

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN  
KHOA KHOA HỌC MÁY TÍNH

LAB 1 - HADOOP MAPREDUCE



**UIT**  
TRƯỜNG ĐẠI HỌC  
CÔNG NGHỆ THÔNG TIN

Lớp: DS200.P21

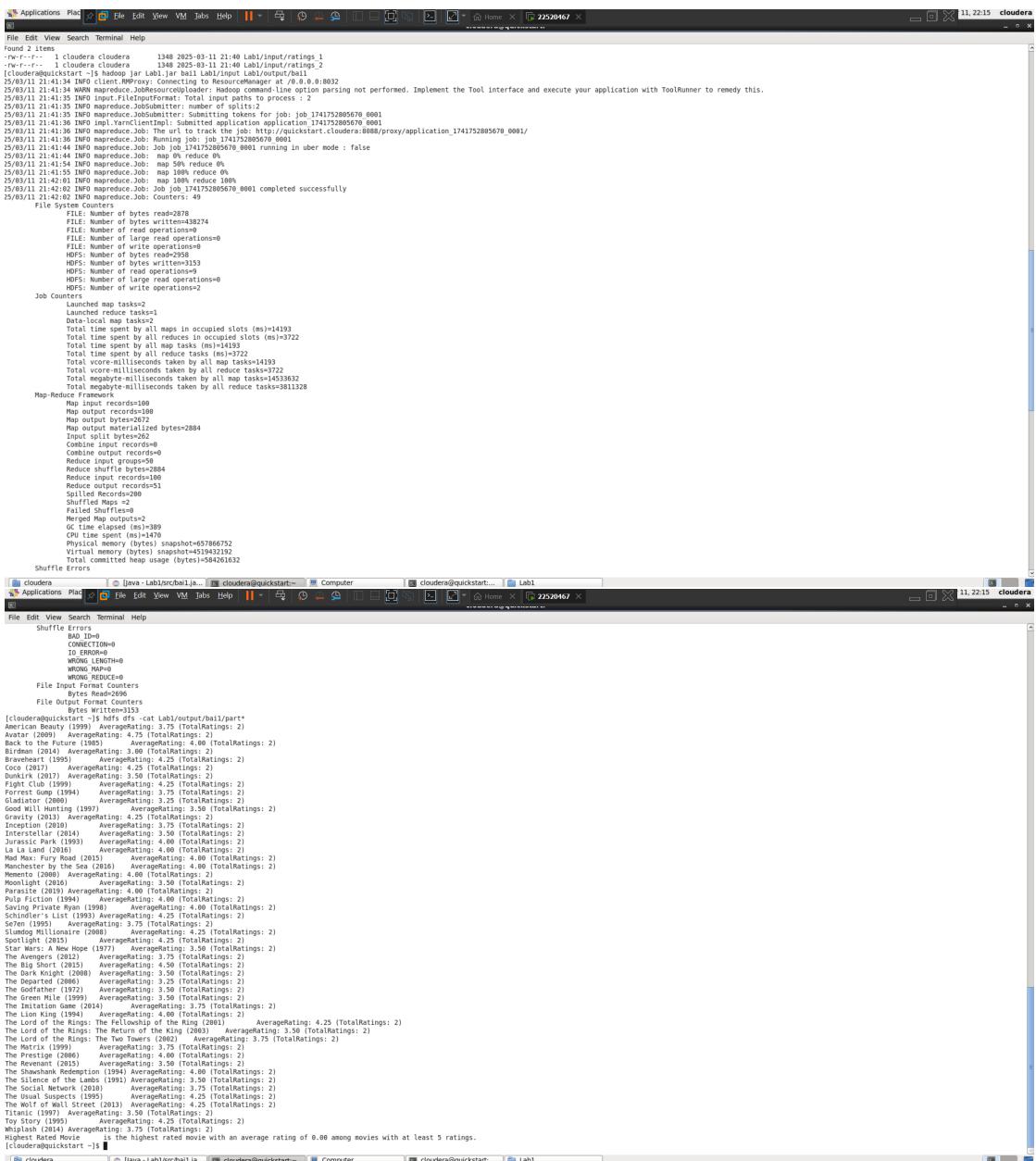
Họ và tên	MSSV
Nguyễn Duy Hoàng	22520467

## Mục lục

<b>1. Bài 1</b>	<b>1</b>
1.1. Minh chứng . . . . .	1
1.2. Code . . . . .	1
<b>2. Bài 2</b>	<b>3</b>
2.1. Minh chứng . . . . .	3
2.2. Code . . . . .	4
<b>3. Bài 3</b>	<b>6</b>
3.1. Minh chứng . . . . .	6
3.2. Code . . . . .	6
<b>4. Bài 4</b>	<b>8</b>
3.1. Minh chứng . . . . .	8
4.2. Code . . . . .	9

