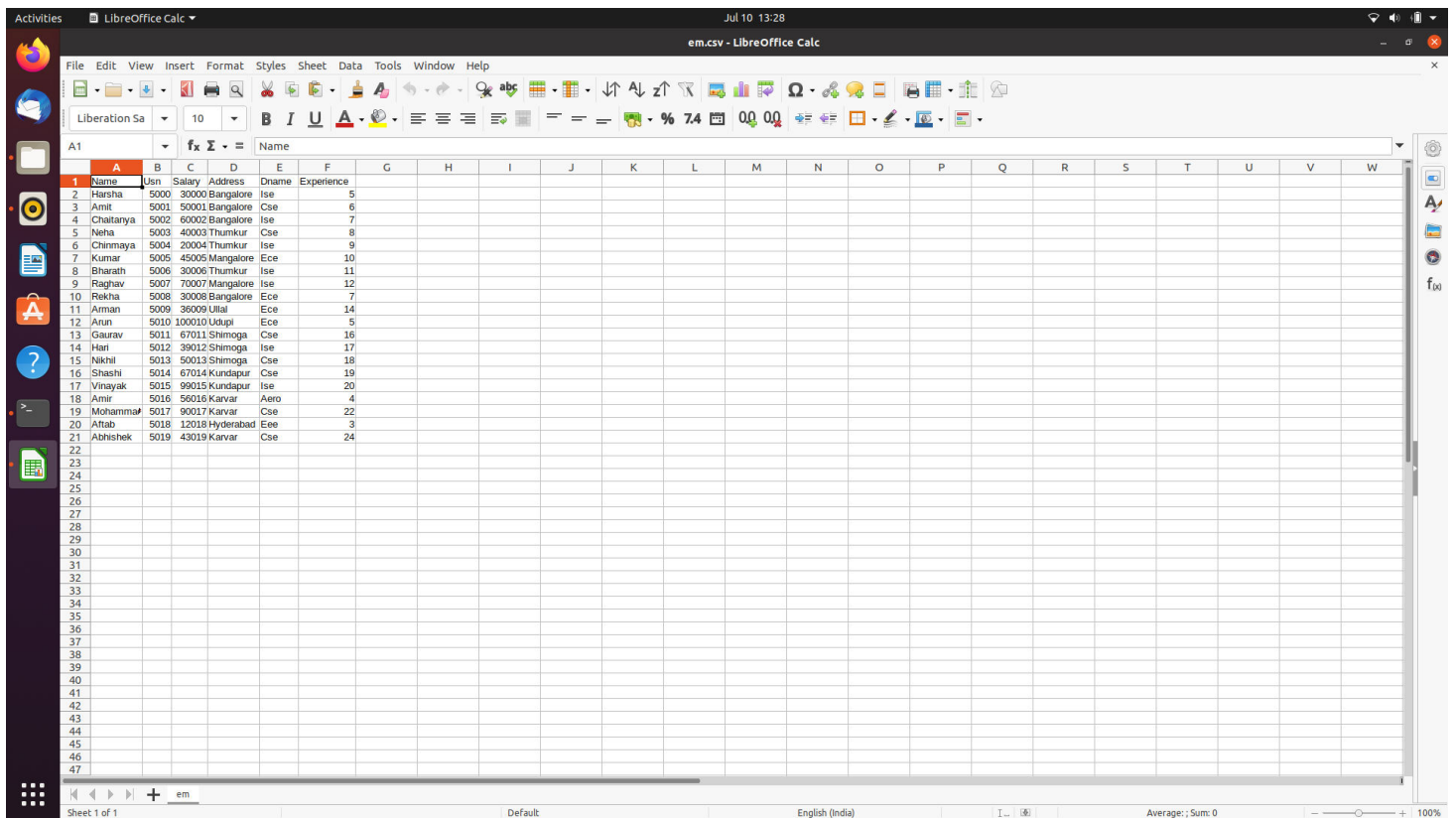# LA2

Chinmayakrishna B
1NT18IS047
