Hadoop MapReduce Join & Counter with Example

What is a Join in MapReduce?

A join operation is used to combine two large datasets in MapReduce. However, this process involves writing lots of code to perform the actual join operation.

Joining of two datasets begins by comparing the size of each dataset. If one dataset is smaller as compared to the other dataset then smaller dataset is distributed to every data node in the cluster. Once it is distributed, either Mapper or Reducer uses the smaller dataset to perform a lookup for matching records from the large dataset and then combine those records to form output records.

In this tutorial, you will learn-

Types of Join

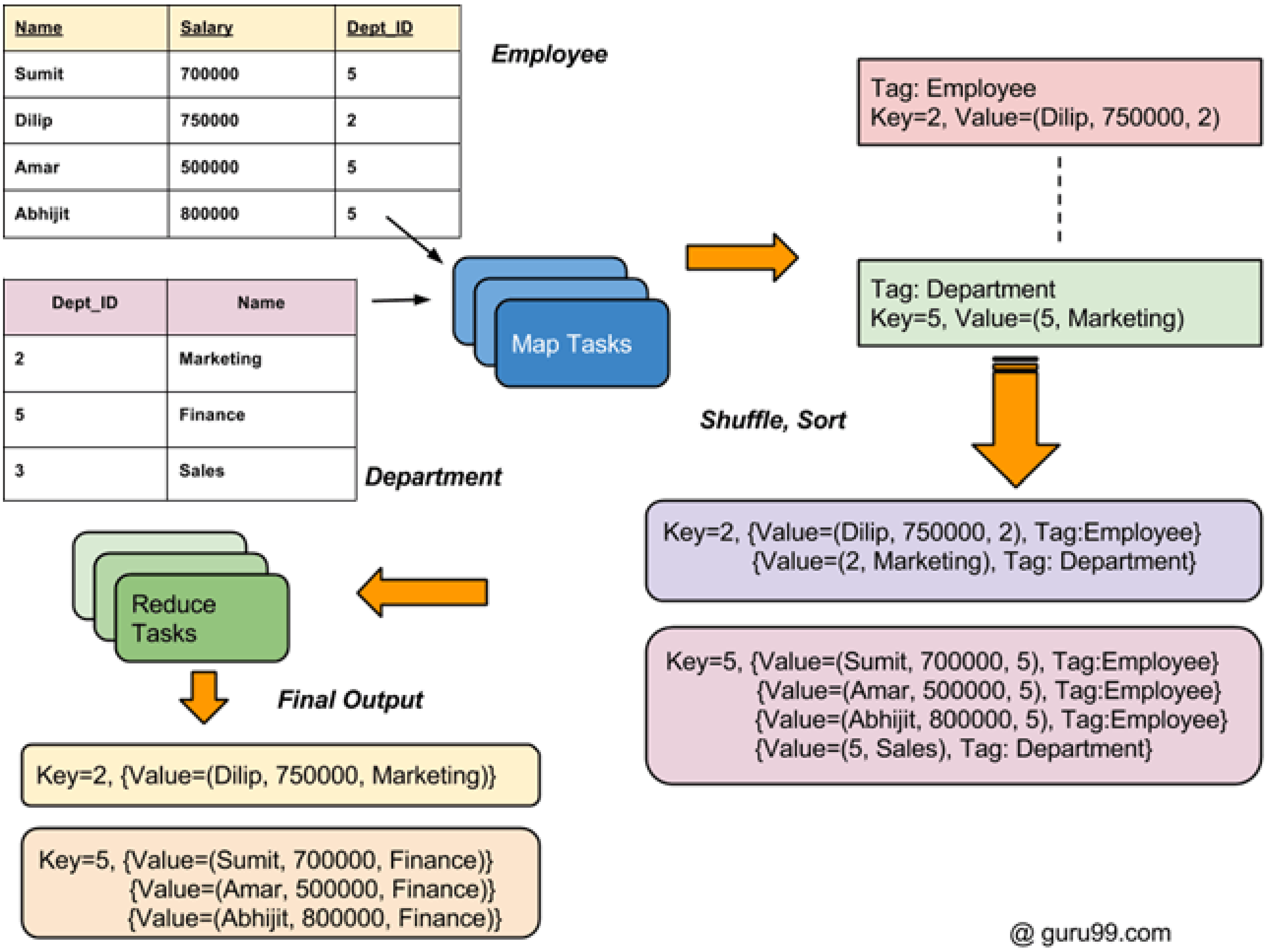
Depending upon the place where the actual join is performed, this join is classified into-

1. Map-side join - When the join is performed by the mapper, it is called as map-side join. In this type, the join is performed before data is actually consumed by the map function. It is mandatory that the input to each map is in the form of a partition and is in sorted order. Also, there must be an equal number of partitions and it must be sorted by the join key.

2. Reduce-side join - When the join is performed by the reducer, it is called as reduce-side join. There is no necessity in this join to have a dataset in a structured form (or partitioned).

Here, map side processing emits join key and corresponding tuples of both the tables. As an effect of this processing, all the tuples with same join key fall into the same reducer which then joins the records with same join key.

An overall process flow is depicted in below diagram.