# 1. Bài 1

## 1.1. Minh chứng



The terminal window displays the contents of the HDFS file 'Lab1'. The file contains a list of movie titles and their average ratings, sorted by rating. The output is as follows:

```
[cloudera@quickstart ~]$ hdfs dfs -cat Lab1/input/lab1/part*
American Beauty (1999) AverageRating: 3.75 (TotalRatings: 2)
Avatar (2009) AverageRating: 4.75 (TotalRatings: 2)
Back to the Future (1985) AverageRating: 4.00 (TotalRatings: 2)
Braveheart (1995) AverageRating: 4.25 (TotalRatings: 2)
Coco Before Chanel (2009) AverageRating: 3.50 (TotalRatings: 2)
Dunkirk (2017) AverageRating: 4.25 (TotalRatings: 2)
Flight Club (1999) AverageRating: 4.25 (TotalRatings: 2)
Forrest Gump (1994) AverageRating: 4.00 (TotalRatings: 2)
Gladiator (2008) AverageRating: 3.25 (TotalRatings: 2)
Good Will Hunting (1997) AverageRating: 3.50 (TotalRatings: 2)
Great Expectations (1946) AverageRating: 3.00 (TotalRatings: 2)
Inception (2010) AverageRating: 3.75 (TotalRatings: 2)
Interstellar (2014) AverageRating: 3.50 (TotalRatings: 2)
Jurassic Park (1993) AverageRating: 3.50 (TotalRatings: 2)
La La Land (2016) AverageRating: 4.00 (TotalRatings: 2)
Mad Max: Fury Road (2015) AverageRating: 4.00 (TotalRatings: 2)
Manchester by the Sea (2016) AverageRating: 4.00 (TotalRatings: 2)
Memento (2000) AverageRating: 4.00 (TotalRatings: 2)
House of Cards (2013) AverageRating: 3.50 (TotalRatings: 2)
Parasite (2019) AverageRating: 4.00 (TotalRatings: 2)
Pulp Fiction (1994) AverageRating: 4.00 (TotalRatings: 2)
Saving Private Ryan (1998) AverageRating: 4.00 (TotalRatings: 2)
Schindler's List (1993) AverageRating: 4.25 (TotalRatings: 2)
Seven (1995) AverageRating: 3.75 (TotalRatings: 2)
Slumdog Millions (2008) AverageRating: 3.25 (TotalRatings: 2)
Spotlight (2015) AverageRating: 4.25 (TotalRatings: 2)
Star Wars: Episode Hope (1977) AverageRating: 4.00 (TotalRatings: 2)
The Avengers (2012) AverageRating: 3.75 (TotalRatings: 2)
The Big Short (2013) AverageRating: 4.50 (TotalRatings: 2)
The Dark Knight (2008) AverageRating: 4.50 (TotalRatings: 2)
The Departed (2006) AverageRating: 3.25 (TotalRatings: 2)
The Godfather (1972) AverageRating: 3.50 (TotalRatings: 2)
The Godfather (1990) AverageRating: 3.50 (TotalRatings: 2)
The Godfather (2005) AverageRating: 3.50 (TotalRatings: 2)
The Imitation Game (2014) AverageRating: 3.75 (TotalRatings: 2)
The Lord of the Rings: The Fellowship of the Ring (2001) AverageRating: 4.25 (TotalRatings: 2)
The Lord of the Rings: The Return of the King (2003) AverageRating: 3.50 (TotalRatings: 2)
The Matrix (1999) AverageRating: 4.00 (TotalRatings: 2)
The Prestige (2006) AverageRating: 3.75 (TotalRatings: 2)
The Silence of the Lambs (1991) AverageRating: 3.50 (TotalRatings: 2)
The Social Network (2010) AverageRating: 4.00 (TotalRatings: 2)
The Usual Suspects (1995) AverageRating: 4.25 (TotalRatings: 2)
The War of the Worlds (2005) AverageRating: 3.50 (TotalRatings: 2)
Titanic (1997) AverageRating: 3.50 (TotalRatings: 2)
Toy Story (1995) AverageRating: 4.25 (TotalRatings: 2)
Whiplash (2014) AverageRating: 4.75 (TotalRatings: 2)
Highest Rated Movie: Toy Story (1995) The highest rated movie with an average rating of 4.25 among movies with at least 5 ratings.
```