Exercise1: Hadoop map reduce
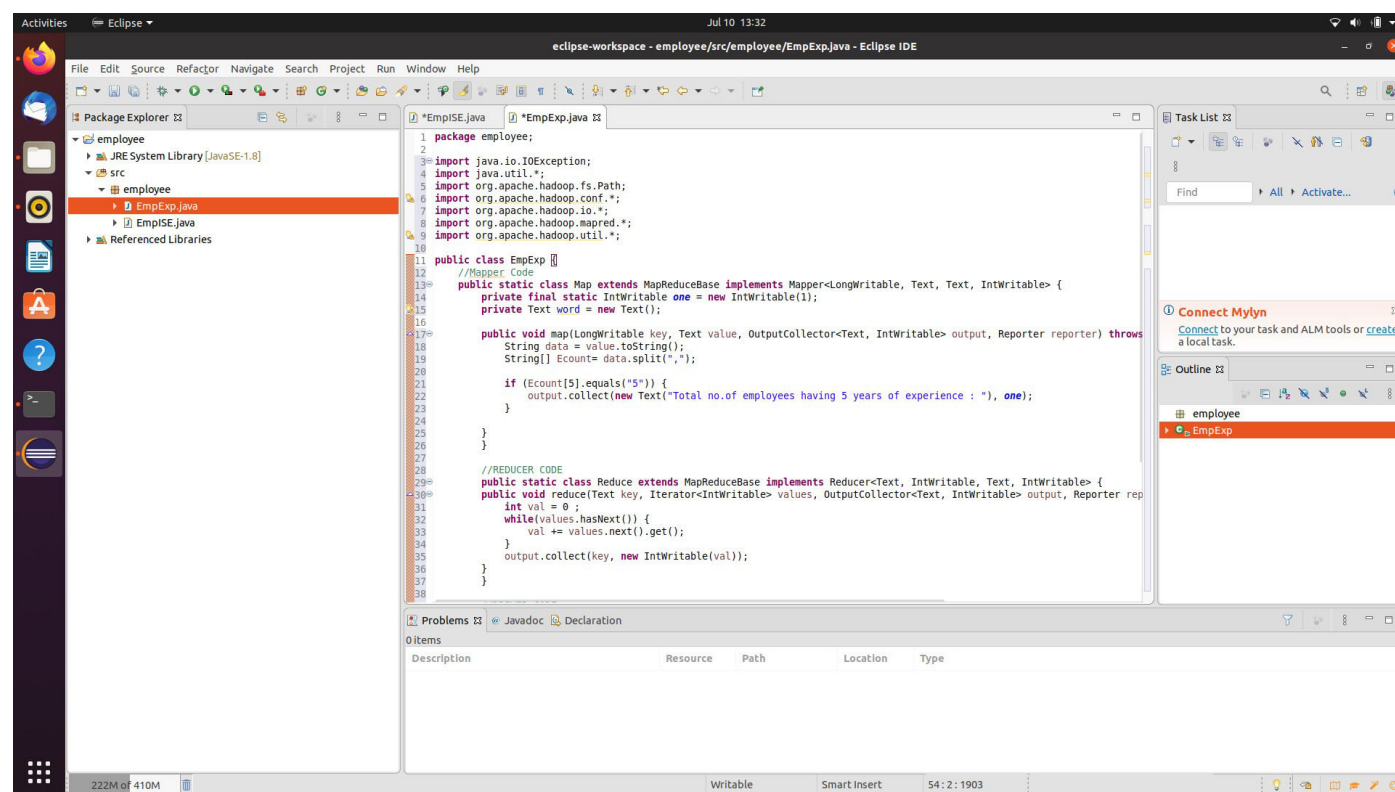