Name	Salary	Dept_ID
Sumit	700000	5
Dilip	750000	2
Amar	500000	5
Abhijit	800000	5

Dept_ID	Name
2	Marketing
5	Finance
3	Sales

Tag: Employee
Key=2, Value=(Dilip, 750000, 2)

Tag: Department
Key=5, Value=(5, Marketing)

Key=2, {Value=(Dilip, 750000, 2), Tag:Employee}
{Value=(2, Marketing), Tag: Department}

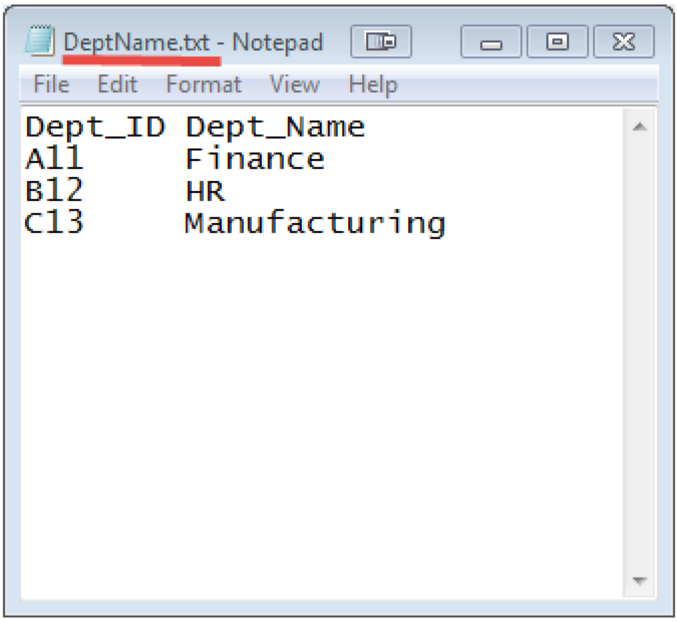
Key=5, {Value=(Sumit, 700000, 5), Tag:Employee}
{Value=(Amar, 500000, 5), Tag:Employee}
{Value=(Abhijit, 800000, 5), Tag:Employee}
{Value=(5, Sales), Tag: Department}

Key=2, {Value=(Dilip, 750000, Marketing)}

Key=5, {Value=(Sumit, 700000, Finance)}
{Value=(Amar, 500000, Finance)}
{Value=(Abhijit, 800000, Finance)}

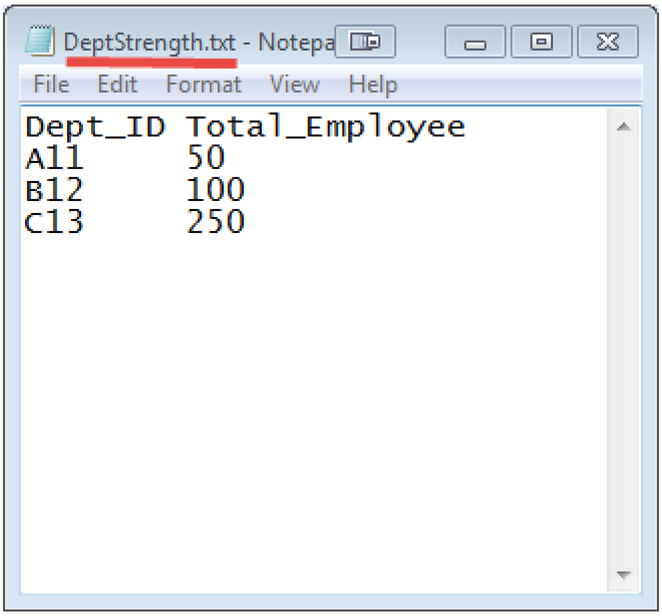
How to Join two DataSets: MapReduce Example

There are two Sets of Data in two Different Files (shown below). The Key Dept_ID is common in both files. The goal is to use MapReduce Join to combine these files



```
Dept_ID Dept_Name
A11     Finance
B12     HR
C13     Manufacturing
```

File 1



```
Dept_ID Total_Employee
A11     50
B12     100
C13     250
```

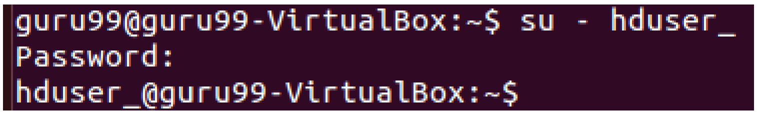
File 2

Input: The input data set is a txt file, **DeptName.txt & DepStrength.txt**

Download Input Files From Here

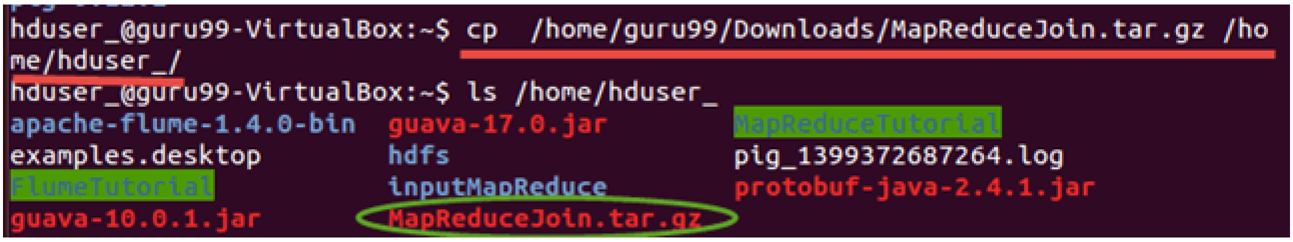
Ensure you have Hadoop installed. Before you start with the actual process, change user to 'hduser' (id used while Hadoop configuration, you can switch to the userid used during your Hadoop config).

```
su - hduser_
```



```
guru99@guru99-VirtualBox:~$ su - hduser_
Password:
hduser_@guru99-VirtualBox:~$
```

