MapReduce 应用案例

该系列课程是应邀实验楼整理编写的，这里需要赞一下实验楼提供了学习的新方式，可以边看博客边上机实验，课程地址为 https://www.shiyanlou.com/courses/237

【注】该系列所使用到安装包、测试数据和代码均可在百度网盘下载，具体地址为 http://pan.baidu.com/s/10PnDs，下载该 PDF 文件

1 环境说明

部署节点操作系统为 CentOS，防火墙和 SElinux 禁用，创建了一个 shiyanlou 用户并在系统根目录下创建/app 目录，用于存放 Hadoop 等组件运行包。因为该目录用于安装 hadoop 等组件程序，用户对 shiyanlou 必须赋予 rwx 权限（一般做法是 root 用户在根目录下创建/app 目录，并修改该目录拥有者为 shiyanlou(chown –R shiyanlou:shiyanlou /app)。

Hadoop 搭建环境：

- 虚拟机操作系统： CentOS6.6 64 位，单核，1G 内存
- JDK：1.7.0_55 64 位
- Hadoop：1.1.2

2 准备测试数据

测试数据包括两个文件 dept（部门）和 emp（员工），其中各字段用逗号分隔：

dept 文件内容：

10,ACCOUNTING,NEW YORK
20,RESEARCH,DALLAS
30,SALES,CHICAGO
40,OPERATIONS,BOSTON

emp 文件内容：

7369,SMITH,CLERK,7902,17-12 月-80,800,,20
7499,ALLEN,SALESMAN,7698,20-2 月-81,1600,300,30

7521,WARD,SALESMAN,7698,22-2 月-81,1250,500,30

7566,JONES,MANAGER,7839,02-4 月-81,2975,,20

7654,MARTIN,SALESMAN,7698,28-9 月-81,1250,1400,30

7698,BLAKE,MANAGER,7839,01-5 月-81,2850,,30

7782,CLARK,MANAGER,7839,09-6 月-81,2450,,10

7839,KING,PRESIDENT,,17-11 月-81,5000,,10

7844,TURNER,SALESMAN,7698,08-9 月-81,1500,0,30

7900,JAMES,CLERK,7698,03-12 月-81,950,,30

7902,FORD,ANALYST,7566,03-12 月-81,3000,,20

7934,MILLER,CLERK,7782,23-1 月-82,1300,,10

在/home/shiyanlou/install-pack/class6 目录可以找到这两个文件，把这两个文件上传到 HDFS 中/class6/input 目录中，执行如下命令：

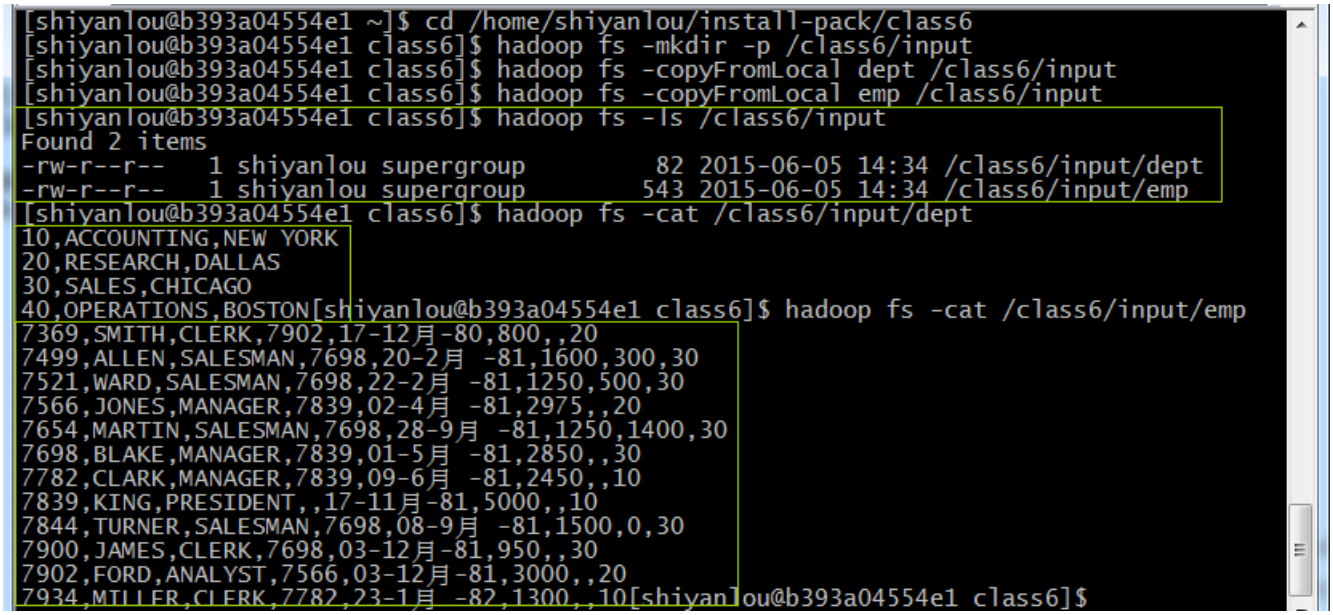
cd /home/shiyanlou/install-pack/class6

hadoop fs -mkdir -p /class6/input

hadoop fs -copyFromLocal dept /class6/input

hadoop fs -copyFromLocal emp /class6/input

hadoop fs -ls /class6/input



3 应用案例

3.1 测试例子 1：求各个部门的总工资

3.1.1 问题分析

MapReduce 中的 join 分为好几种，比如有最常见的 reduce side join、map side join 和 semi join 等。reduce join 在 shuffle 阶段要进行大量的数据传输，会造成大量的网络 IO 效率低下，而 map side join 在处理多个小表关联大表时非常有用 。

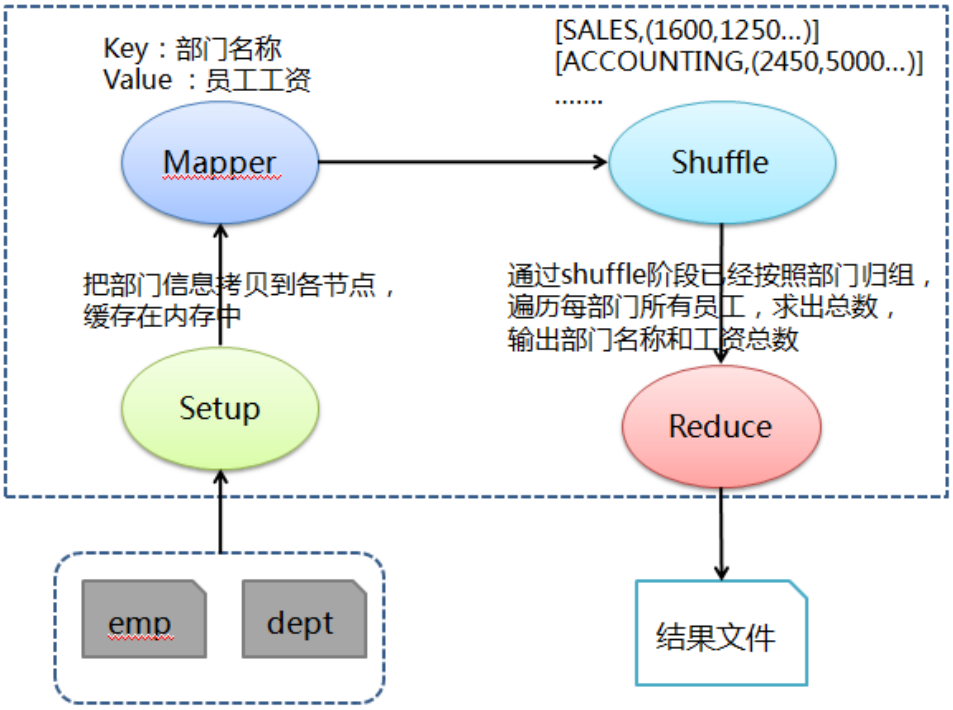
Map side join 是针对以下场景进行的优化：两个待连接表中，有一个表非常大，而另一个表非常小，以至于小表可以直接存放到内存中。这样我们可以将小表复制多份，让每个 map task 内存中存在一份（比如存放到 hash table 中），然后只扫描大表：对于大表中的每一条记录 key/value，在 hash table 中查找是否有相同的 key 的记录，如果有，则连接后输出即可。为了支持文件的复制，Hadoop 提供了一个类 DistributedCache，使用该类的方法如下：

