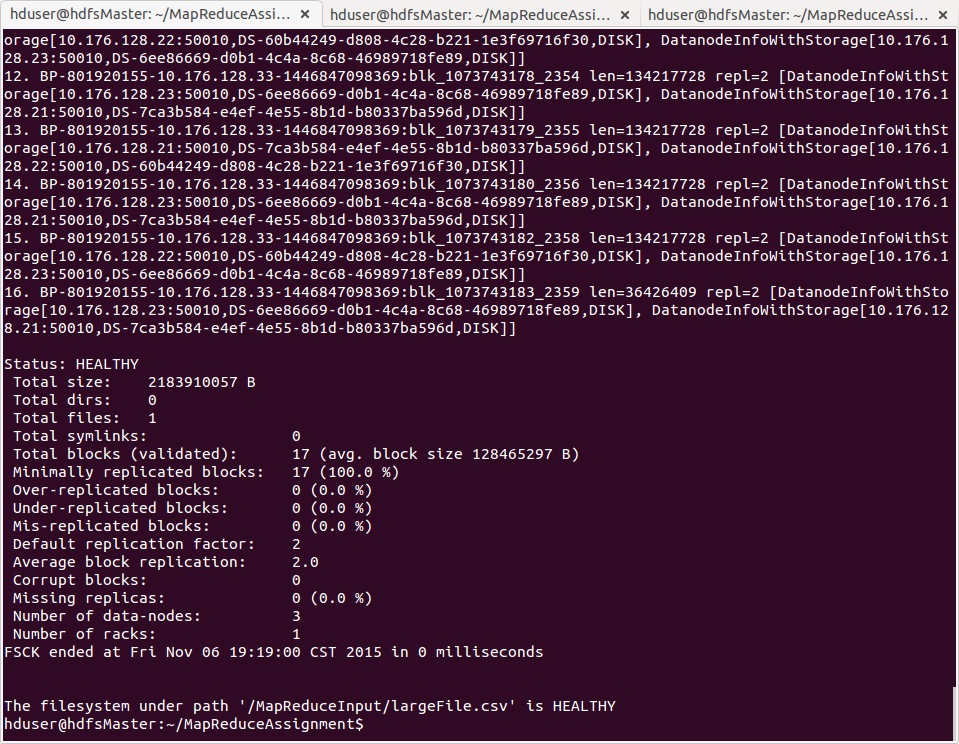


Figure 5 Block Distribution of large file

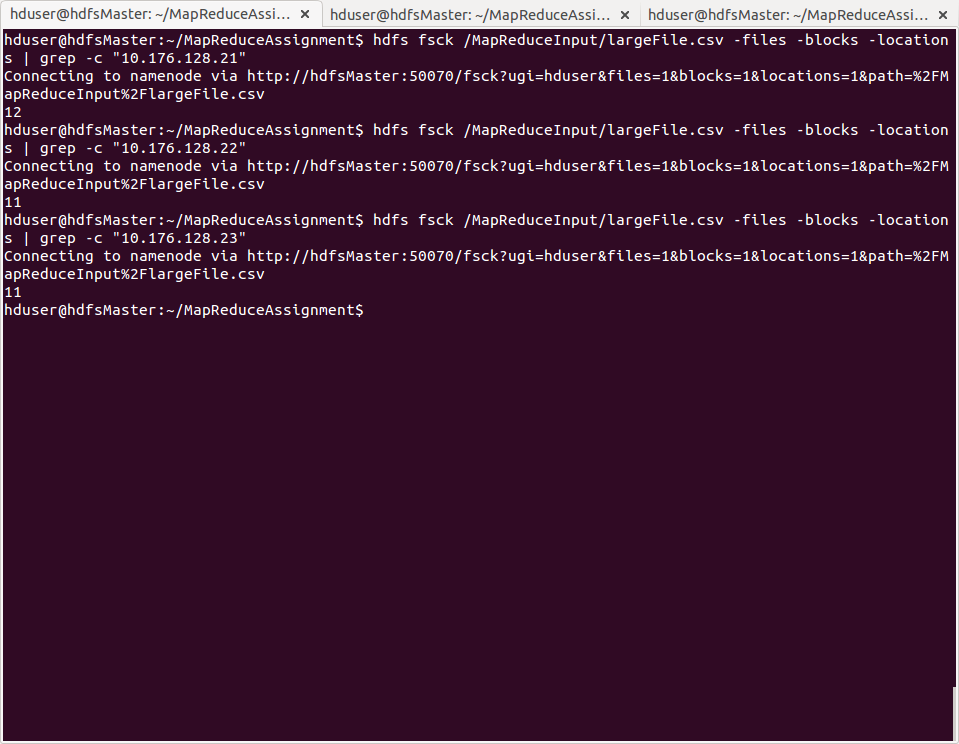


Figure 6 Block Distribution of large file

* The blocks of this file is distributed over the 3 datanodes

|  |  |  |
| --- | --- | --- |
| **No.** | **DataNode IP Address** | **No. of blocks** |
| 1 | 10.176.128.21 | 12 |
| 2 | 10.176.128.22 | 11 |
| 3 | 10.176.128.23 | 11 |