## 1.2. Code

```
// Import necessary libraries
import java.io.*;
import java.util.*;
import java.net.URI;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.lib.output.FileOutputFormat;
```

```

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

public class bai1 {

    public static class RatingMapper extends Mapper <Object, Text, Text, Text>
    {
        private Map<Integer, String> movieMap = new HashMap<>();

        public void setup(Context context) throws IOException, InterruptedException
        {
            try (BufferedReader br = new BufferedReader(new FileReader("movies"))){
                String line;
                while ((line = br.readLine()) != null){
                    String columns[] = line.split(",");
                    String id = columns[0].trim();
                    String name = columns[1].trim();
                    movieMap.put(Integer.parseInt(id), name);
                }
            }
            catch (IOException e){
                e.printStackTrace();
            }
        }

        public void map(Object key, Text value, Context context) throws IOException, InterruptedException
        {
            String record = value.toString().trim();
            String parts[] = record.split(",");

            String id_movie = parts[1].trim();
            String rating = parts[2].trim();
            String name_movie = movieMap.get(Integer.parseInt(id_movie));
            context.write(new Text(name_movie), new Text(rating));
        }
    }

    public static class RatingReducer extends Reducer <Text, Text, Text, Text>
    {
        String max_rating_movie = "";
        double max_rating = 0.0;

        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
        <-- InterruptedException{
            double sum_rating = 0.0;
            int count_rating = 0;
            double temp_avg = 0.0;
            for (Text val : values) {
                sum_rating += Double.parseDouble(val.toString().trim());
                count_rating++;
            }
            if (count_rating >= 5){
                temp_avg = (count_rating == 0) ? 0.0 : sum_rating / count_rating;
                if (temp_avg > max_rating){
                    max_rating_movie = key.toString();
                    max_rating = temp_avg;
                }
            }
            double avg = (count_rating == 0) ? 0.0 : sum_rating / count_rating;

            context.write(key, new Text(String.format("AverageRating: %.2f (TotalRatings: %d)",
            <-- avg, count_rating)));
        }

        public void cleanup(Context context) throws IOException, InterruptedException {
            context.write(new Text("Highest Rated Movie"), new Text(String.format("%s is the
            <-- highest rated movie with an average rating of %.2f among movies with at least 5
            <-- ratings.", max_rating_movie, max_rating)));
        }
    }

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = new Job(conf, "Bai 1");
        job.setJarByClass(bai1.class);
        job.setMapperClass(RatingMapper.class);
    }
}

```

```

job.setReducerClass(RatingReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
// Setting reducer to zero
//job.setNumReduceTasks(0);

try {

    job.addCacheFile(new URI("hdfs://localhost:8020/mycache/Lab1/movies"));

}
catch (Exception e) {
    System.out.println("File Not Added");
    System.exit(1);
}

FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

## 2. Bài 2

### 2.1. Minh chứng

```

[cloudera@quickstart ~]$ hadoop jar Lab1.jar lab1
15/03/11 22:38:10 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
15/03/11 22:38:17 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
15/03/11 22:38:17 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
    at java.lang.Object.wait(Native Method)
    at java.lang.Thread.join(Thread.java:1281)
    at java.lang.Thread.join(Thread.java:1257)
    at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DFSOutputStream.java:967)
    at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutputStream.java:705)
    at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.close(DFSOutputStream.java:694)
15/03/11 22:38:17 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
    at java.lang.Object.wait(Native Method)
    at java.lang.Thread.join(Thread.java:1281)
    at java.lang.Thread.join(Thread.java:1257)
    at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DFSOutputStream.java:967)
    at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutputStream.java:705)
    at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.close(DFSOutputStream.java:694)
15/03/11 22:38:17 INFO mapreduce.JobSubmitter: number of splits:2
15/03/11 22:38:17 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1741752805670_0002
15/03/11 22:38:17 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8888/proxy/application_1741752805670_0002/
15/03/11 22:38:20 INFO mapreduce.Job: Running job: job_1741752805670_0002 running in uber mode : false
15/03/11 22:38:25 INFO mapreduce.Job: map 0% reduce 0%
15/03/11 22:38:30 INFO mapreduce.Job: map 100% reduce 0%
15/03/11 22:38:39 INFO mapreduce.Job: Job job_1741752805670_0002 complete successfully
15/03/11 22:38:39 INFO mapreduce.Job: Counters
  File System Counters
    FILE: Number of bytes read=3446
    FILE: Number of bytes written=43910
    FILE: Number of read operations=0
    FILE: Number of write operations=0
    DFS: Number of bytes read=2958
    DFS: Number of bytes written=0
    DFS: Number of read operations=9
    DFS: Number of large read operations=0
    DFS: Number of write operations=2
  Job Counters
    Launched map tasks=2
    Launched reduce tasks=1
    Data-local map tasks=2
    Total time spent by all maps in occupied slots (ms)=1335
    Total time spent by all reduces in occupied slots (ms)=3536
    Total time spent by all map tasks (ms)=1335
    Total time spent by all reduce tasks (ms)=3536
    Total vcore-milliseconds taken by all map tasks=1335
    Total vcore-milliseconds taken by all reduce tasks=3536
    Total framework-milliseconds taken by all map tasks=1855940
    Total framework-milliseconds taken by all reduce tasks=3620864
  Map-Reduce Framework
    Map input records=180
    Map output records=256
    Map output bytes=262
    Map output materialized bytes=3452
    Input split bytes=262
    Combine output records=0
    Reduce input groups=17
    Reduce input records=0
    Reduce output bytes=0

```

The terminal window displays the following output:

```

File Edit View Terminal Help
HDFS: Number of write operations=2
Job Counters
  Launched map tasks=2
  Data-local map tasks=1
  Total time spent by all maps in occupied slots (ms)=13235
  Total time spent by all reduces in occupied slots (ms)=3536
  Total time spent by all map tasks (ms)=13591
  Total time spent by all reduce tasks (ms)=3536
  Total vcore-milliseconds taken by all map tasks=1335
  Total megabyte-milliseconds taken by all map tasks=13590
  Total megabyte-milliseconds taken by all map tasks=1355048
  Total megabyte-milliseconds taken by all reduce tasks=3620864
Map-Reduce Framework
  Map input records=198
  Map output records=256
  Map output bytes=256
  Map output materialized bytes=3452
  Input split bytes=256
  Combined input records=0
  Reduce input records=0
  Reduce shuffle bytes=452
  Reduce input records=256
  Redundant map tasks=0
  Spilled Records=312
  Shuffled Maps =2
  Failed map tasks=0
  Merged Map outputs=0
  GC time elapsed (ms)=152
  CPU time spent (ms)=1478
  Physical memory (bytes) snapshot=658141184
  Virtual memory (bytes) snapshot=451945272
  Total committed heap usage (bytes)=584291632
Shuffle Errors
  CONNECTION=0
  IO ERROR=0
  WRONG LENGTH=0
  WRONG MAP=0
  WRONG REDUCE=0
File Input Format Counters
  Bytes Read=2096
  File Output Counters
  Bytes Written=479

```

Below the terminal window, the Java application's code is visible:

```

java -jar Lab1.jar

```

## 2.2. Code

```

import java.io.*;
import java.util.*;
import java.net.URI;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

public class bai2 {

    public static class RatingMapper extends Mapper <Object, Text, Text, Text>
    {
        private Map<Integer, String> movieMap = new HashMap<>();

        public void setup(Context context) throws IOException, InterruptedException
        {
            try (BufferedReader br = new BufferedReader(new FileReader("movies"))){
                String line;
                while ((line = br.readLine()) != null){
                    String columns[] = line.split(",");
                    String id = columns[0].trim();
                    String genres = columns[2].trim();
                    movieMap.put(Integer.parseInt(id), genres);
                }
            }
            catch (IOException e){
                e.printStackTrace();
            }
        }

        public void map(Object key, Text value, Context context) throws IOException, InterruptedException
        {
            String record = value.toString().trim();
            String parts[] = record.split(",");

            String id_movie = parts[1].trim();
            String rating = parts[2].trim();
        }
    }
}

```

```

String genres = movieMap.get(Integer.parseInt(id_movie));

for (String genre : genres.split("\\|")) {
    context.write(new Text(genre.trim()), new Text(rating));
}
}

public static class RatingReducer extends Reducer <Text, Text, Text, Text>
{
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
    ↪ InterruptedException{
        double sum_rating = 0.0;
        int count_rating = 0;
        for (Text val : values) {
            sum_rating += Double.parseDouble(val.toString().trim());
            count_rating++;
        }

        double avg = (count_rating == 0) ? 0.0 : sum_rating / count_rating;

        context.write(key, new Text(String.format("Avg: %.2f ,Count: %d", avg,
        ↪ count_rating)));
    }
}

public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
Job job = new Job(conf, "Bai 2");
job.setJarByClass(bai2.class);
job.setMapperClass(RatingMapper.class);
job.setReducerClass(RatingReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
// Setting reducer to zero
//job.setNumReduceTasks(0);

try {

    job.addCacheFile(new URI("hdfs://localhost:8020/mycache/Lab1/movies"));
}
catch (Exception e) {
    System.out.println("File Not Added");
    System.exit(1);
}

FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

### 3. Bài 3

### 3.1. Minh chứng

```
[Applications Plat... File Edit View VM Jobs Help || | Home x 2259467 x 11,23:00 cloudera

[clouderajpg@ckstart ~]~ hadoop jar Lab1.jar balsi Lab1/input/balsi/part*
23/03/11 22:59:19 INFO Client: RMProxy: Connecting to ResourceManager at 0.0.0.0:8832
23/03/11 22:59:20 WARN mapred.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
23/03/11 22:59:20 INFO mapred.JobClient: Input path: /tmp/hadoop-clouderajpg/part*
23/03/11 22:59:20 INFO mapred.JobClient: Number of splits: 2
23/03/11 22:59:20 INFO mapred.JobClient: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1741752805670_0003/
23/03/11 22:59:21 INFO mapred.JobClient: Job: job_1741752805670_0003 running in uber mode : false
23/03/11 22:59:21 INFO mapred.JobClient: User: quickstart
23/03/11 22:59:21 INFO mapred.JobClient: Running job: job_1741752805670_0003
23/03/11 22:59:21 INFO mapred.MapTask: map 0% reduce 0%
23/03/11 22:59:21 INFO mapred.MapTask: map 50% reduce 0%
23/03/11 22:59:21 INFO mapred.MapTask: map 100% reduce 0%
23/03/11 22:59:28 INFO mapred.MapTask: map 100% reduce 100%
23/03/11 22:59:45 INFO mapred.MapTask: Job job_1741752805670_0003 completed successfully
23/03/11 22:59:45 INFO mapred.Task: Task: attempt_1741752805670_0003_m_000000 finished in 49 sec
File System Counters
FILE: Number of bytes read=0
FILE: Number of bytes written=39316
FILE: Number of read operations=0
FILE: Number of write operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=2958
HDFS: Number of bytes written=39316
HDFS: Number of read operations=9
HDFS: Number of large read operations=0
HDFS: Number of read operations=2
Job Counters
Launched map tasks=2
Launched reduce tasks=1
Data-local map tasks=2
Total map tasks=2, 2 map tasks in occupied slots (ms)=14298
Total time spent by all reduces in occupied slots (ms)=13882
Total time spent by all map tasks (ms)=14298
Total vcore-milliseconds taken by all map tasks=14298
Total vcore-milliseconds taken by all reduce tasks=3882
Total vcore-milliseconds taken by all map tasks=14298
Total mapreduce-milliseconds taken by all map tasks=3975168
Map-Reduce Framework
Map input records=180
Map output records=180
Map input bytes=180
Map output bytes=180
Map output totalized bytes=3884
Input split bytes=262
Combine output records=0
Reduce input groups=58
Reduce input bytes=262
Reduce output records=58
Reduce output bytes=108
Reduce output records=58
SPLIT_RAW_BYTES=108
Shuffled Maps =2
Fallible Map =0
Merged Map outputs=2
GC Time elapsed (ms)=418
CPU Time spent (ms)=14298
Physical memory (bytes) snapshot=64568288
Virtual memory (bytes) snapshot=4519423192
Total committed heap usage (bytes)=50281632
Shuffle Errors
BAD_ID=0
CONNECTION=0
File Input Format Counters
Bytes Read=2656
File Input Format Counters
Bytes Written=2368
File Input Format Counters
Bytes Written=2368
[clouderajpg@ckstart ~]~ hadoop fs -cat Lab1/output/balsi/part*
American Beauty (1999) Male: 3.00 ,Female: 4.00
Avatar (2009) Male: 5.00 ,Female: 4.50
Back to the Future (1985) Male: 3.00 ,Female: 4.00
Birdman (2014) Male: 3.00 ,Female: 3.00
Braveheart (1995) Male: 4.50 ,Female: 4.00
Cloud Atlas (2012) Male: 5.00 ,Female: 4.50
Junkirk (2017) Male: 4.00 ,Female: 3.00
Light Club (1990) Male: 5.00 ,Female: 3.50
Terminator 2 (1991) Male: 5.00 ,Female: 5.00
Gladiator (2000) Male: 3.50 ,Female: 3.00
Good Will Hunting (1997) Male: 4.00 ,Female: 4.00
Gravity (2013) Male: 4.00 ,Female: 4.50
Inception (2010) Male: 3.50 ,Female: 4.00
Interstellar (2014) Male: 4.50 ,Female: 3.50
Jurassic Park (1993) Male: 3.50 ,Female: 4.50
a La Land (2016) Male: 5.00 ,Female: 3.00
Mad Max: Fury Road (2015) Male: 4.00 ,Female: 4.50
Manchester by the Sea (2016) Male: 3.50 ,Female: 4.50
Memento (2000) Male: 3.50 ,Female: 3.50
Moon (2009) Male: 3.00 ,Female: 3.00
Parasite (2019) Male: 4.00 ,Female: 4.00
The Milk Fiction (2004) Male: 4.50 ,Female: 3.50
Say Anything... (1989) Male: 4.00 ,Female: 4.00
Schindler's List (1993) Male: 4.00 ,Female: 4.50
The Dark Knight (2008) Male: 4.50 ,Female: 4.00
The Iluminod Dog Millionaire (2008) Male: 4.50 ,Female: 4.00
Spotlight (2015) Male: 4.50 ,Female: 4.00
Star Wars: Episode Hope (2019) Male: 4.00 ,Female: 3.00
The Avengers (2012) Male: 3.50 ,Female: 4.00
The Big Short (2015) Male: 5.00 ,Female: 4.00
The Godfather (1972) Male: 4.00 ,Female: 4.00
The Godfather (1990) Male: 4.00 ,Female: 4.00
The Green Mile (1999) Male: 3.50 ,Female: 4.00
The Imitation Game (2014) Male: 4.50 ,Female: 3.00
The King's Speech (2010) Male: 4.50 ,Female: 3.50
The Lord of the Rings: The Fellowship of the Ring (2001) Male: 4.50 ,Female: 4.00
The Lord of the Rings: The Return of the King (2003) Male: 4.00 ,Female: 3.00
The Lord of the Rings: The Two Towers (2002) Male: 4.50 ,Female: 3.00
The Matrix (1999) Male: 3.00 ,Female: 4.50
The Prestige (2006) Male: 4.00 ,Female: 4.00
The Revenant (2015) Male: 4.50 ,Female: 3.50
The Shawshank Redemption (1994) Male: 3.00 ,Female: 5.00
The Silence of the Lambs (1991) Male: 3.50 ,Female: 3.50
The Social Network (2010) Male: 4.00 ,Female: 4.00
The Usual Suspects (1995) Male: 4.00 ,Female: 4.50
The War of the Worlds (2005) Male: 3.00 ,Female: 4.50
Titanic (1997) Male: 4.00 ,Female: 3.00
Toy Story (1995) Male: 4.50 ,Female: 4.00
True Grit (2010) Male: 4.00 ,Female: 3.00
[clouderajpg@ckstart ~]~
```