（1）用户使用静态方法 DistributedCache.addCacheFile()指定要复制的文件，它的参数是文件的 URI（如果是 HDFS 上的文件，可以这样：hdfs://jobtracker:50030/home/XXX/file）。JobTracker 在作业启动之前会获取这个 URI 列表，并将相应的文件拷贝到各个 TaskTracker 的本地磁盘上。

（2）用户使用 DistributedCache.getLocalCacheFiles()方法获取文件目录，并使用标准的文件读写 API 读取相应的文件。

在下面代码中，将会把数据量小的表(部门 dept）缓存在内存中，在 Mapper 阶段对员工部门编号映射成部门名称，该名称作为 key 输出到 Reduce 中，在 Reduce 中计算按照部门计算各个部门的总工资。

3.1.2 处理流程图



3.1.3 测试代码

Q1SumDeptSalary.java 代码（vi 编辑代码是不能存在中文）：

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;
```

```
public class Q1SumDeptSalary extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

        // 用于缓存 dept文件中的数据
        private Map<String, String> deptMap = new HashMap<String, String>();
        private String[] kv;

        // 此方法会在Map方法执行之前执行且执行一次
        @Override
        protected void setup(Context context) throws IOException, InterruptedException {
            BufferedReader in = null;
            try {

                // 从当前作业中获取要缓存的文件
                Path[] paths = DistributedCache.getLocalCacheFiles(context.getConfiguration());
                String deptIdName = null;
                for (Path path : paths) {

                    // 对部门文件字段进行拆分并缓存到deptMap中
                    if (path.toString().contains("dept")) {
                        in = new BufferedReader(new FileReader(path.toString()));
                        while (null != (deptIdName = in.readLine())) {

                            // 对部门文件字段进行拆分并缓存到deptMap中
                            // 其中Map中key为部门编号，value为所在部门名称
                            deptMap.put(deptIdName.split(",")[0], deptIdName.split(",")[1]);
                        }
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    if (in != null) {
                        in.close();
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        public void map(LongWritable key, Text value, Context context) throws IOException,
        InterruptedException {

            // 对员工文件字段进行拆分
```

```
            kv = value.toString().split(",");

            // map join: 在map阶段过滤掉不需要的数据，输出key为部门名称和value为员工工资
            if (deptMap.containsKey(kv[7])) {
                if (null != kv[5] && !"".equals(kv[5].toString())) {
                    context.write(new Text(deptMap.get(kv[7].trim())), new
Text(kv[5].trim()));
                }
            }
        }
    }

    public static class Reduce extends Reducer<Text, Text, Text, LongWritable> {

        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
        InterruptedException {

            // 对同一部门的员工工资进行求和
            long sumSalary = 0;
            for (Text val : values) {
                sumSalary += Long.parseLong(val.toString());
            }

            // 输出key为部门名称和value为该部门员工工资总和
            context.write(key, new LongWritable(sumSalary));
        }
    }

    @Override
    public int run(String[] args) throws Exception {

        // 实例化作业对象，设置作业名称、Mapper和Reduce类
        Job job = new Job(getConf(), "Q1SumDeptSalary");
        job.setJobName("Q1SumDeptSalary");
        job.setJarByClass(Q1SumDeptSalary.class);
        job.setMapperClass(MapClass.class);
        job.setReducerClass(Reduce.class);

        // 设置输入格式类
        job.setInputFormatClass(TextInputFormat.class);

        // 设置输出格式
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        // 第1个参数为缓存的部门数据路径、第2个参数为员工数据路径和第3个参数为输出路径
```

```
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        DistributedCache.addCacheFile(new Path(otherArgs[0]).toUri(),
        job.getConfiguration());
         FileInputFormat.addInputPath(job, new Path(otherArgs[1]));
         FileOutputFormat.setOutputPath(job, new Path(otherArgs[2]));

         job.waitForCompletion(true);
         return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q1SumDeptSalary(), args);
        System.exit(res);
    }
}
```

3.1.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q1SumDeptSalary.java 程序代码(代码页可以使用/home/shiyanlou/install-pack/class6/Q1SumDeptSalary.java 文件)

cd /app/hadoop-1.1.2/myclass/class6

vi Q1SumDeptSalary.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q1SumDeptSalary.java

把编译好的代码打成 jar 包（如果不打成 jar 形式运行会提示 class 无法找到的错误）

jar cvf ./Q1SumDeptSalary.jar ./Q1SumDept.class*

*mv *.jar ../..*

rm Q1SumDept.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib
/commons-cli-1.2.jar Q1SumDeptSalary.java
[shiyanlou@b393a04554e1 class6]$ ll
total 16
-rw-rw-r-- 1 shiyanlou shiyanlou 2440 Jun  5 14:44 Q1SumDeptSalary.class
-rw-r--r-- 1 shiyanlou shiyanlou 3705 Jun  5 14:40 Q1SumDeptSalary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 3394 Jun  5 14:44 Q1SumDeptSalary$MapClass.class
-rw-rw-r-- 1 shiyanlou shiyanlou 1718 Jun  5 14:44 Q1SumDeptSalary$Reduce.class
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q1SumDeptSalary.jar ./Q1SumDept*.class
added manifest
adding: Q1SumDeptSalary.class(in = 2440) (out= 1144)(deflated 53%)
adding: Q1SumDeptSalary$MapClass.class(in = 3394) (out= 1487)(deflated 56%)
adding: Q1SumDeptSalary$Reduce.class(in = 1718) (out= 731)(deflated 57%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q1SumDept*.class
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ ls ../../
AvgTemperature.jar          hadoop-minicluster-1.1.2.jar  MinTemperature.jar
bin                         hadoop-test-1.1.2.jar         myclass
build.xml                   hadoop-tools-1.1.2.jar        NOTICE.txt
c++                         hdfs                          Q1SumDeptSalary.jar
CHANGES.txt                 input                         README.txt
conf                        ivy                           sbin
contrib                     ivy.xml                       share
hadoop-ant-1.1.2.jar        lib                           src
hadoop-client-1.1.2.jar     libexec                       tmp
hadoop-core-1.1.2.jar       LICENSE.txt                   webapps
hadoop-examples-1.1.2.jar   logs
[shiyanlou@b393a04554e1 class6]$
```

3.1.5 运行并查看结果

运行 Q1SumDeptSalary 时需要输入部门数据路径、员工数据路径和输出路径三个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 部门数据路径：hdfs://hadoop:9000/class6/input/dept，部门数据将缓存在各运行任务的节点内容中，可以提供处理的效率
- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out1

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q1SumDeptSalary.jar Q1SumDeptSalary

hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp

hdfs://hadoop:9000/class6/out1

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q1SumDeptSalary.jar Q1SumDeptSalary h
dfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9
000/class6/out1
15/06/05 14:47:56 INFO input.FileInputFormat: Total input paths to process : 1
15/06/05 14:47:56 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/05 14:47:56 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/05 14:47:57 INFO mapred.JobClient: Running job: job_201506040132_0004
15/06/05 14:47:58 INFO mapred.JobClient:  map 0% reduce 0%
15/06/05 14:48:04 INFO mapred.JobClient:  map 100% reduce 0%
15/06/05 14:48:12 INFO mapred.JobClient:  map 100% reduce 33%
15/06/05 14:48:13 INFO mapred.JobClient:  map 100% reduce 100%
15/06/05 14:48:14 INFO mapred.JobClient: Job complete: job_201506040132_0004
15/06/05 14:48:14 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out1 目录，打开 part-r-00000 文件