### Output

**Mapper and Reducer Count**

|  |  |
| --- | --- |
| **Mapper Count** | 17 |
| **Reducer count** | 1 |

**Time required for region digits**

|  |  |  |
| --- | --- | --- |
| **Command** | **Region digits** | **Total time (sec.)** |
| $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceInput/largeFile.csv  /MapreduceOutputMultipleRohini 1 | 1 | 220 |
| $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceInput/largeFile.csv  /MapreduceOutputMultipleRohini 3 | 3 | 229 |
| $ hadoop jar HadoopCrimeCount.jar CrimeCountDriver /MapReduceInput/largeFile.csv  /MapreduceOutputMultipleRohini 5 | 5 | 234 |

**Output after running the command:**

*File System Counters  
        FILE: Number of bytes read=906645926  
        FILE: Number of bytes written=5424739931  
        FILE: Number of read operations=0  
        FILE: Number of large read operations=0  
        FILE: Number of write operations=0  
        HDFS: Number of bytes read=22622119250  
        HDFS: Number of bytes written=204592  
        HDFS: Number of read operations=397  
        HDFS: Number of large read operations=0  
        HDFS: Number of write operations=20  
    Map-Reduce Framework  
        Map input records=15669890  
        Map output records=15565136  
        Map output bytes=422004820  
        Map output materialized bytes=453135194  
        Input split bytes=1989  
        Combine input records=0  
        Combine output records=0  
        Reduce input groups=7841  
        Reduce shuffle bytes=453135194  
        Reduce input records=15565136  
        Reduce output records=7841  
        Spilled Records=31130272  
        Shuffled Maps =17  
        Failed Shuffles=0  
        Merged Map outputs=17  
        GC time elapsed (ms)=2505  
        Total committed heap usage (bytes)=*[*3519283200*](tel:3519283200) *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=*[*2183975593*](tel:2183975593) *File Output Format Counters   
        Bytes Written=204592*

# Error Handling

**Error in data**

The input lines of the file containing lines with incomplete data are ignored. This is validated by checking the number of tokens after the line is split. The lines not containing the data in the East and North location are also ignored while processing in the mapper.

**Error in cluster**

The data is replicated on the datanodes for keeping the data available in case of any failures. The replication factor in this cluster is 2. If a node fails then the tasks related to data present on that node will be redirected to other nodes containing the replica of the data.

Following properties can be configured to set the number of times an exception can be thrown before the node is completely stopped.

For Map tasks: *mapreduce.map.maxattempts*  
For reducer tasks: *mapreduce.reduce.maxattempts*

# Observations

* Depending on the size of the input files the MapReducer will split the input chunks into one or more parts.

|  |  |
| --- | --- |
| File Size | Splits |
| Small File - 2 MB | 1 |
| Multiple small size files (1341 number of files) | 1341 |
| Large File – 2 GB | 17 |

* The number of mappers is dependent on the number of splits and the reducer required is just one.
* As the region digit is increased the numbers of keys are increased and hence the time required for 5 digit of region is more as compared to 1 and 3.
* MapReduce Frameworks variables:
  + Reduce shuffle bytes: this is the shuffled map output on the reducer, which was obtained through the TaskTracker using the intermediate files of Map phase
  + Spilled Records: when the buffer memory gets filled and no space is left for new records, old records are written to the disk.
  + Map output bytes: Output bytes from the map phase.
  + Combine input records: Records merged by the combiner during the map phase.
  + Reduce input records: Number of values associated to the keys from the mapper phase.
  + Reduce input groups: Number of unique keys fed to reducer from mappers.
  + Reduce output records: The number of records combined by all reducers.
  + Combine output records: No of records combined during the reduce phase.
* The time required for the same map and reduce program for processing large number of small files is more than the time required for the combined large file. This happens because now the block size is 128 MB. So, there are certain files which are occupying less than a single block. Thus, there are many unoccupied spaces in the blocks. Hence, HDFS is recommended for small number of large files.