### 3.2. Code

```
import java.io.*;
import java.util.*;
import java.net.URI;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.fs.Path;
import java.net.URI;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
```

```

public class bai3 {

    public static class RatingMapper extends Mapper <Object, Text, Text, Text>
    {
        private Map<Integer, String> movieMap = new HashMap<>();
        private Map<Integer, String> userMap = new HashMap<>();

        public void setup(Context context) throws IOException, InterruptedException
        {
            URI[] CacheFiles = context.getCacheFiles();
            System.out.println("Number of cache files: " + (CacheFiles == null ? 0 :
                CacheFiles.length));

            if (CacheFiles != null && CacheFiles.length > 0){
                try (BufferedReader br = new BufferedReader(new FileReader("movies"))){
                    String line;
                    while ((line = br.readLine()) != null){
                        String columns[] = line.split(",");
                        String movie_id = columns[0].trim();
                        String movie_name = columns[1].trim();
                        movieMap.put(Integer.parseInt(movie_id), movie_name);
                    }
                }
                catch (IOException e){
                    e.printStackTrace();
                }
            }

            if (CacheFiles != null && CacheFiles.length > 1){
                try (BufferedReader br = new BufferedReader(new FileReader("users"))){
                    String line;
                    while ((line = br.readLine()) != null){
                        String columns[] = line.split(",");
                        String user_id = columns[0].trim();
                        String user_gender = columns[1].trim();
                        userMap.put(Integer.parseInt(user_id), user_gender);
                    }
                }
                catch (IOException e){
                    e.printStackTrace();
                }
            }
        }

        public void map(Object key, Text value, Context context) throws IOException, InterruptedException
        {
            String record = value.toString().trim();
            String parts[] = record.split(",");
            String user_id = parts[0].trim();
            String movie_id = parts[1].trim();
            String rating = parts[2].trim();

            String movie_name = movieMap.get(Integer.parseInt(movie_id));
            String user_gender = userMap.get(Integer.parseInt(user_id));

            context.write(new Text(movie_name), new Text(user_gender + " " + rating));
        }
    }

    public static class RatingReducer extends Reducer <Text, Text, Text, Text>
    {
        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
            InterruptedException{
            double sum_male_rating = 0.0;
            double sum_female_rating = 0.0;
            int count_male_rating = 0;
            int count_female_rating = 0;
            for (Text record : values) {
                String parts[] = record.toString().trim().split(" ");
                String user_gender = parts[0];
                if (user_gender.equals("F")){
                    sum_female_rating += Double.parseDouble(parts[1]);
                    count_female_rating += 1;
                }
            }
        }
    }
}

```

```

        else if (user_gender.equals("M")){
            sum_male_rating += Double.parseDouble(parts[1]);
            count_male_rating += 1;
        }
    }

    double female_avg = (count_female_rating == 0) ? 0.0 : sum_female_rating /
    ↪ count_female_rating;
    double male_avg = (count_male_rating == 0) ? 0.0 : sum_male_rating /
    ↪ count_male_rating;
    context.write(key, new Text(String.format("Male: %.2f ,Female: %.2f", male_avg,
    ↪ female_avg)));
}

}

public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
Job job = new Job(conf, "Bai 3");
job.setJarByClass(bai3.class);
job.setMapperClass(RatingMapper.class);
job.setReducerClass(RatingReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
// Setting reducer to zero
//job.setNumReduceTasks(0);

try {

    job.addCacheFile(new URI("hdfs://localhost:8020/mycache/Lab1/movies"));
    job.addCacheFile(new URI("hdfs://localhost:8020/mycache/Lab1/users"));
}
catch (Exception e) {
    System.out.println("File Not Added");
    System.exit(1);
}

FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

## 4. Bài 4

### 4.1. Minh chứng