hadoop fs -ls /class6/out1

hadoop fs -cat /class6/out1/part-r-00000

可以看到运行结果：

ACCOUNTING 8750

RESEARCH 6775

SALES 9400

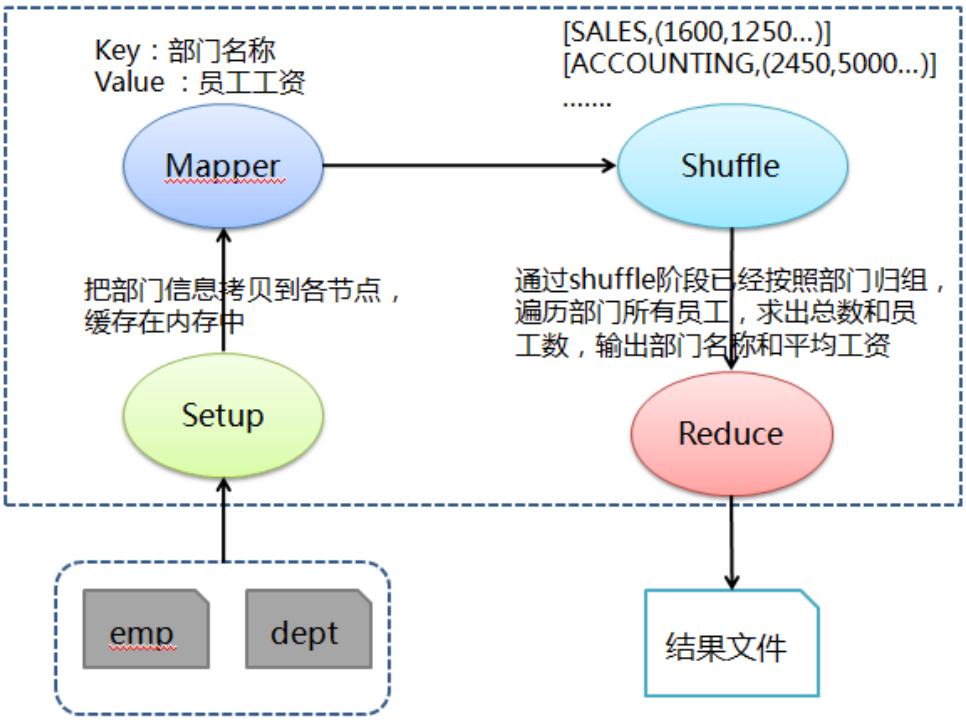
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out1
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-05 14:48 /class6/out1/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-05 14:47 /class6/out1/_logs
-rw-r--r--   1 shiyanlou supergroup         41 2015-06-05 14:48 /class6/out1/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out1/part-r-00000
ACCOUNTING      8750
RESEARCH        6775
SALES   9400
[shiyanlou@b393a04554e1 hadoop-1.1.2]$
```

3.2 测试例子 2：求各个部门的人数和平均工资

3.2.1 问题分析

求各个部门的人数和平均工资，需要得到各部门工资总数和部门人数，通过两者相除获取各部门平均工资。首先和问题 1 类似在 Mapper 的 Setup 阶段缓存部门数据，然后在 Mapper 阶段抽取出部门编号和员工工资，利用缓存部门数据把部门编号对应为部门名称，接着在 Shuffle 阶段把传过来的数据处理为部门名称对应该部门所有员工工资的列表，最后在 Reduce 中按照部门归组，遍历部门所有员工，求出总数和员工数，输出部门名称和平均工资。

3.2.2 处理流程图



3.2.3 编写代码

Q2DeptNumberAveSalary.java 代码：

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q2DeptNumberAveSalary extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

        // 用于缓存 dept文件中的数据
        private Map<String, String> deptMap = new HashMap<String, String>();
        private String[] kv;

        // 此方法会在Map方法执行之前执行且执行一次
        @Override
        protected void setup(Context context) throws IOException, InterruptedException {
            BufferedReader in = null;
            try {
                // 从当前作业中获取要缓存的文件
                Path[] paths = DistributedCache.getLocalCacheFiles(context.getConfiguration());
                String deptIdName = null;
                for (Path path : paths) {

                    // 对部门文件字段进行拆分并缓存到deptMap中
                    if (path.toString().contains("dept")) {
```



```
                    in = new BufferedReader(new FileReader(path.toString()));
                    while (null != (deptIdName = in.readLine())) {

                        // 对部门文件字段进行拆分并缓存到deptMap中
                        // 其中Map中key为部门编号，value为所在部门名称
                        deptMap.put(deptIdName.split(",")[0], deptIdName.split(",")[1]);
                    }
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                if (in != null) {
                    in.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    public void map(LongWritable key, Text value, Context context) throws IOException,
    InterruptedException {

        // 对员工文件字段进行拆分
        kv = value.toString().split(",");

        // map join: 在map阶段过滤掉不需要的数据，输出key为部门名称和value为员工工资
        if (deptMap.containsKey(kv[7])) {
            if (null != kv[5] && !"".equals(kv[5].toString())) {
                context.write(new Text(deptMap.get(kv[7].trim())), new Text(kv[5].trim()));
            }
        }
    }
}

public static class Reduce extends Reducer<Text, Text, Text, Text> {

   public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
    InterruptedException {

        long sumSalary = 0;
        int deptNumber = 0;

        // 对同一部门的员工工资进行求和
        for (Text val : values) {
```

```
            sumSalary += Long.parseLong(val.toString());
            deptNumber++;
        }

        // 输出key为部门名称和value为该部门员工工资平均值
        context.write(key, new Text("Dept Number:" + deptNumber + ", Ave Salary:" + sumSalary
        / deptNumber));
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称、Mapper和Reduce类
    Job job = new Job(getConf(), "Q2DeptNumberAveSalary");
    job.setJobName("Q2DeptNumberAveSalary");
    job.setJarByClass(Q2DeptNumberAveSalary.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(Reduce.class);

    // 设置输入格式类
    job.setInputFormatClass(TextInputFormat.class);

    // 设置输出格式类
    job.setOutputFormatClass(TextOutputFormat.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);

    // 第1个参数为缓存的部门数据路径、第2个参数为员工数据路径和第3个参数为输出路径
    String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
    args).getRemainingArgs();
    DistributedCache.addCacheFile(new Path(otherArgs[0]).toUri(), job.getConfiguration());
    FileInputFormat.addInputPath(job, new Path(otherArgs[1]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[2]));

    job.waitForCompletion(true);
    return job.isSuccessful() ? 0 : 1;
}

/**
 * 主方法，执行入口
 * @param args 输入参数
 */
public static void main(String[] args) throws Exception {
    int res = ToolRunner.run(new Configuration(), new Q2DeptNumberAveSalary(), args);
    System.exit(res);
}
```

}

3.2.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q2DeptNumberAveSalary.java 程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q2DeptNumberAveSalary.java 文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q2DeptNumberAveSalary.java

编译代码

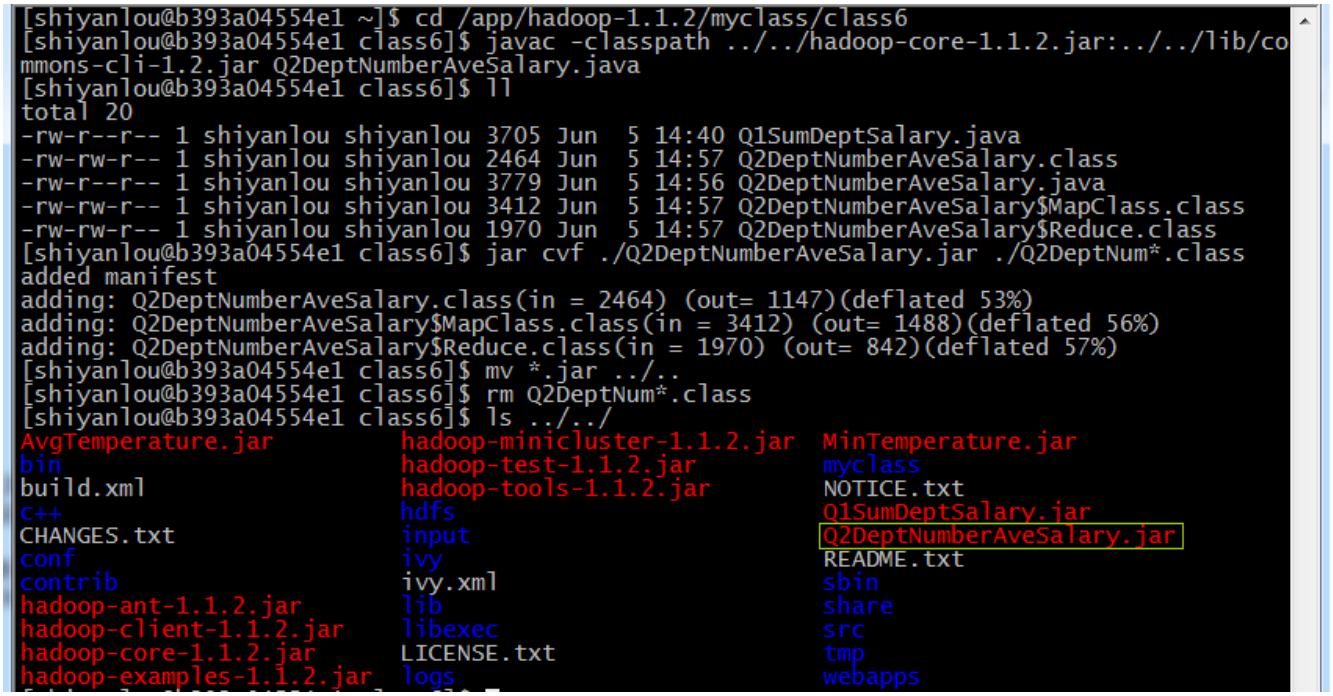
javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q2DeptNumberAveSalary.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q2DeptNumberAveSalary.jar ./Q2DeptNum.class*