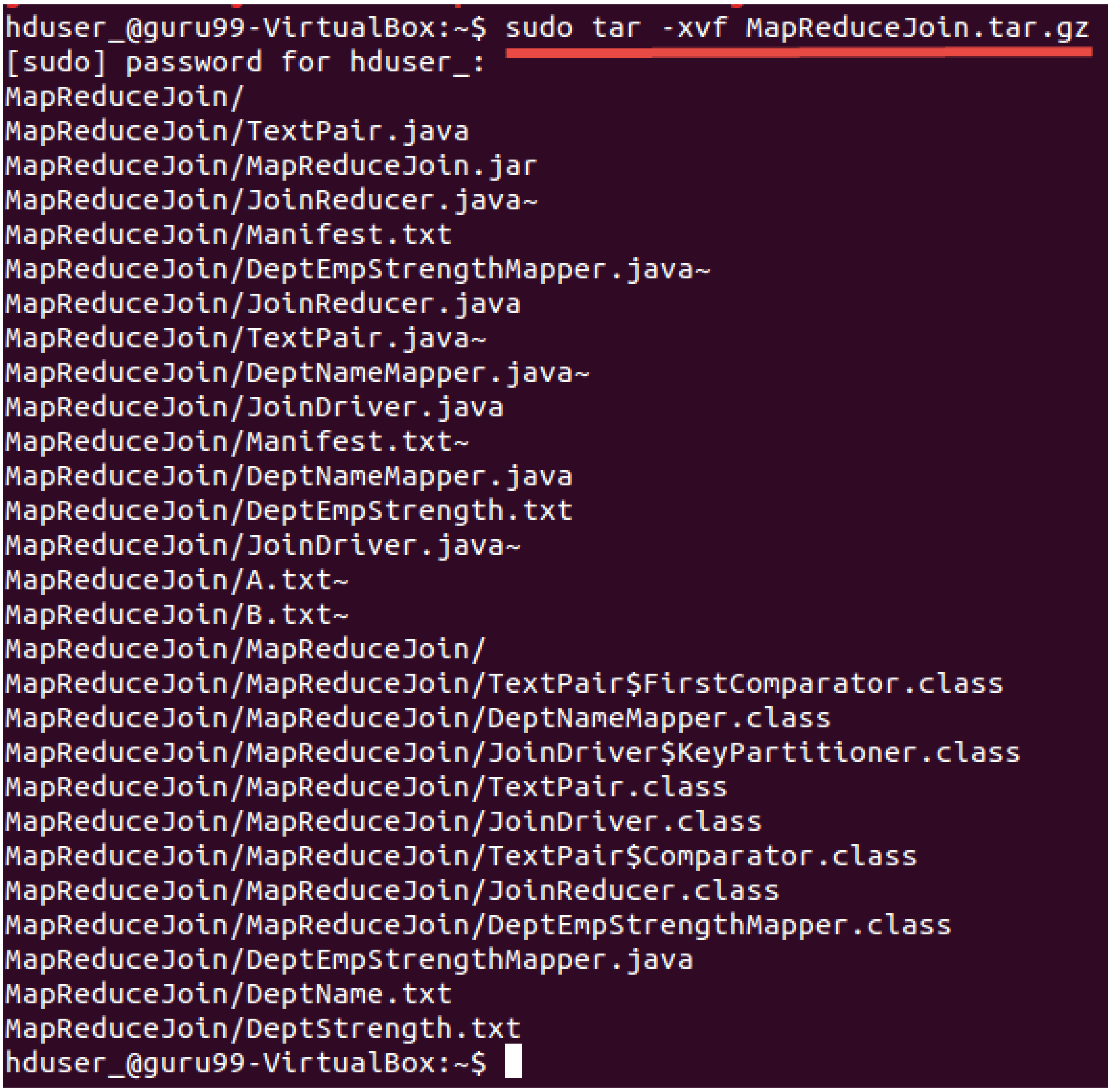
Step 1) Copy the zip file to the location of your choice



```
hduser_@guru99-VirtualBox:~$ cp  /home/guru99/Downloads/MapReduceJoin.tar.gz /ho
me/hduser_/
hduser_@guru99-VirtualBox:~$ ls /home/hduser_
apache-flume-1.4.0-bin  guava-17.0.jar          MapReduceTutorial
examples.desktop        hdfs                    pig_1399372687264.log
FlumeTutorial           inputMapReduce          protobuf-java-2.4.1.jar
guava-10.0.1.jar        MapReduceJoin.tar.gz
```

Step 2) Uncompress the Zip File

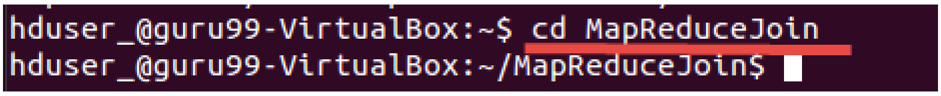
```
sudo tar -xvf MapReduceJoin.tar.gz
```



```
hduser_@guru99-VirtualBox:~$ sudo tar -xvf MapReduceJoin.tar.gz
[sudo] password for hduser_:
MapReduceJoin/
MapReduceJoin/TextPair.java
MapReduceJoin/MapReduceJoin.jar
MapReduceJoin/JoinReducer.java~
MapReduceJoin/Manifest.txt
MapReduceJoin/DeptEmpStrengthMapper.java~
MapReduceJoin/JoinReducer.java
MapReduceJoin/TextPair.java~
MapReduceJoin/DeptNameMapper.java~
MapReduceJoin/JoinDriver.java
MapReduceJoin/Manifest.txt~
MapReduceJoin/DeptNameMapper.java
MapReduceJoin/DeptEmpStrength.txt
MapReduceJoin/JoinDriver.java~
MapReduceJoin/A.txt~
MapReduceJoin/B.txt~
MapReduceJoin/MapReduceJoin/
MapReduceJoin/MapReduceJoin/TextPair$FirstComparator.class
MapReduceJoin/MapReduceJoin/DeptNameMapper.class
MapReduceJoin/MapReduceJoin/JoinDriver$KeyPartitioner.class
MapReduceJoin/MapReduceJoin/TextPair.class
MapReduceJoin/MapReduceJoin/JoinDriver.class
MapReduceJoin/MapReduceJoin/TextPair$Comparator.class
MapReduceJoin/MapReduceJoin/JoinReducer.class
MapReduceJoin/MapReduceJoin/DeptEmpStrengthMapper.class
MapReduceJoin/DeptEmpStrengthMapper.java
MapReduceJoin/DeptName.txt
MapReduceJoin/DeptStrength.txt
hduser_@guru99-VirtualBox:~$
```