```

[cloudera@quickstart ~]$ hadoop jar bai3.jar Bai3 /input/Lab1/input /output/bai4
15/03/11 23:08:01 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8082
15/03/11 23:08:01 INFO input.FileInputFormat: Total input paths to process : 2
15/03/11 23:08:01 INFO mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
15/03/11 23:08:01 INFO mapreduce.JobSubmitter: Submits tokens for job: job_1741752805670_0004
15/03/11 23:08:02 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1741752805670_0004
15/03/11 23:08:02 INFO yarn.Client: Submitted application application_1741752805670_0004
15/03/11 23:08:02 INFO mapreduce.Job: Job job_1741752805670_0004 running in uber mode : false
15/03/11 23:08:02 INFO mapreduce.Job:  map 0% reduce 0%
15/03/11 23:08:17 INFO mapreduce.Job:  map 50% reduce 0%
15/03/11 23:08:20 INFO mapreduce.Job:  map 100% reduce 0%
15/03/11 23:08:23 INFO mapreduce.Job:  Job job_1741752805670_0004 completed successfully
15/03/11 23:08:23 INFO mapreduce.Job: Job job_1741752805670_0004 completed successfully
File System Counters
FILE: Number of bytes read=319
FILE: Number of bytes written=439516
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=2958
HDFS: Number of read operations=2
HDFS: Number of read operations=9
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=2
Total time spent by all maps in occupied slots (ms)=12831
Total time spent by all reduces in occupied slots (ms)=3495
Total time spent by all map tasks (ms)=12831
Total time spent by all reduce tasks (ms)=3495
Total wcr=milliseconds taken by all map tasks=12831
Total wcr=milliseconds taken by all reduce tasks=3495
Total shuffle=milliseconds taken by all map tasks=1130944
Total megabyte=milliseconds taken by all reduce tasks=3578800
Map-Reduce Framework
Map input records=100
Map output records=100
Map output bytes=1024
Map spilt bytes=1024
Input split bytes=262
Combine input records=0
Combine output records=0
Reduce input groups=50
Reduce input bytes=13104
Reduce input records=100
Reduce output records=50
Reduce output bytes=900
Shuffled Maps =2
Failed shuffle=0
Merge sort output=2
GC time elapsed (ms)=1338
CPU time spent (ms)=1338
Physical memory (bytes) snapshot=656547840
Virtual memory (bytes) snapshot=4519432192
Total committed heap usage (bytes)=564261632
Shuffle Errors
BAD ID=0
CONNECTION=0

```

```

Total committed heap usage (bytes)=584261632
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File InputFormat counters
  Bytes Read=2696
  File Output Format Counters
  Bytes Written=292
[cloudera@quickstart:~]$ hdfs dfs -cat hdfs://quickstart:9000/user/root/movies/part-r-00000
American Beauty (1999) 0-18: NA 18-35: 4.50 35-50: 5.00 50+: NA
Avatar (2009) 0-18: NA 18-35: 4.50 35-50: 5.00 50+: NA
Back to the Future (1985) 0-18: NA 18-35: 4.00 35-50: NA 50+: NA
Braveheart (1995) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Coco (2017) 0-18: NA 18-35: 4.25 35-50: NA 50+: NA
Dunkirk (2017) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Fight Club (1999) 0-18: NA 18-35: 4.25 35-50: NA 50+: NA
Forrest Gump (1994) 0-18: NA 18-35: 3.75 35-50: 3.50 50+: NA
Gladiator (2000) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Good Will Hunting (1997) 0-18: NA 18-35: 3.50 35-50: NA 50+: NA
Gravity (2013) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Inception (2010) 0-18: NA 18-35: 3.75 35-50: NA 50+: NA
Interstellar (2014) 0-18: NA 18-35: 3.50 35-50: 3.50 50+: NA
Jurassic Park (1993) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
La La Land (2016) 0-18: NA 18-35: 3.80 35-50: 5.00 50+: NA
Mad Max: Fury Road (2015) 0-18: NA 18-35: 4.00 35-50: NA 50+: NA
Memento (2000) 0-18: NA 18-35: 4.50 35-50: 3.50 50+: NA
Moonlight (2016) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Parasite (2019) 0-18: NA 18-35: 4.50 35-50: 4.00 50+: NA
Pulp Fiction (1994) 0-18: NA 18-35: 4.00 35-50: NA 50+: NA
Saving Private Ryan (1998) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Schindler's List (1993) 0-18: NA 18-35: 4.50 35-50: 4.00 50+: NA
Se7en (1995) 0-18: NA 18-35: 3.75 35-50: NA 50+: NA
Snowpiercer (2013) 0-18: NA 18-35: 3.50 35-50: 4.00 50+: NA
Spotlight (2015) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Star Wars: A New Hope (1977) 0-18: NA 18-35: 3.00 35-50: 4.00 50+: NA
The Apartment (1960) 0-18: NA 18-35: 3.50 35-50: 4.00 50+: NA
The Big Short (2015) 0-18: NA 18-35: 4.50 35-50: NA 50+: NA
The Dark Knight (2008) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
The Godfather (1972) 0-18: NA 18-35: 3.50 35-50: NA 50+: NA
The Godfather Part II (1974) 0-18: NA 18-35: 3.50 35-50: NA 50+: NA
The Initiation Game (2014) 0-18: NA 18-35: 3.00 35-50: 4.50 50+: NA
The Lion King (1994) 0-18: NA 18-35: 4.00 35-50: NA 50+: NA
The Lord of the Rings: The Fellowship of the Ring (2001) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
The Lord of the Rings: The Return of the King (2003) 0-18: NA 18-35: 3.50 35-50: NA 50+: NA
The Lord of the Rings: The Two Towers (2002) 0-18: NA 18-35: 3.00 35-50: 4.50 50+: NA
The Matrix (1999) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
The Prestige (2006) 0-18: NA 18-35: 4.00 35-50: 4.00 50+: NA
The Silence of the Lambs (1991) 0-18: NA 18-35: 3.50 35-50: 3.50 50+: NA
The Social Network (2010) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
The Usual Suspects (1995) 0-18: NA 18-35: 4.25 35-50: NA 50+: NA
The Wolf of Wall Street (2013) 0-18: NA 18-35: 4.50 35-50: 4.00 50+: NA
Titanic (1997) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Toy Story (1995) 0-18: NA 18-35: 4.00 35-50: 4.50 50+: NA
Whiplash (2014) 0-18: NA 18-35: 3.00 35-50: 4.50 50+: NA
[cloudera@quickstart:~]$
```

## 4.2. Code