*mv *.jar ../..*

rm Q2DeptNum.class*



3.2.5 运行并查看结果

运行 Q2DeptNumberAveSalary 时需要输入部门数据路径、员工数据路径和输出路径三个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 部门数据路径：hdfs://hadoop:9000/class6/input/dept，部门数据将缓存在各运行任务的节点内容中，可以提供处理的效率
- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out2

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q2DeptNumberAveSalary.jar Q2DeptNumberAveSalary hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out2

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q2DeptNumberAveSalary.jar Q2DeptNumberAv
eSalary hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp hdfs://ha
doop:9000/class6/out2
15/06/05 15:00:33 INFO input.FileInputFormat: Total input paths to process : 1
15/06/05 15:00:33 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/05 15:00:33 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/05 15:00:33 INFO mapred.JobClient: Running job: job_201506040132_0005
15/06/05 15:00:34 INFO mapred.JobClient:  map 0% reduce 0%
15/06/05 15:00:39 INFO mapred.JobClient:  map 100% reduce 0%
15/06/05 15:00:47 INFO mapred.JobClient:  map 100% reduce 33%
15/06/05 15:00:49 INFO mapred.JobClient:  map 100% reduce 100%
15/06/05 15:00:50 INFO mapred.JobClient: Job complete: job_201506040132_0005
15/06/05 15:00:50 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out2 目录

hadoop fs -ls /class6/out2

hadoop fs -cat /class6/out2/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

ACCOUNTING Dept Number:3,Ave Salary:2916

RESEARCH Dept Number:3,Ave Salary:2258

SALES Dept Number:6,Ave Salary:1566

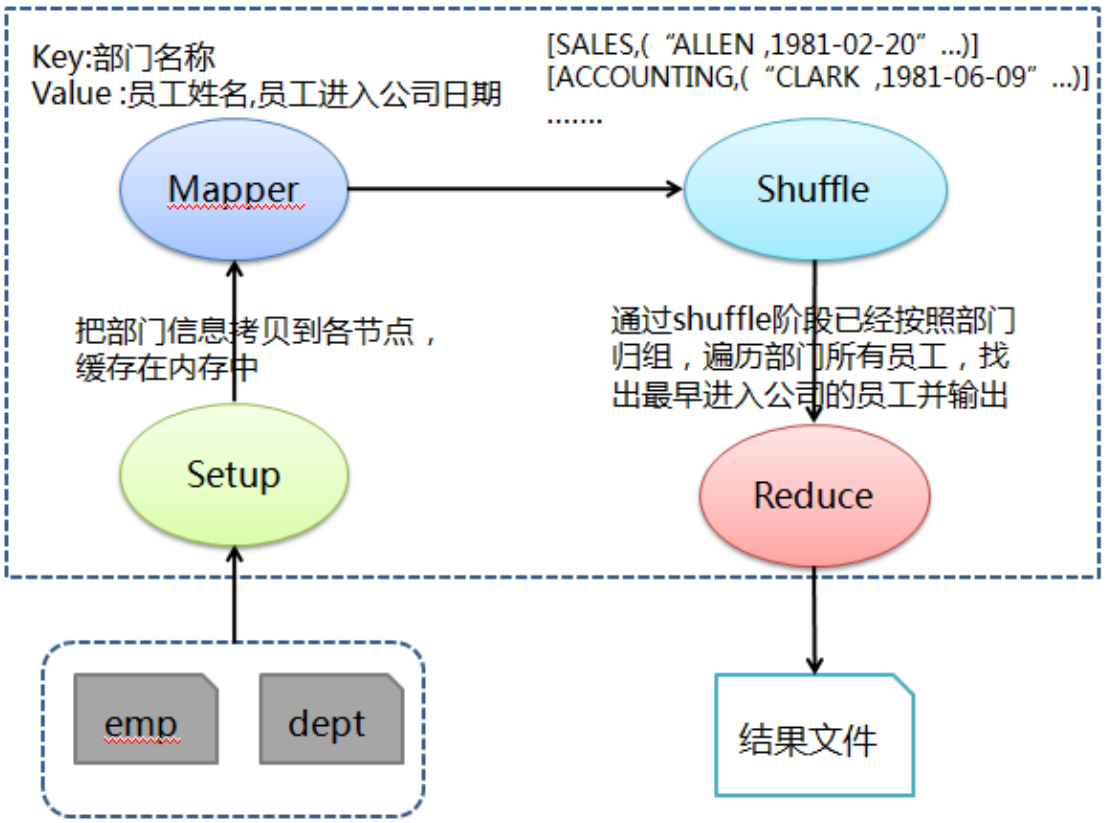
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out2
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-05 15:00 /class6/out2/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-05 15:00 /class6/out2/_logs
-rw-r--r--   1 shiyanlou supergroup        119 2015-06-05 15:00 /class6/out2/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out2/part-r-00000
ACCOUNTING      Dept Number:3, Ave Salary:2916
RESEARCH        Dept Number:3, Ave Salary:2258
SALES   Dept Number:6, Ave Salary:1566
[shiyanlou@b393a04554e1 hadoop-1.1.2]$
```

3.3 测试例子 3：求每个部门最早进入公司的员工姓名

3.3.1 问题分析

求每个部门最早进入公司员工姓名，需要得到各部门所有员工的进入公司日期，通过比较获取最早进入公司员工姓名。首先和问题 1 类似在 Mapper 的 Setup 阶段缓存部门数据，然后 Mapper 阶段抽取出 key 为部门名称（利用缓存部门数据把部门编号对应为部门名称），value 为员工姓名和进入公司日期，接着在 Shuffle 阶段把传过来的数据处理为部门名称对应该部门所有员工+进入公司日期的列表，最后在 Reduce 中按照部门归组，遍历部门所有员工，找出最早进入公司的员工并输出。

3.3.2 处理流程图



3.3.3 编写代码

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
```

```
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q3DeptEarliestEmp extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

        // 用于缓存 dept文件中的数据
        private Map<String, String> deptMap = new HashMap<String, String>();
        private String[] kv;

        // 此方法会在Map方法执行之前执行且执行一次
        @Override
        protected void setup(Context context) throws IOException, InterruptedException {
            BufferedReader in = null;
            try {
                // 从当前作业中获取要缓存的文件
                Path[] paths = DistributedCache.getLocalCacheFiles(context.getConfiguration());
                String deptIdName = null;
                for (Path path : paths) {
                    if (path.toString().contains("dept")) {
                        in = new BufferedReader(new FileReader(path.toString()));
                        while (null != (deptIdName = in.readLine())) {

                            // 对部门文件字段进行拆分并缓存到deptMap中
                            // 其中Map中key为部门编号，value为所在部门名称
                            deptMap.put(deptIdName.split(",")[0], deptIdName.split(",")[1]);
                        }
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    if (in != null) {
                        in.close();
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        public void map(LongWritable key, Text value, Context context) throws IOException,
    InterruptedException {
```

```
        // 对员工文件字段进行拆分
        kv = value.toString().split(",");

        // map join: 在map阶段过滤掉不需要的数据
        // 输出key为部门名称和value为员工姓名+","+员工进入公司日期
        if (deptMap.containsKey(kv[7])) {
            if (null != kv[4] && !"".equals(kv[4].toString())) {
                context.write(new Text(deptMap.get(kv[7].trim())), new Text(kv[1].trim()
                + "," + kv[4].trim()));
            }
        }
    }
}

public static class Reduce extends Reducer<Text, Text, Text, Text> {

    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
    InterruptedException {

        // 员工姓名和进入公司日期
        String empName = null;
        String empEnterDate = null;

        // 设置日期转换格式和最早进入公司的员工、日期
        DateFormat df = new SimpleDateFormat("dd-MM月-yy");

        Date earliestDate = new Date();
        String earliestEmp = null;

        // 遍历该部门下所有员工，得到最早进入公司的员工信息
        for (Text val : values) {
            empName = val.toString().split(",")[0];
            empEnterDate = val.toString().split(",")[1].toString().trim();
            try {
                System.out.println(df.parse(empEnterDate));
                if (df.parse(empEnterDate).compareTo(earliestDate) < 0) {
                    earliestDate = df.parse(empEnterDate);
                    earliestEmp = empName;
                }
            } catch (ParseException e) {
                e.printStackTrace();
            }
        }

        // 输出key为部门名称和value为该部门最早进入公司员工
        context.write(key, new Text("The earliest emp of dept:" + earliestEmp + ", Enter
        date:" + new SimpleDateFormat("yyyy-MM-dd").format(earliestDate)));
```

```
        }
    }

    @Override
    public int run(String[] args) throws Exception {

        // 实例化作业对象，设置作业名称
        Job job = new Job(getConf(), "Q3DeptEarliestEmp");
        job.setJobName("Q3DeptEarliestEmp");

        // 设置Mapper和Reduce类
        job.setJarByClass(Q3DeptEarliestEmp.class);
        job.setMapperClass(MapClass.class);
        job.setReducerClass(Reduce.class);

        // 设置输入格式类
        job.setInputFormatClass(TextInputFormat.class);

        // 设置输出格式类
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        // 第1个参数为缓存的部门数据路径、第2个参数为员工数据路径和第三个参数为输出路径
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        DistributedCache.addCacheFile(new Path(otherArgs[0]).toUri(),
        job.getConfiguration());
        FileInputFormat.addInputPath(job, new Path(otherArgs[1]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[2]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q3DeptEarliestEmp(), args);
        System.exit(res);
    }
}
```