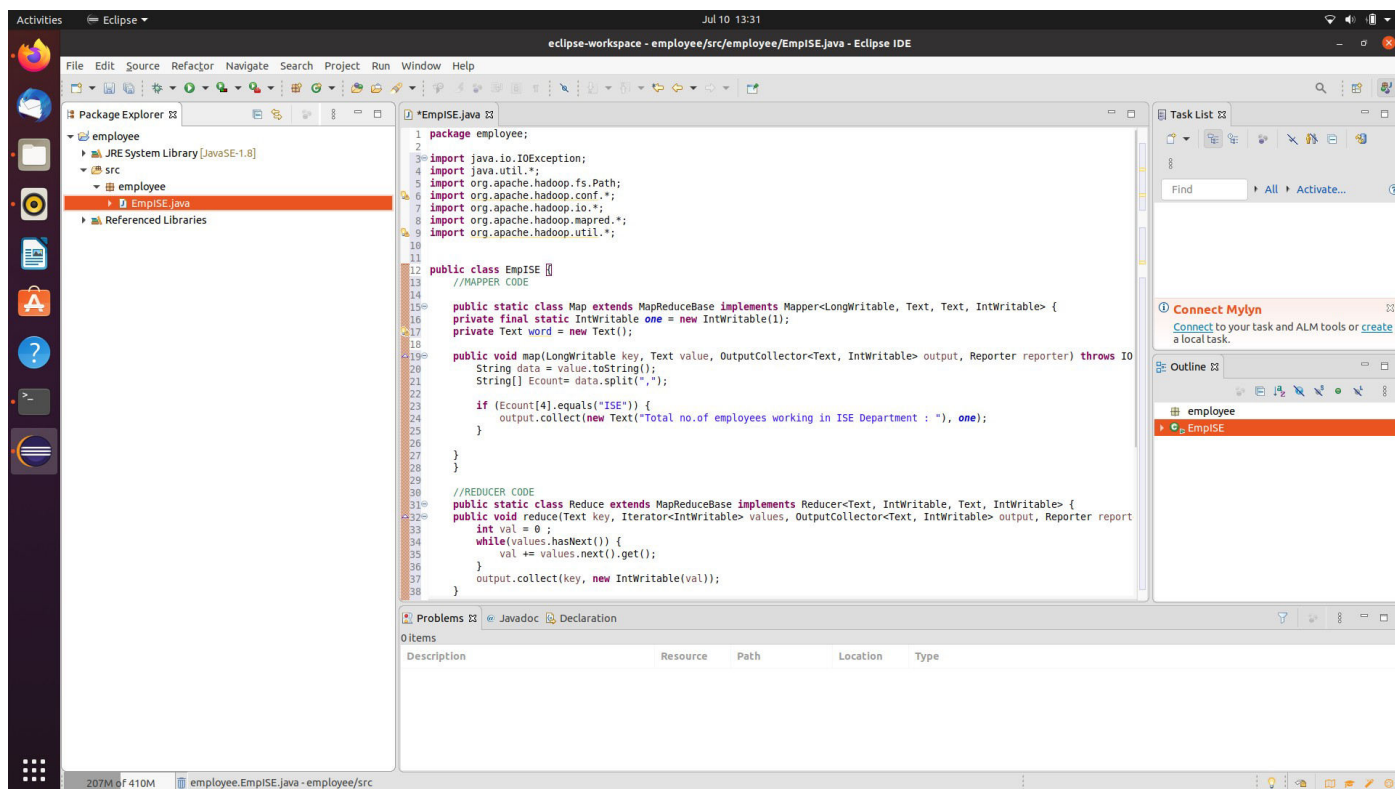
-> CSV file
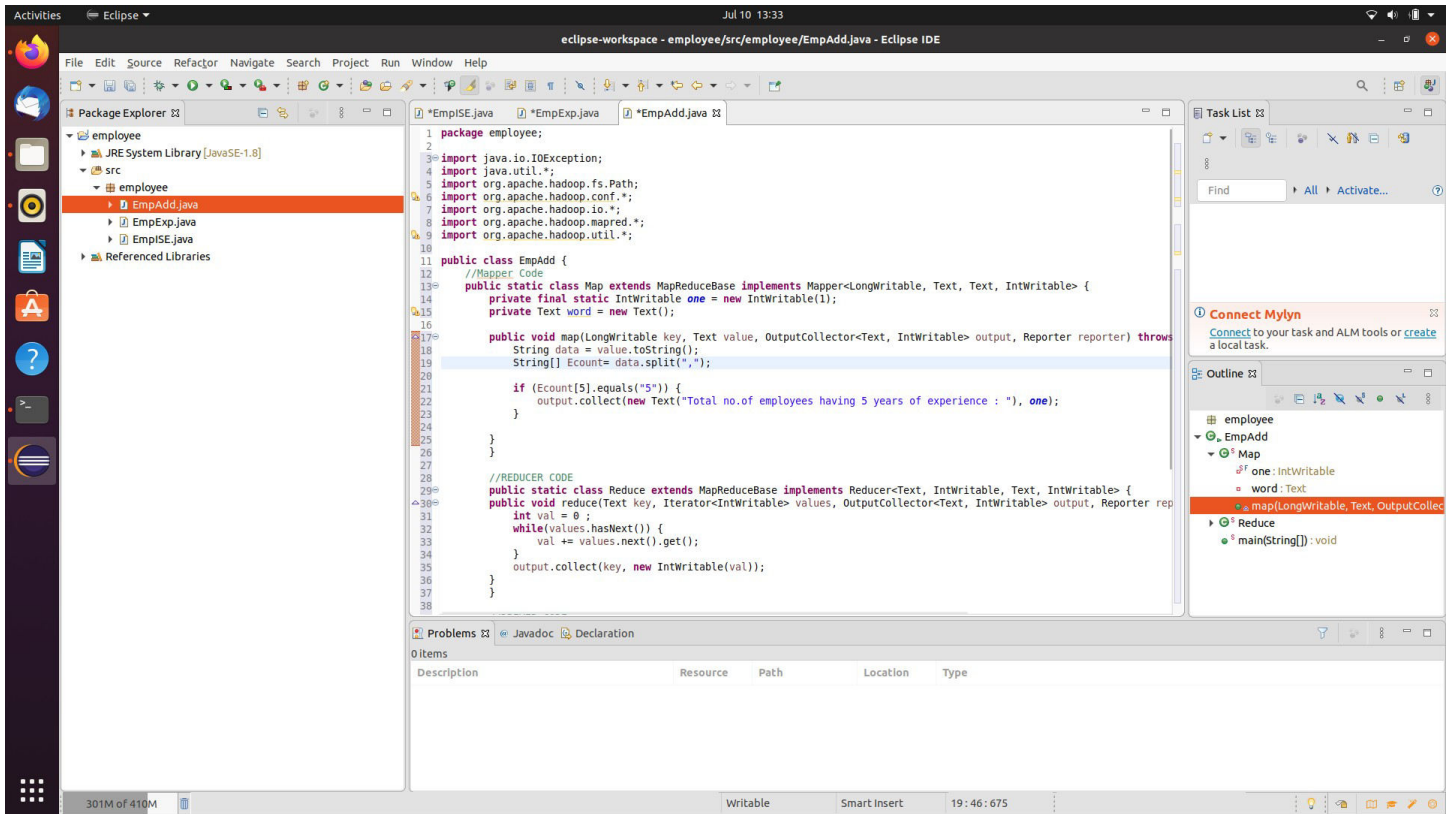