Step 3) Go to directory MapReduceJoin/

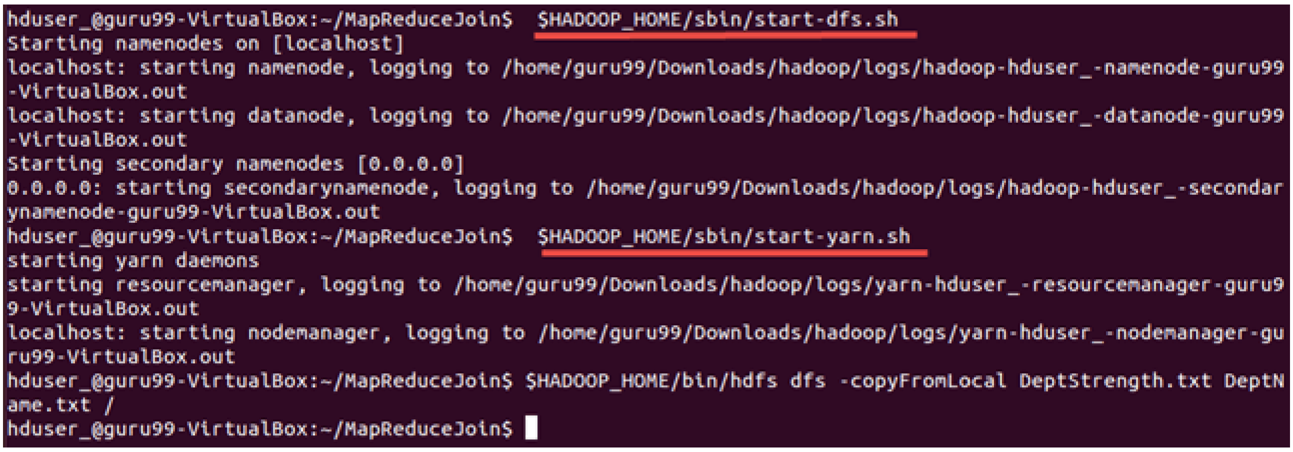
```
cd MapReduceJoin/
```



```
hduser_@guru99-VirtualBox:~$ cd MapReduceJoin
hduser_@guru99-VirtualBox:~/MapReduceJoin$ 
```

Step 4) Start Hadoop

```
$HADOOP_HOME/sbin/start-dfs.sh
$HADOOP_HOME/sbin/start-yarn.sh
```

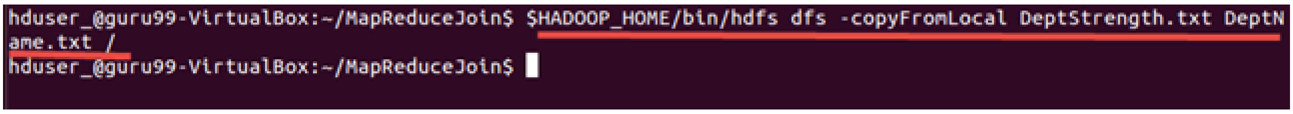


```
hduser_@guru99-VirtualBox:~/MapReduceJoin$  $HADOOP_HOME/sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/guru99/Downloads/hadoop/logs/hadoop-hduser_-namenode-guru99
-VirtualBox.out
localhost: starting datanode, logging to /home/guru99/Downloads/hadoop/logs/hadoop-hduser_-datanode-guru99
-VirtualBox.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/guru99/Downloads/hadoop/logs/hadoop-hduser_-secondar
ynamenode-guru99-VirtualBox.out
hduser_@guru99-VirtualBox:~/MapReduceJoin$  $HADOOP_HOME/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/guru99/Downloads/hadoop/logs/yarn-hduser_-resourcemanager-guru9
9-VirtualBox.out
localhost: starting nodemanager, logging to /home/guru99/Downloads/hadoop/logs/yarn-hduser_-nodemanager-gu
ru99-VirtualBox.out
hduser_@guru99-VirtualBox:~/MapReduceJoin$ $HADOOP_HOME/bin/hdfs dfs -copyFromLocal DeptStrength.txt DeptN
ame.txt /
hduser_@guru99-VirtualBox:~/MapReduceJoin$ 
```

Step 5) DeptStrength.txt and DeptName.txt are the input files used for this program.

These file needs to be copied to HDFS using below command-

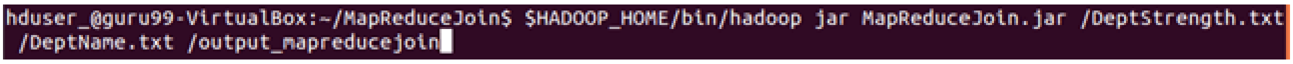
```
$HADOOP_HOME/bin/hdfs dfs -copyFromLocal DeptStrength.txt DeptName.txt /
```