3.3.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q3DeptEarliestEmp.java 程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q3DeptEarliestEmp.java 文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q3DeptEarliestEmp.java

编译代码

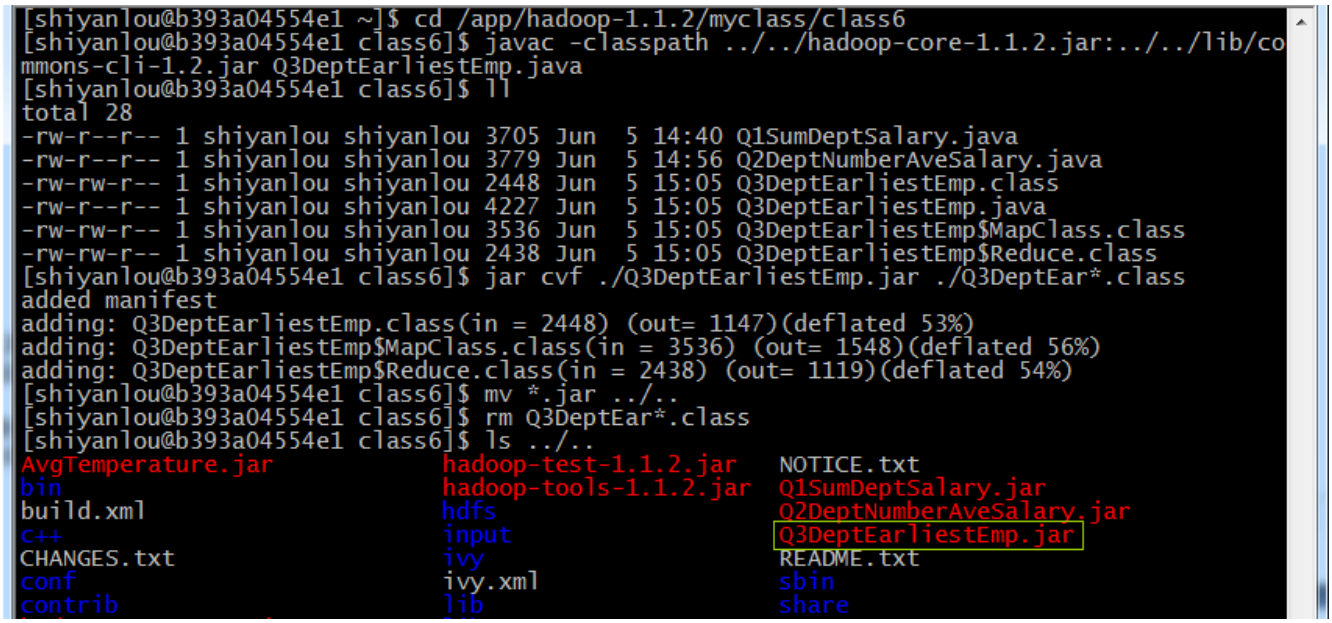
javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q3DeptEarliestEmp.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q3DeptEarliestEmp.jar ./Q3DeptEar.class*

*mv *.jar ../..*

rm Q3DeptEar.class*



3.3.5 运行并查看结果

运行 Q3DeptEarliestEmp 时需要输入部门数据路径、员工数据路径和输出路径三个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 部门数据路径：hdfs://hadoop:9000/class6/input/dept，部门数据将缓存在各运行任务的节点内容中，可以提供处理的效率
- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out3

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q3DeptEarliestEmp.jar Q3DeptEarliestEmp

hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp

hdfs://hadoop:9000/class6/out3

```
[shiyanlou@b393a04554e1 class6]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q3DeptEarliestEmp.jar Q3DeptEarliestEmp
hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:900
0/class6/out3
15/06/05 15:06:58 INFO input.FileInputFormat: Total input paths to process : 1
15/06/05 15:06:59 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/05 15:06:59 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/05 15:06:59 INFO mapred.JobClient: Running job: job_201506040132_0006
15/06/05 15:07:00 INFO mapred.JobClient:  map 0% reduce 0%
15/06/05 15:07:05 INFO mapred.JobClient:  map 100% reduce 0%
15/06/05 15:07:13 INFO mapred.JobClient:  map 100% reduce 33%
15/06/05 15:07:14 INFO mapred.JobClient:  map 100% reduce 100%
15/06/05 15:07:16 INFO mapred.JobClient: Job complete: job_201506040132_0006
15/06/05 15:07:16 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out3 目录

hadoop fs -ls /class6/out3

hadoop fs -cat /class6/out3/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

ACCOUNTING The earliest emp of dept:CLARK, Enter date:1981-06-09

RESEARCH The earliest emp of dept:SMITH, Enter date:1980-12-17

SALES The earliest emp of dept:ALLEN, Enter date:1981-02-20

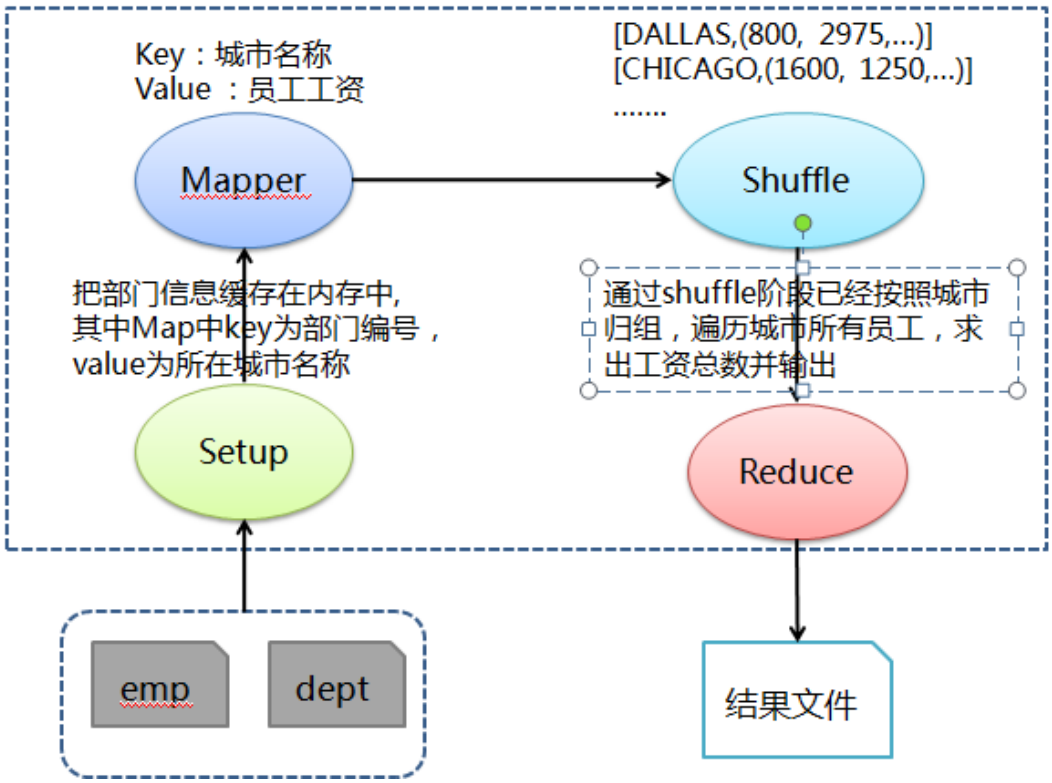
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out3
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-05 15:07 /class6/out3/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-05 15:06 /class6/out3/_logs
-rw-r--r--   1 shiyanlou supergroup        185 2015-06-05 15:07 /class6/out3/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out3/part-r-00000
ACCOUNTING      The earliest emp of dept:null, Enter date:2015-06-05
RESEARCH        The earliest emp of dept:null, Enter date:2015-06-05
SALES   The earliest emp of dept:null, Enter date:2015-06-05
```