	Name	Usn	Salary	Address	Dname	Experience
1	Name	Usn	Salary	Address	Dname	Experience
2	Harsha	5000	30000	Bangalore	Ise	5
3	Amit	5001	50001	Bangalore	Cse	6
4	Chaitanya	5002	60002	Bangalore	Ise	7
5	Neha	5003	40003	Thumkur	Cse	8
6	Chinmaya	5004	20004	Thumkur	Ise	9
7	Kumar	5005	45005	Mangalore	Ece	10
8	Bharath	5006	30006	Thumkur	Ise	11
9	Raghav	5007	70007	Mangalore	Ise	12
10	Rekha	5008	30008	Bangalore	Ece	7
11	Arman	5009	36009	Ullal	Ece	14
12	Arun	5010	100010	Udupi	Ece	5
13	Gaurav	5011	67011	Shimoga	Cse	16
14	Hari	5012	39012	Shimoga	Ise	17
15	Nikhil	5013	50013	Shimoga	Cse	18
16	Shashi	5014	67014	Kundapur	Cse	19
17	Vinayak	5015	99015	Kundapur	Ise	20
18	Amir	5016	56016	Karvar	Aero	4
19	Mohammad	5017	90017	Karvar	Cse	22
20	Aftab	5018	12018	Hyderabad	Eee	3
21	Abhishek	5019	43019	Karvar	Cse	24