```

import java.io.*;
import java.util.*;
import java.net.URI;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.fs.Path;
import java.net.URI;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

public class bai4 {

    public static class RatingMapper extends Mapper <Object, Text, Text, Text>
    {
        private Map<Integer, String> movieMap = new HashMap<>();
        private Map<Integer, String> userMap = new HashMap<>();

        public void setup(Context context) throws IOException, InterruptedException
        {
            URI[] CacheFiles = context.getCacheFiles();
            System.out.println("Number of cache files: " + (CacheFiles == null ? 0 : CacheFiles.length));

            if (CacheFiles != null && CacheFiles.length > 0){
                try (BufferedReader br = new BufferedReader(new FileReader("movies"))){
                    String line;
                    while ((line = br.readLine()) != null){
                        String columns[] = line.split(",");
                        String movie_id = columns[0].trim();
                        String movie_name = columns[1].trim();
                        movieMap.put(Integer.parseInt(movie_id), movie_name);
                    }
                }
                catch (IOException e){
                    e.printStackTrace();
                }
            }
        }
    }
}
```

```

        if (CacheFiles != null && CacheFiles.length > 1){
            try (BufferedReader br = new BufferedReader(new FileReader("users"))){
                String line;
                while ((line = br.readLine()) != null){
                    String columns[] = line.split(",");
                    String user_id = columns[0].trim();
                    String user_age = columns[2].trim();
                    userMap.put(Integer.parseInt(user_id), user_age);
                }
            } catch (IOException e){
                e.printStackTrace();
            }
        }
    }

    public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
        String record = value.toString().trim();
        String parts[] = record.split(",");
        String user_id = parts[0].trim();
        String movie_id = parts[1].trim();
        String rating = parts[2].trim();

        String movie_name = movieMap.get(Integer.parseInt(movie_id));
        String user_age = userMap.get(Integer.parseInt(user_id));

        context.write(new Text(movie_name), new Text(user_age + " " + rating));
    }
}

public static class RatingReducer extends Reducer<Text, Text, Text, Text> {
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
    InterruptedException {
        double sum_0_18 = 0.0, sum_18_35 = 0.0, sum_35_50 = 0.0, sum_50_plus = 0.0;
        int count_0_18 = 0, count_18_35 = 0, count_35_50 = 0, count_50_plus = 0;

        for (Text record : values) {
            String parts[] = record.toString().trim().split(" ");
            int user_age = Integer.parseInt(parts[0]);
            double rating = Double.parseDouble(parts[1]);

            if (user_age <= 18) {
                sum_0_18 += rating;
                count_0_18++;
            } else if (user_age <= 35) {
                sum_18_35 += rating;
                count_18_35++;
            } else if (user_age <= 50) {
                sum_35_50 += rating;
                count_35_50++;
            } else {
                sum_50_plus += rating;
                count_50_plus++;
            }
        }

        String avg_0_18 = (count_0_18 == 0) ? "NA" : String.format("%.2f", sum_0_18 /
        count_0_18);
        String avg_18_35 = (count_18_35 == 0) ? "NA" : String.format("%.2f", sum_18_35 /
        count_18_35);
        String avg_35_50 = (count_35_50 == 0) ? "NA" : String.format("%.2f", sum_35_50 /
        count_35_50);
        String avg_50_plus = (count_50_plus == 0) ? "NA" : String.format("%.2f", sum_50_plus /
        count_50_plus);

        context.write(key, new Text(String.format("0-18: %s 18-35: %s 35-50: %s 50+: %s",
        avg_0_18, avg_18_35, avg_35_50, avg_50_plus)));
    }
}

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = new Job(conf, "Bai 4");
    job.setJarByClass(bai4.class);
    job.setMapperClass(RatingMapper.class);
}

```

```
job.setReducerClass(RatingReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
// Setting reducer to zero
//job.setNumReduceTasks(0);

try {
    job.addCacheFile(new URI("hdfs://localhost:8020/mycache/Lab1/movies"));
    job.addCacheFile(new URI("hdfs://localhost:8020/mycache/Lab1/users"));
}
catch (Exception e) {
    System.out.println("File Not Added");
    System.exit(1);
}

FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```