3.4 测试例子 4：求各个城市的员工的总工资

3.4.1 问题分析

求各个城市员工的总工资，需要得到各个城市所有员工的工资，通过对各个城市所有员工工资求和得到总工资。首先和测试例子 1 类似在 Mapper 的 Setup 阶段缓存部门对应所在城市数据，然后在 Mapper 阶段抽取出 key 为城市名称（利用缓存数据把部门编号对应为所在城市名称），value 为员工工资，接着在 Shuffle 阶段把传过来的数据处理为城市名称对应该城市所有员工工资，最后在 Reduce 中按照城市归组，遍历城市所有员工，求出工资总数并输出。

3.4.2 处理流程图



3.4.3 编写代码

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q4SumCitySalary extends Configured implements Tool {
```

```
public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

    // 用于缓存 dept文件中的数据
    private Map<String, String> deptMap = new HashMap<String, String>();
    private String[] kv;

    // 此方法会在Map方法执行之前执行且执行一次
    @Override
    protected void setup(Context context) throws IOException, InterruptedException {
        BufferedReader in = null;
        try {
            // 从当前作业中获取要缓存的文件
            Path[] paths = DistributedCache.getLocalCacheFiles(context.getConfiguration());
            String deptIdName = null;
            for (Path path : paths) {
                if (path.toString().contains("dept")) {
                    in = new BufferedReader(new FileReader(path.toString()));
                    while (null != (deptIdName = in.readLine())) {

                        // 对部门文件字段进行拆分并缓存到deptMap中
                        // 其中Map中key为部门编号，value为所在城市名称
                        deptMap.put(deptIdName.split(",")[0], deptIdName.split(",")[2]);
                    }
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                if (in != null) {
                    in.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {

        // 对员工文件字段进行拆分
        kv = value.toString().split(",");

        // map join: 在map阶段过滤掉不需要的数据，输出key为城市名称和value为员工工资
        if (deptMap.containsKey(kv[7])) {
```



```
            if (null != kv[5] && !"".equals(kv[5].toString())) {
                context.write(new Text(deptMap.get(kv[7].trim())), new Text(kv[5].trim()));
            }
        }
    }
}

public static class Reduce extends Reducer<Text, Text, Text, LongWritable> {

    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
    InterruptedException {

        // 对同一城市的员工工资进行求和
        long sumSalary = 0;
        for (Text val : values) {
            sumSalary += Long.parseLong(val.toString());
        }

        // 输出key为城市名称和value为该城市工资总和
        context.write(key, new LongWritable(sumSalary));
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称
    Job job = new Job(getConf(), "Q4SumCitySalary");
    job.setJobName("Q4SumCitySalary");

    // 设置Mapper和Reduce类
    job.setJarByClass(Q4SumCitySalary.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(Reduce.class);

    // 设置输入格式类
    job.setInputFormatClass(TextInputFormat.class);

    // 设置输出格式类
    job.setOutputFormatClass(TextOutputFormat.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);

    // 第1个参数为缓存的部门数据路径、第2个参数为员工数据路径和第3个参数为输出路径
    String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
    args).getRemainingArgs();
    DistributedCache.addCacheFile(new Path(otherArgs[0]).toUri(),
```

```
        job.getConfiguration());
        FileInputFormat.addInputPath(job, new Path(otherArgs[1]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[2]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q4SumCitySalary(), args);
        System.exit(res);
    }
}
```

3.4.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q4SumCitySalary.java 程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q4SumCitySalary.java 文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q4SumCitySalary.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q4SumCitySalary.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q4SumCitySalary.jar ./Q4SumCity.class*

*mv *.jar ../..*

rm Q4SumCity.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q4SumCitySalary.java
[shiyanlou@b393a04554e1 class6]$ ll
total 32
-rw-r--r-- 1 shiyanlou shiyanlou 3705 Jun  5 14:40 Q1SumDeptSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3779 Jun  5 14:56 Q2DeptNumberAveSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 4227 Jun  5 15:05 Q3DeptEarliestEmp.java
-rw-rw-r-- 1 shiyanlou shiyanlou 2440 Jun  5 15:15 Q4SumCitySalary.class
-rw-r--r-- 1 shiyanlou shiyanlou 3616 Jun  5 15:15 Q4SumCitySalary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 3394 Jun  5 15:15 Q4SumCitySalary$MapClass.class
-rw-rw-r-- 1 shiyanlou shiyanlou 1718 Jun  5 15:15 Q4SumCitySalary$Reduce.class
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q4SumCitySalary.jar ./Q4SumCity*.class
added manifest
adding: Q4SumCitySalary.class(in = 2440) (out= 1147)(deflated 52%)
adding: Q4SumCitySalary$MapClass.class(in = 3394) (out= 1485)(deflated 56%)
adding: Q4SumCitySalary$Reduce.class(in = 1718) (out= 732)(deflated 57%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q4SumCity*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar            hadoop-test-1.1.2.jar   NOTICE.txt
bin                           hadoop-tools-1.1.2.jar  Q1SumDeptSalary.jar
build.xml                     hdfs                    Q2DeptNumberAveSalary.jar
c++                           input                   Q3DeptEarliestEmp.jar
CHANGES.txt                   ivy                     Q4SumCitySalary.jar
conf                          ivy.xml                 README.txt
contrib                       lib                     sbin
hadoop-ant-1.1.2.jar          libexec                 share
```

3.4.5 运行并查看结果

运行 Q4SumCitySalary 时需要输入部门数据路径、员工数据路径和输出路径三个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 部门数据路径：hdfs://hadoop:9000/class6/input/dept，部门数据将缓存在各运行任务的节点内容中，可以提供处理的效率
- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out4

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q4SumCitySalary.jar Q4SumCitySalary

hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp

hdfs://hadoop:9000/class6/out4

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q4SumCitySalary.jar Q4SumCitySalary hdfs
://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/cl
ass6/out4
15/06/05 15:17:02 INFO input.FileInputFormat: Total input paths to process : 1
15/06/05 15:17:02 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/05 15:17:02 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/05 15:17:02 INFO mapred.JobClient: Running job: job_201506040132_0007
15/06/05 15:17:03 INFO mapred.JobClient:  map 0% reduce 0%
15/06/05 15:17:08 INFO mapred.JobClient:  map 100% reduce 0%
15/06/05 15:17:16 INFO mapred.JobClient:  map 100% reduce 33%
15/06/05 15:17:17 INFO mapred.JobClient:  map 100% reduce 100%
15/06/05 15:17:18 INFO mapred.JobClient: Job complete: job_201506040132_0007
15/06/05 15:17:18 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out4 目录

hadoop fs -ls /class6/out4

hadoop fs -cat /class6/out4/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