->Eclipse

```java
package employee;

import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;


public class EmpISE {
    //MAPPER CODE

    public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IO
            String data = value.toString();
            String[] Ecount= data.split(",");

            if (Ecount[4].equals("ISE")) {
                output.collect(new Text("Total no.of employees working in ISE Department : "), one);
            }

        }
    }

    //REDUCER CODE
    public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter report
        int val = 0 ;
        while(values.hasNext()) {
            val += values.next().get();
        }
        output.collect(key, new IntWritable(val));
    }
}
```

```java
package employee;

import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class EmpExp {
    //Mapper Code
    public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws
            String data = value.toString();
            String[] Ecount= data.split(",");

            if (Ecount[5].equals("5")) {
                output.collect(new Text("Total no.of employees having 5 years of experience : "), one);
            }

        }
    }

    //REDUCER CODE
    public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter rep
        int val = 0 ;
        while(values.hasNext()) {
            val += values.next().get();
        }
        output.collect(key, new IntWritable(val));
    }
}
```

Task
1)Total number of employees who work in ISE department

2)Total Number of employees with experience=5years

```
                  Total time spent by all map tasks (ms)=3918
                  Total time spent by all reduce tasks (ms)=1705
                  Total vcore-milliseconds taken by all map tasks=3918
                  Total vcore-milliseconds taken by all reduce tasks=1705
                  Total megabyte-milliseconds taken by all map tasks=4012032
                  Total megabyte-milliseconds taken by all reduce tasks=1745920
         Map-Reduce Framework
                  Map input records=21
                  Map output records=8
                  Map output bytes=464
                  Map output materialized bytes=132
                  Input split bytes=184
                  Combine input records=8
                  Combine output records=2
                  Reduce input groups=1
                  Reduce shuffle bytes=132
                  Reduce input records=2
                  Reduce output records=1
                  Spilled Records=4
                  Shuffled Maps =2
                  Failed Shuffles=0
                  Merged Map outputs=2
                  GC time elapsed (ms)=120
                  CPU time spent (ms)=1240
                  Physical memory (bytes) snapshot=904175616
                  Virtual memory (bytes) snapshot=7668494336
                  Total committed heap usage (bytes)=770179072
                  Peak Map Physical memory (bytes)=337760256
                  Peak Map Virtual memory (bytes)=2553155584
                  Peak Reduce Physical memory (bytes)=230051840
                  Peak Reduce Virtual memory (bytes)=2562248704
         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=1020
         File Output Format Counters
                  Bytes Written=56
hdoop@Chinmayakb:~/Documents$ hadoop fs -ls
Found 3 items
drwxr-xr-x   - hdoop supergroup          0 2021-07-10 13:35 EmpExp
drwxr-xr-x   - hdoop supergroup          0 2021-07-10 13:25 EmpISE.txt
-rw-r--r--   1 hdoop supergroup        680 2021-07-10 13:25 emp.csv
hdoop@Chinmayakb:~/Documents$ hadoop fs -ls EmpExp
Found 2 items
-rw-r--r--   1 hdoop supergroup          0 2021-07-10 13:35 EmpExp/_SUCCESS
-rw-r--r--   1 hdoop supergroup         56 2021-07-10 13:35 EmpExp/part-00000
hdoop@Chinmayakb:~/Documents$ hadoop fs -cat EmpExp/part-00000
2021-07-10 13:36:31,453 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
Total no.of employees having 2 years of experience :    8
hdoop@Chinmayakb:~/Documents$
```