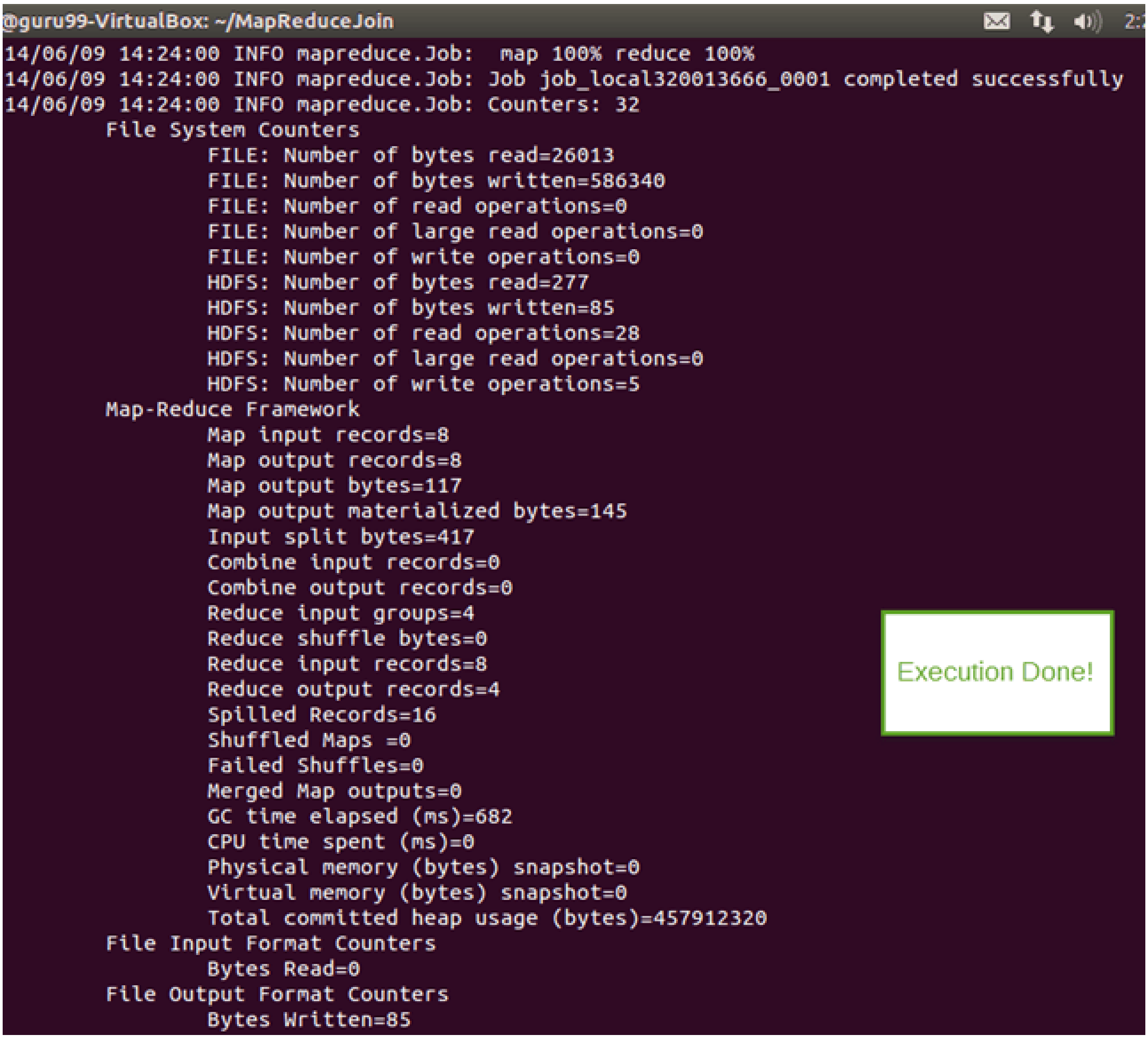
```
hduser_@guru99-VirtualBox:~/MapReduceJoin$ $HADOOP_HOME/bin/hdfs dfs -copyFromLocal DeptStrength.txt DeptN
ame.txt /
hduser_@guru99-VirtualBox:~/MapReduceJoin$ 
```

Step 6) Run the program using below command-

```
$HADOOP_HOME/bin/hadoop jar MapReduceJoin.jar MapReduceJoin/JoinDriver/DeptStrengt
h.txt /DeptName.txt /output_mapreducejoin
```



```
hduser_@guru99-VirtualBox:~/MapReduceJoin$ $HADOOP_HOME/bin/hadoop jar MapReduceJoin.jar /DeptStrength.txt
 /DeptName.txt /output_mapreducejoin
```



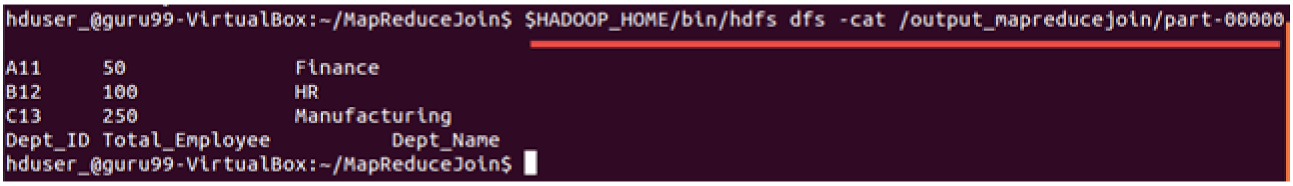
```
@guru99-VirtualBox: ~/MapReduceJoin
14/06/09 14:24:00 INFO mapreduce.Job:  map 100% reduce 100%
14/06/09 14:24:00 INFO mapreduce.Job: Job job_local320013666_0001 completed successfully
14/06/09 14:24:00 INFO mapreduce.Job: Counters: 32
        File System Counters
                FILE: Number of bytes read=26013
                FILE: Number of bytes written=586340
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=277
                HDFS: Number of bytes written=85
                HDFS: Number of read operations=28
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=5
        Map-Reduce Framework
                Map input records=8
                Map output records=8
                Map output bytes=117
                Map output materialized bytes=145
                Input split bytes=417
                Combine input records=0
                Combine output records=0
                Reduce input groups=4
                Reduce shuffle bytes=0
                Reduce input records=8
                Reduce output records=4
                Spilled Records=16
                Shuffled Maps =0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=682
                CPU time spent (ms)=0
                Physical memory (bytes) snapshot=0
                Virtual memory (bytes) snapshot=0
                Total committed heap usage (bytes)=457912320
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=85
```

Execution Done!