CHICAGO 9400

DALLAS 6775

NEW YORK 8750

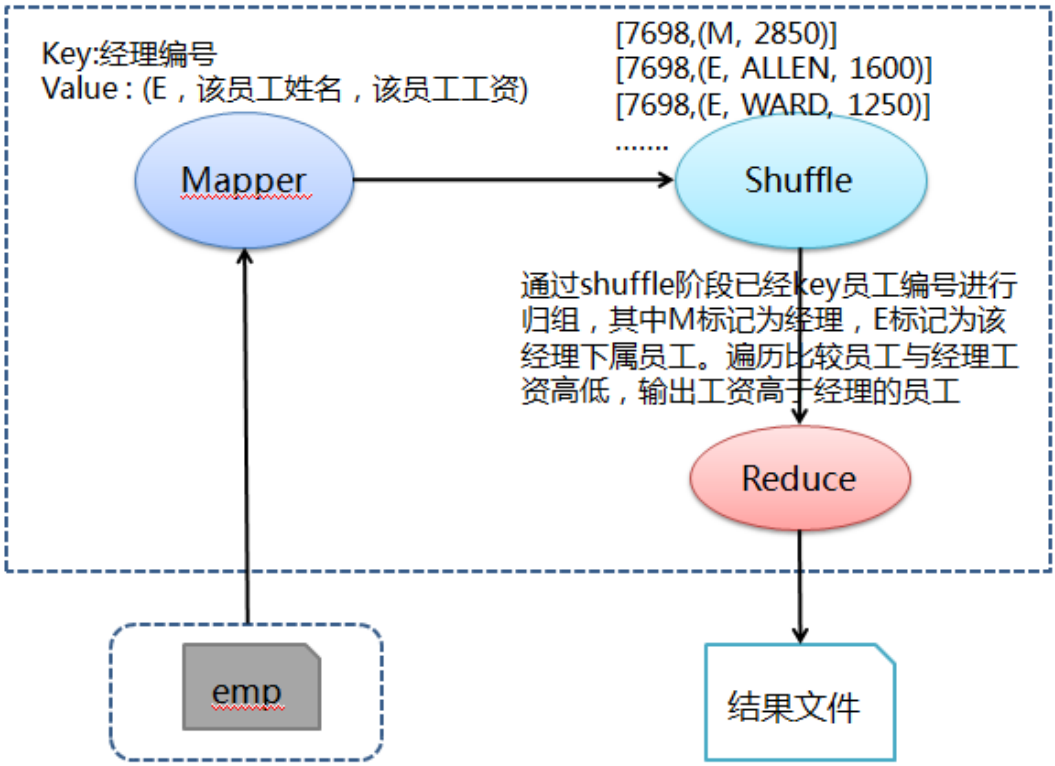
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out4
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-05 15:17 /class6/out4/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-05 15:17 /class6/out4/_logs
-rw-r--r--   1 shiyanlou supergroup         39 2015-06-05 15:17 /class6/out4/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out4/part-r-00000
CHICAGO 9400
DALLAS  6775
NEW YORK        8750
```

3.5 测试例子 5：列出工资比上司高的员工姓名及其工资

3.5.1 问题分析

求工资比上司高的员工姓名及工资，需要得到上司工资及上司所有下属员工，通过比较他们工资高低得到比上司工资高的员工。在 Mapper 阶段输出经理数据和员工对应经理表数据，其中经理数据 key 为员工编号、value 为"M ,该员工工资" ,员工对应经理表数据 key 为经理编号、value 为"E ,该员工姓名 ,该员工工资" ;然后在 Shuffle 阶段把传过来的经理数据和员工对应经理表数据进行归组，如编号为 7698 员工，value 中标志 M 为自己工资，value 中标志 E 为其下属姓名及工资；最后在 Reduce 中遍历比较员工与经理工资高低，输出工资高于经理的员工。

3.5.2 处理流程图



3.5.3 编写代码

```
import java.io.IOException;
import java.util.HashMap;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q5EarnMoreThanManager extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

        public void map(LongWritable key, Text value, Context context) throws IOException,
    InterruptedException {

            // 对员工文件字段进行拆分
            String[] kv = value.toString().split(",");

            // 输出经理表数据，其中key为员工编号和value为M+该员工工资
            context.write(new Text(kv[0].toString()), new Text("M," + kv[5]));

            // 输出员工对应经理表数据，其中key为经理编号和value为(E，该员工姓名，该员工工资)
            if (null != kv[3] && !"".equals(kv[3].toString())) {
                context.write(new Text(kv[3].toString()), new Text("E," + kv[1] + "," + kv[5]));
            }
        }
    }

    public static class Reduce extends Reducer<Text, Text, Text, Text> {

        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
        InterruptedException {

            // 定义员工姓名、工资和存放部门员工Map
```

```
        String empName;
        long empSalary = 0;
        HashMap<String, Long> empMap = new HashMap<String, Long>();

        // 定义经理工资变量
        long mgrSalary = 0;

        for (Text val : values) {
            if (val.toString().startsWith("E")) {
                // 当是员工标示时，获取该员工对应的姓名和工资并放入Map中
                empName = val.toString().split(",")[1];
                empSalary = Long.parseLong(val.toString().split(",")[2]);
                empMap.put(empName, empSalary);
            } else {
                // 当时经理标志时，获取该经理工资
                mgrSalary = Long.parseLong(val.toString().split(",")[1]);
            }
        }

        // 遍历该经理下属，比较员工与经理工资高低，输出工资高于经理的员工
        for (java.util.Map.Entry<String, Long> entry : empMap.entrySet()) {
            if (entry.getValue() > mgrSalary) {
                context.write(new Text(entry.getKey()), new Text("" + entry.getValue()));
            }
        }
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称
    Job job = new Job(getConf(), "Q5EarnMoreThanManager");
    job.setJobName("Q5EarnMoreThanManager");

    // 设置Mapper和Reduce类
    job.setJarByClass(Q5EarnMoreThanManager.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(Reduce.class);

    // 设置输入格式类
    job.setInputFormatClass(TextInputFormat.class);

    // 设置输出格式类
    job.setOutputFormatClass(TextOutputFormat.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
```

```
        // 第1个参数为员工数据路径和第2个参数为输出路径
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q5EarnMoreThanManager(), args);
        System.exit(res);
    }
}
```

3.5.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6目录中新建Q5EarnMoreThanManager.java程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q5EarnMoreThanManager.java文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q5EarnMoreThanManager.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q5EarnMoreThanManager.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q5EarnMoreThanManager.jar ./Q5EarnMore.class*

*mv *.jar ../..*

rm Q5EarnMore.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q5EarnMoreThanManager.java
[shiyanlou@b393a04554e1 class6]$ ll
total 36
-rw-r--r-- 1 shiyanlou shiyanlou 3705 Jun  5 14:40 Q1SumDeptSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3779 Jun  5 14:56 Q2DeptNumberAveSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 4227 Jun  5 15:05 Q3DeptEarliestEmp.java
-rw-r--r-- 1 shiyanlou shiyanlou 3616 Jun  5 15:15 Q4SumCitySalary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 2270 Jun  5 15:21 Q5EarnMoreThanManager.class
-rw-r--r-- 1 shiyanlou shiyanlou 3060 Jun  5 15:20 Q5EarnMoreThanManager.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1880 Jun  5 15:21 Q5EarnMoreThanManager$MapClass.class
-rw-rw-r-- 1 shiyanlou shiyanlou 2613 Jun  5 15:21 Q5EarnMoreThanManager$Reduce.class
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q5EarnMoreThanManager.jar ./Q5EarnMore*.class
added manifest
adding: Q5EarnMoreThanManager.class(in = 2270) (out= 1062)(deflated 53%)
adding: Q5EarnMoreThanManager$MapClass.class(in = 1880) (out= 757)(deflated 59%)
adding: Q5EarnMoreThanManager$Reduce.class(in = 2613) (out= 1158)(deflated 55%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q5EarnMore*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar          hadoop-test-1.1.2.jar   NOTICE.txt
bin                         hadoop-tools-1.1.2.jar  Q1SumDeptSalary.jar
build.xml                   hdfs                    Q2DeptNumberAveSalary.jar
c++                         input                   Q3DeptEarliestEmp.jar
CHANGES.txt                 ivy                     Q4SumCitySalary.jar
conf                        ivy.xml                 Q5EarnMoreThanManager.jar
contrib                     lib                     README.txt
hadoop-ant-1.1.2.jar        libexec                 sbin
```

3.5.5 运行并查看结果

运行 Q5EarnMoreThanManager 运行的员工数据路径和输出路径两个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out5