3) Count the number of employees who lives in Bangalore


```
         at java.lang.ClassLoader.loadClass(ClassLoader.java:351)
         at java.lang.Class.forName0(Native Method)
         at java.lang.Class.forName(Class.java:348)
         at org.apache.hadoop.util.RunJar.run(RunJar.java:316)
         at org.apache.hadoop.util.RunJar.main(RunJar.java:236)
hdoop@Chinmayakb:~/Documents$ hadoop jar EmpAdd.jar ExpAdd emp.csv EmpAdd
2021-07-10 13:38:32,107 INFO client.RMProxy: Connecting to ResourceManager at /127.0.0.1:8032
2021-07-10 13:38:32,234 INFO client.RMProxy: Connecting to ResourceManager at /127.0.0.1:8032
2021-07-10 13:38:32,360 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2021-07-10 13:38:32,411 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hdoop/.staging/job_1625897919713_0015
2021-07-10 13:38:32,495 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-07-10 13:38:32,592 INFO mapred.FileInputFormat: Total input files to process : 1
2021-07-10 13:38:32,656 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-07-10 13:38:32,768 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-07-10 13:38:32,779 INFO mapreduce.JobSubmitter: number of splits:2
2021-07-10 13:38:32,924 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-07-10 13:38:32,934 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1625897919713_0015
2021-07-10 13:38:32,935 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-07-10 13:38:33,055 INFO conf.Configuration: resource-types.xml not found
2021-07-10 13:38:33,055 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-07-10 13:38:33,097 INFO impl.YarnClientImpl: Submitted application application_1625897919713_0015
2021-07-10 13:38:33,122 INFO mapreduce.Job: The url to track the job: http://Chinmayakb:8088/proxy/application_1625897919713_0015/
2021-07-10 13:38:33,123 INFO mapreduce.Job: Running job: job_1625897919713_0015
2021-07-10 13:38:38,198 INFO mapreduce.Job: Job job_1625897919713_0015 running in uber mode : false
2021-07-10 13:38:38,200 INFO mapreduce.Job:  map 0% reduce 0%
2021-07-10 13:38:42,254 INFO mapreduce.Job:  map 100% reduce 0%
2021-07-10 13:38:46,303 INFO mapreduce.Job:  map 100% reduce 100%
2021-07-10 13:38:47,336 INFO mapreduce.Job: Job job_1625897919713_0015 completed successfully
2021-07-10 13:38:47,425 INFO mapreduce.Job: Counters: 54
         File System Counters
                  FILE: Number of bytes read=58
                  FILE: Number of bytes written=677497
                  FILE: Number of read operations=0
                  FILE: Number of large read operations=0
                  FILE: Number of write operations=0
                  HDFS: Number of bytes read=1204
                  HDFS: Number of bytes written=48
                  HDFS: Number of read operations=11
                  HDFS: Number of large read operations=0
                  HDFS: Number of write operations=2
                  HDFS: Number of bytes read erasure-coded=0
         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)=4046
                  Total time spent by all reduces in occupied slots (ms)=1686
                  Total time spent by all map tasks (ms)=4046
                  Total time spent by all reduce tasks (ms)=1686
                  Total vcore-milliseconds taken by all map tasks=4046
                  Total vcore-milliseconds taken by all reduce tasks=1686
                  Total megabyte-milliseconds taken by all map tasks=4143104
                  Total megabyte-milliseconds taken by all reduce tasks=1726464
         Map-Reduce Framework
                  Map input records=21
                  Map output records=1
```