Step 7) After execution, output file (named 'part-00000') will stored in the directory /output_mapreducejoin on HDFS

Results can be seen using the command line interface

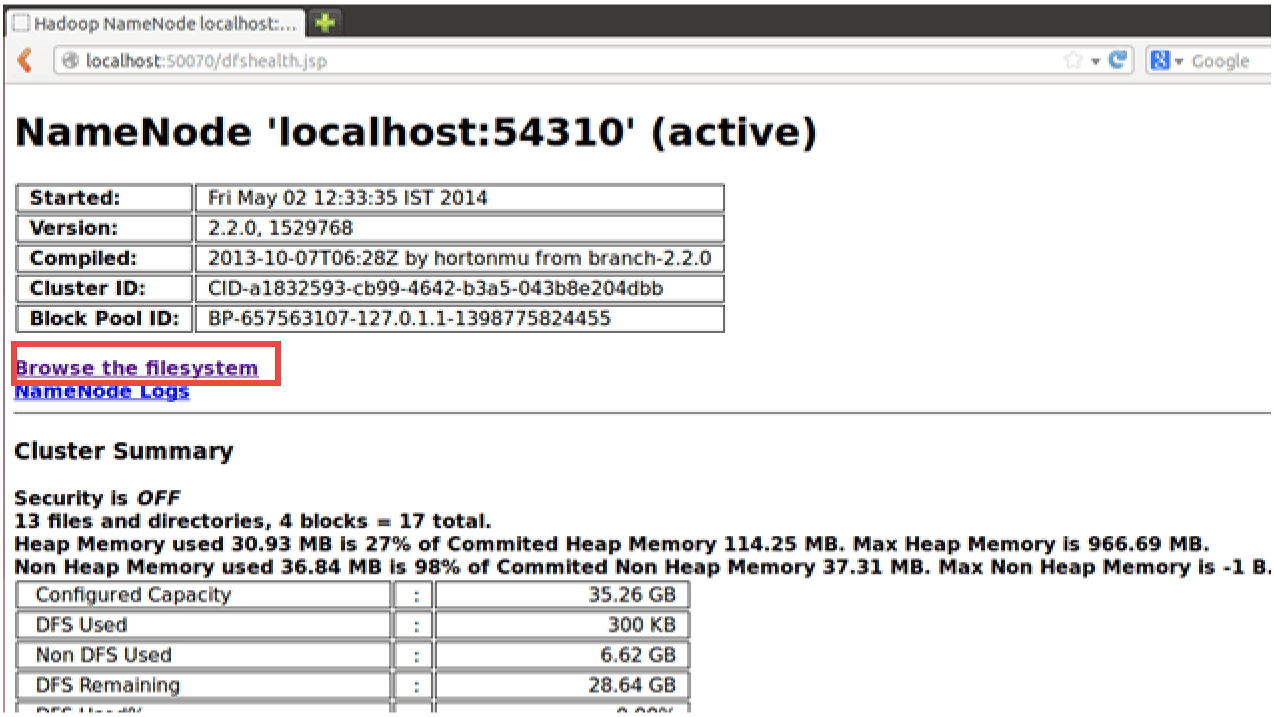
```
$HADOOP_HOME/bin/hdfs dfs -cat /output_mapreducejoin/part-00000
```



```
hduser_@guru99-VirtualBox:~/MapReduceJoin$ $HADOOP_HOME/bin/hdfs dfs -cat /output_mapreducejoin/part-00000

A11     50              Finance
B12     100             HR
C13     250             Manufacturing
Dept_ID Total_Employee          Dept_Name
hduser_@guru99-VirtualBox:~/MapReduceJoin$ 
```

Results can also be seen via a web interface as-



NameNode 'localhost:54310' (active)

Started:	Fri May 02 12:33:35 IST 2014
Version:	2.2.0, 1529768
Compiled:	2013-10-07T06:28Z by hortonmu from branch-2.2.0
Cluster ID:	CID-a1832593-cb99-4642-b3a5-043b8e204dbb
Block Pool ID:	BP-657563107-127.0.1.1-1398775824455

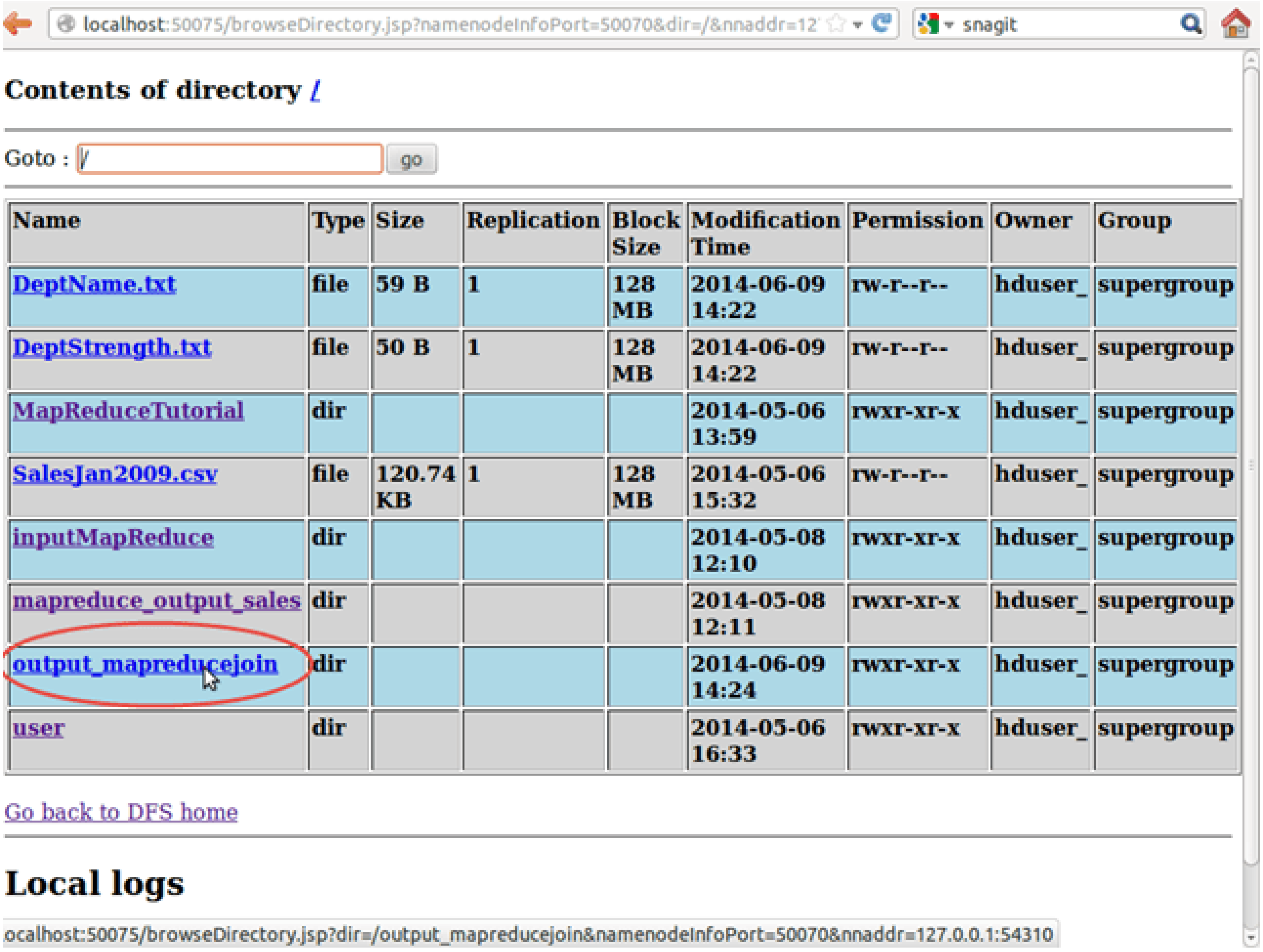
Browse the filesystem
NameNode Logs

Cluster Summary

Security is *OFF*
13 files and directories, 4 blocks = 17 total.
Heap Memory used 30.93 MB is 27% of Commited Heap Memory 114.25 MB. Max Heap Memory is 966.69 MB.
Non Heap Memory used 36.84 MB is 98% of Commited Non Heap Memory 37.31 MB. Max Non Heap Memory is -1 B.

Configured Capacity	:	35.26 GB
DFS Used	:	300 KB
Non DFS Used	:	6.62 GB
DFS Remaining	:	28.64 GB

Now select **'Browse the filesystem'** and navigate upto **/output_mapreducejoin**


localhost:50075/browseDirectory.jsp?namenodeInfoPort=50070&dir=/&nnaddr=12
snagit

Contents of directory /

Goto : / go

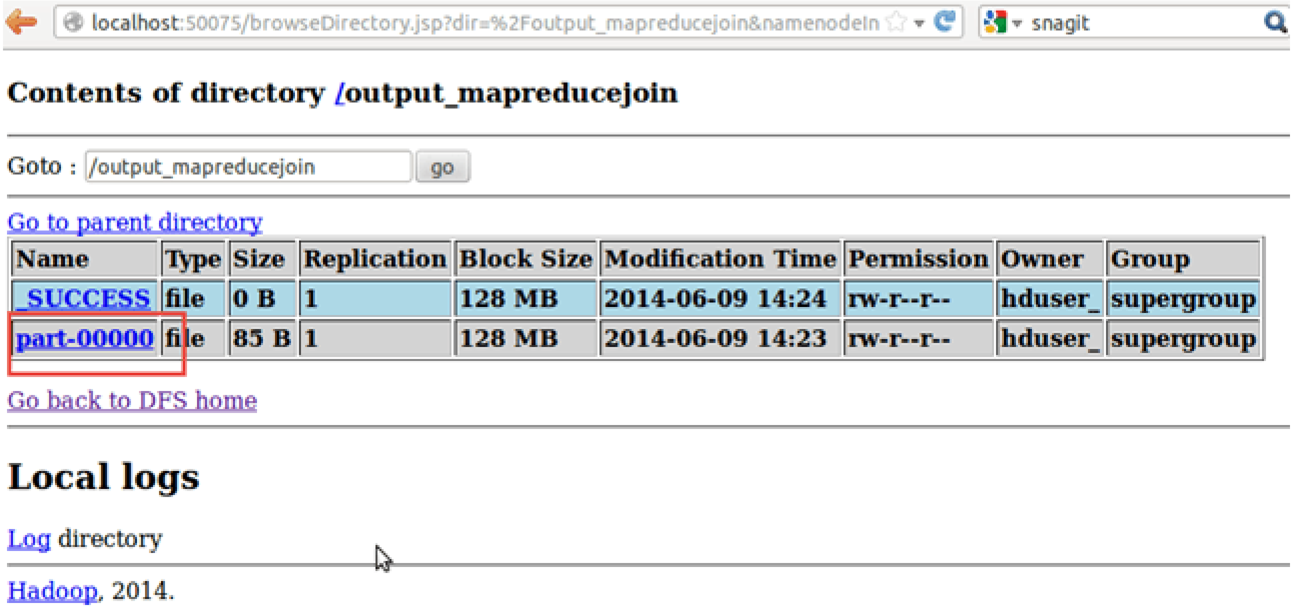
Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
DeptName.txt	file	59 B	1	128 MB	2014-06-09 14:22	rw-r--r--	hduser_	supergroup
DeptStrength.txt	file	50 B	1	128 MB	2014-06-09 14:22	rw-r--r--	hduser_	supergroup
MapReduceTutorial	dir				2014-05-06 13:59	rwxr-xr-x	hduser_	supergroup
SalesJan2009.csv	file	120.74 KB	1	128 MB	2014-05-06 15:32	rw-r--r--	hduser_	supergroup
inputMapReduce	dir				2014-05-08 12:10	rwxr-xr-x	hduser_	supergroup
mapreduce_output_sales	dir				2014-05-08 12:11	rwxr-xr-x	hduser_	supergroup
output_mapreducejoin	dir				2014-06-09 14:24	rwxr-xr-x	hduser_	supergroup
user	dir				2014-05-06 16:33	rwxr-xr-x	hduser_	supergroup