Task1:Code:

```
package employee;
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;


public class EmpISE {
        //MAPPER CODE

        public static class Map extends MapReduceBase implements Mapper<LongWritable, Text,
Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
Reporter reporter) throws IOException {
                String data = value.toString();
                String[] Ecount= data.split(",");

                if (Ecount[4].equals("ISE")) {
                        output.collect(new Text("Total no.of employees working in ISE Department :
"), one);
```

```
            }

        }
    }

    //REDUCER CODE
    public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable,
Text, IntWritable> {
        public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text,
IntWritable> output, Reporter reporter) throws IOException { //{little: {1,1}}
            int val = 0 ;
            while(values.hasNext()) {
                val += values.next().get();
            }
            output.collect(key, new IntWritable(val));
        }
    }

    //DRIVER CODE
    public static void main(String[] args) throws Exception {
        JobConf conf = new JobConf(EmpISE.class);
        conf.setJobName("Total no.of employees working in ISE Department");
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(Map.class);
        conf.setCombinerClass(Reduce.class);
        conf.setReducerClass(Reduce.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

Task2:code:

```
package employee;

import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class EmpExp {
    //Mapper Code
```

```java
        public static class Map extends MapReduceBase implements Mapper<LongWritable,
Text, Text, IntWritable> {
                private final static IntWritable one = new IntWritable(1);
                private Text word = new Text();

                public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output, Reporter reporter) throws IOException {
                        String data = value.toString();
                        String[] Ecount= data.split(",");

                        if (Ecount[5].equals("5")) {
                                output.collect(new Text("Total no.of employees having 5 years
of experience : "), one);
                        }

                }
                }

                //REDUCER CODE
                public static class Reduce extends MapReduceBase implements
Reducer<Text, IntWritable, Text, IntWritable> {
                public void reduce(Text key, Iterator<IntWritable> values,
OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
//{little: {1,1}}
                        int val = 0 ;
                        while(values.hasNext()) {
                                val += values.next().get();
                        }
                        output.collect(key, new IntWritable(val));
                }
                }

                //DRIVER CODE
                public static void main(String[] args) throws Exception {
                        JobConf conf = new JobConf(EmpExp.class);
                        conf.setJobName("Total no.of employees having 5 years of
experience");
                        conf.setOutputKeyClass(Text.class);
                        conf.setOutputValueClass(IntWritable.class);
                        conf.setMapperClass(Map.class);
                        conf.setCombinerClass(Reduce.class);
                        conf.setReducerClass(Reduce.class);
                        conf.setInputFormat(TextInputFormat.class);
                        conf.setOutputFormat(TextOutputFormat.class);
                        FileInputFormat.setInputPaths(conf, new Path(args[0]));
                        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
                        JobClient.runJob(conf);
                }
```

}

Task3:Code:

```java
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class ExpAdd {
        //Mapper Code
        public static class Map extends MapReduceBase implements Mapper<LongWritable,
Text, Text, IntWritable> {
                private final static IntWritable one = new IntWritable(1);
                private Text word = new Text();

                public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output, Reporter reporter) throws IOException {
                        String data = value.toString();
                        String[] Ecount= data.split(",");

                        if (Ecount[3].equals("Bengaluru")) {
                                output.collect(new Text("Total no.of employees working in
Bengaluru : "), one);
                        }

                }
                }

                //REDUCER CODE
                public static class Reduce extends MapReduceBase implements
Reducer<Text, IntWritable, Text, IntWritable> {
                public void reduce(Text key, Iterator<IntWritable> values,
OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
//{little: {1,1}}
                        int val = 0 ;
                        while(values.hasNext()) {
                                val += values.next().get();
                        }
                        output.collect(key, new IntWritable(val));
                }
                }

                //DRIVER CODE
```

```java
public static void main(String[] args) throws Exception {
    JobConf conf = new JobConf(EmpExp.class);
    conf.setJobName("Total no.of employees working in Bengaluru");
    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);
    conf.setMapperClass(Map.class);
    conf.setCombinerClass(Reduce.class);
    conf.setReducerClass(Reduce.class);
    conf.setInputFormat(TextInputFormat.class);
    conf.setOutputFormat(TextOutputFormat.class);
    FileInputFormat.setInputPaths(conf, new Path(args[0]));
    FileOutputFormat.setOutputPath(conf, new Path(args[1]));
    JobClient.runJob(conf);
}
}
```