Go back to DFS home

Local logs

ocalhost:50075/browseDirectory.jsp?dir=/output_mapreducejoin&namenodeInfoPort=50070&nnaddr=127.0.0.1:54310

Open **part-r-00000**



localhost:50075/browseDirectory.jsp?dir=%2Foutput_mapreducejoin&namenodeIn
snagit

Contents of directory /output_mapreducejoin

Goto : /output_mapreducejoin go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
_SUCCESS	file	0 B	1	128 MB	2014-06-09 14:24	rw-r--r--	hduser_	supergroup
part-00000	file	85 B	1	128 MB	2014-06-09 14:23	rw-r--r--	hduser_	supergroup

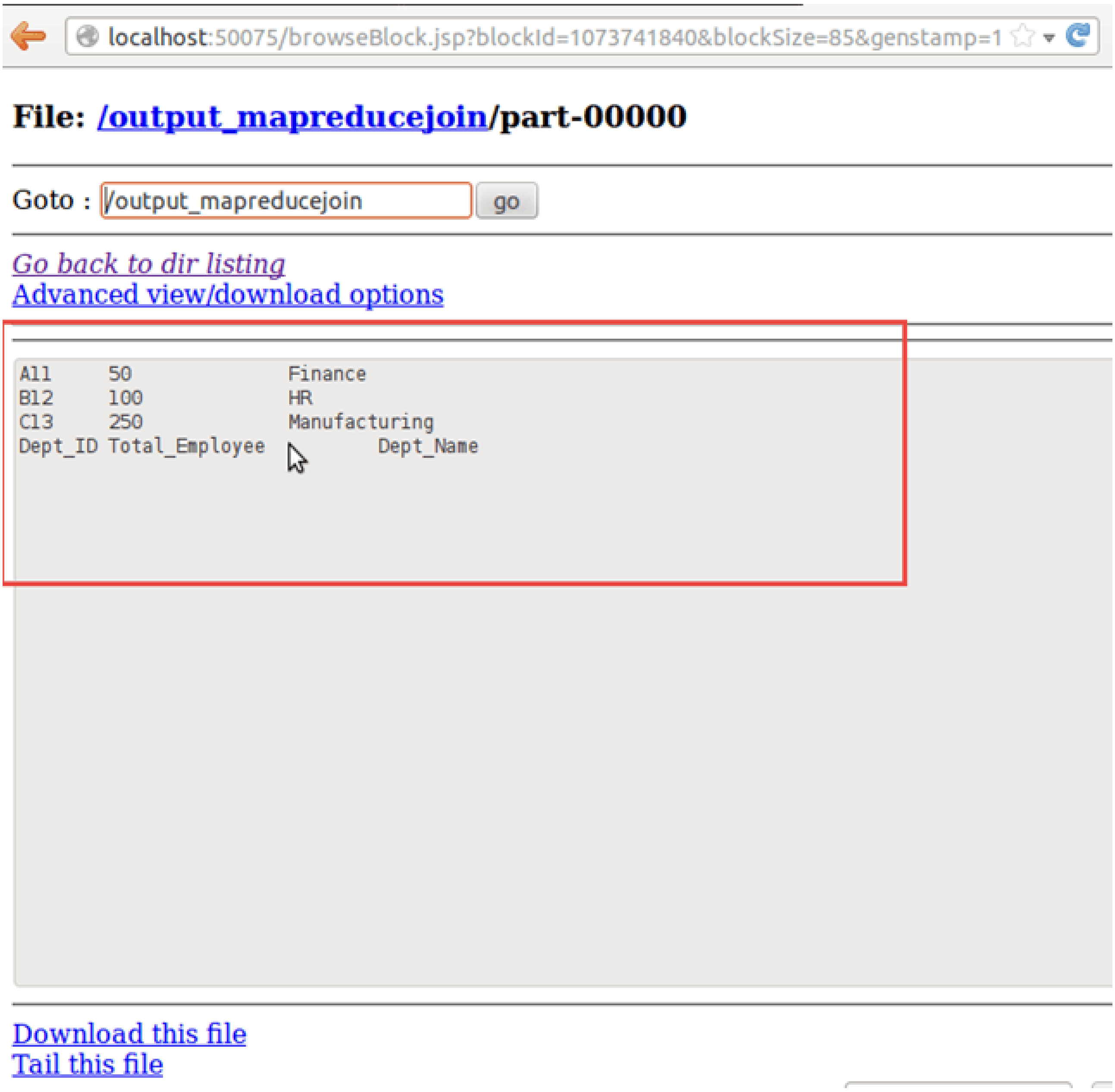
Go back to DFS home

Local logs

Log directory

Hadoop, 2014.

Results are shown



localhost:50075/browseBlock.jsp?blockId=1073741840&blockSize=85&genstamp=1
File: /output_mapreducejoin/part-00000
Goto : /output_mapreducejoin
go
Go back to dir listing
Advanced view/download options

```
A11      50              Finance
B12      100             HR
C13      250             Manufacturing
Dept_ID Total_Employee          Dept_Name
```

Download this file
Tail this file

NOTE: Please note that before running this program for the next time, you will need to delete output directory /output_mapreducejoin

```
$HADOOP_HOME/bin/hdfs dfs -rm -r /output_mapreducejoin
```

Alternative is to use a different name for the output directory.