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q5EarnMoreThanManager.jar Q5EarnMoreThanManager

hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out5

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q5EarnMoreThanManager.jar Q5EarnMoreThan
Manager hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out5
15/06/05 15:22:51 INFO input.FileInputFormat: Total input paths to process : 1
15/06/05 15:22:51 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/05 15:22:51 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/05 15:22:51 INFO mapred.JobClient: Running job: job_201506040132_0008
15/06/05 15:22:52 INFO mapred.JobClient:  map 0% reduce 0%
15/06/05 15:22:57 INFO mapred.JobClient:  map 100% reduce 0%
15/06/05 15:23:05 INFO mapred.JobClient:  map 100% reduce 33%
15/06/05 15:23:06 INFO mapred.JobClient:  map 100% reduce 100%
15/06/05 15:23:07 INFO mapred.JobClient: Job complete: job_201506040132_0008
15/06/05 15:23:07 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out5 目录

hadoop fs -ls /class6/out5

hadoop fs -cat /class6/out5/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

FORD 3000

```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out5
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-05 15:23 /class6/out5/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-05 15:22 /class6/out5/_logs
-rw-r--r--   1 shiyanlou supergroup         10 2015-06-05 15:23 /class6/out5/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out5/part-r-00000
FORD    3000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$
```

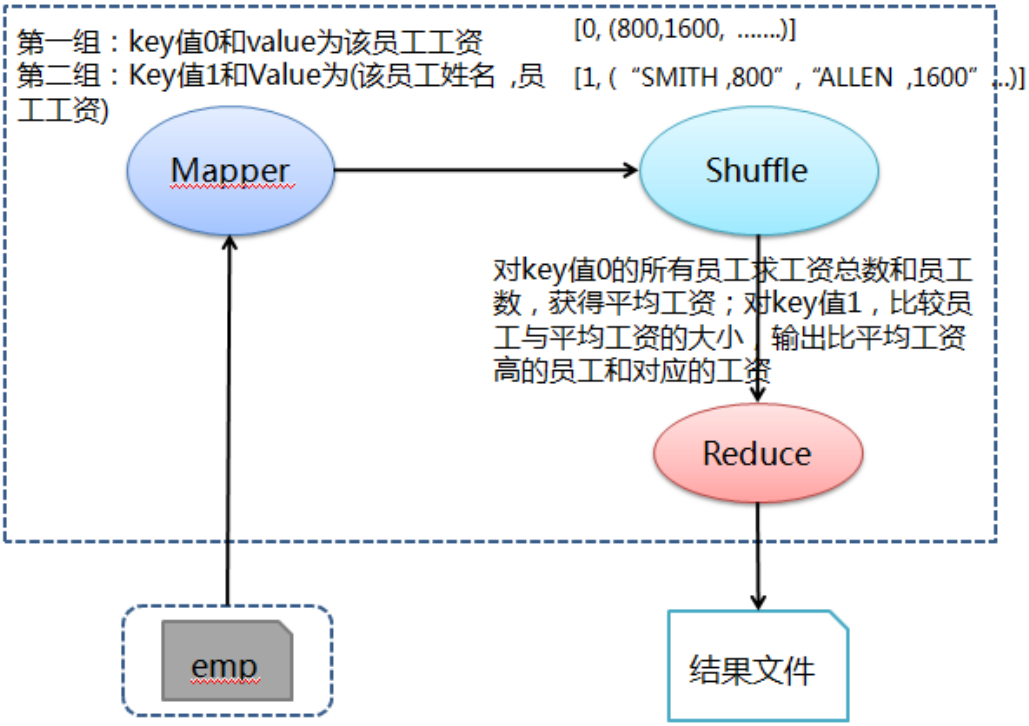
3.6 测试例子 6：列出工资比公司平均工资要高的员工姓名及其工资

3.6.1 问题分析

求工资比公司平均工资要高的员工姓名及工资，需要得到公司的平均工资和所有员工工资，通过比较得出工资比平均工资高的员工姓名及工资。这个问题可以分两个作业进行解决，先求出公司的平均工资，然后与所有员工进行比较得到结果；也可以在一个作业进行解决，这里就得使用作业 setNumReduceTasks 方法，设置 Reduce 任务数为 1，保证每次运行一个 reduce 任务，从而能先求出平均工资，然后进行比较得出结果。

在 Mapper 阶段输出两份所有员工数据，其中一份 key 为 0、value 为该员工工资，另外一份 key 为 0、value 为"该员工姓名 ,员工工资"；然后在 Shuffle 阶段把传过来数据按照 key 进行归组，在该任务中有 key 值为 0 和 1 两组数据；最后在 Reduce 中对 key 值 0 的所有员工求工资总数和员工数，获得平均工资；对 key 值 1，比较员工与平均工资的大小，输出比平均工资高的员工和对应的工资。

3.6.2 处理流程图



3.6.3 编写代码

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q6HigherThanAveSalary extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, IntWritable, Text> {

        public void map(LongWritable key, Text value, Context context) throws IOException,
    InterruptedException {

            // 对员工文件字段进行拆分
            String[] kv = value.toString().split(",");

            // 获取所有员工数据，其中key为0和value为该员工工资
            context.write(new IntWritable(0), new Text(kv[5]));

            // 获取所有员工数据，其中key为0和value为(该员工姓名 ,员工工资)
            context.write(new IntWritable(1), new Text(kv[1] + "," + kv[5]));
        }
    }

    public static class Reduce extends Reducer<IntWritable, Text, Text, Text> {

        // 定义员工工资、员工数和平均工资
        private long allSalary = 0;
        private int allEmpCount = 0;
        private long aveSalary = 0;

        // 定义员工工资变量
        private long empSalary = 0;
```

```
    public void reduce(IntWritable key, Iterable<Text> values, Context context) throws
IOException, InterruptedException {

        for (Text val : values) {
            if (0 == key.get()) {
                // 获取所有员工工资和员工数
                allSalary += Long.parseLong(val.toString());
                allEmpCount++;
                System.out.println("allEmpCount = " + allEmpCount);
            } else if (1 == key.get()) {
                if (aveSalary == 0) {
                    aveSalary = allSalary / allEmpCount;
                    context.write(new Text("Average Salary = "), new Text("" + aveSalary));
                    context.write(new Text("Following employees have salarys higher than
                    Average:"), new Text(""));
                }

                // 获取员工的平均工资
                System.out.println("Employee salary = " + val.toString());
                aveSalary = allSalary / allEmpCount;

                // 比较员工与平均工资的大小，输出比平均工资高的员工和对应的工资
                empSalary = Long.parseLong(val.toString().split(",")[1]);
                if (empSalary > aveSalary) {
                    context.write(new Text(val.toString().split(",")[0]), new Text("" +
                    empSalary));
                }
            }
        }
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称
    Job job = new Job(getConf(), "Q6HigherThanAveSalary");
    job.setJobName("Q6HigherThanAveSalary");

    // 设置Mapper和Reduce类
    job.setJarByClass(Q6HigherThanAveSalary.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(Reduce.class);

    // 必须设置Reduce任务数为1 # -D mapred.reduce.tasks = 1
    // 这是该作业设置的核心，这样才能够保证各reduce是串行的
    job.setNumReduceTasks(1);
```

```
        // 设置输出格式类
        job.setMapOutputKeyClass(IntWritable.class);
        job.setMapOutputValueClass(Text.class);

        // 设置输出键和值类型
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(LongWritable.class);

        // 第1个参数为员工数据路径和第2个参数为输出路径
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q6HigherThanAveSalary(), args);
        System.exit(res);
    }
}
```

3.6.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q5EarnMoreThanManager.java 程序代码 (代码页可以使用/home/shiyanlou/install-pack/class6/Q6HigherThanAveSalary.java 文件)

cd /app/hadoop-1.1.2/myclass/class6

vi Q6HigherThanAveSalary.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q6HigherThanAveSalary.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q6HigherThanAveSalary.jar ./Q6HigherThan.class*

*mv *.jar ../..*

rm Q6HigherThan.class*

```
[shiyanlou@b393a04554e1 class6]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q6HigherThanAveSalary.java
[shiyanlou@b393a04554e1 class6]$ ll
total 40
-rw-r--r-- 1 shiyanlou shiyanlou 3705 Jun  5 14:40 Q1SumDeptSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3779 Jun  5 14:56 Q2DeptNumberAveSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 4227 Jun  5 15:05 Q3DeptEarliestEmp.java
-rw-r--r-- 1 shiyanlou shiyanlou 3616 Jun  5 15:15 Q4SumCitySalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3060 Jun  5 15:20 Q5EarnMoreThanManager.java
-rw-rw-r-- 1 shiyanlou shiyanlou 2374 Jun  6 13:46 Q6HigherThanAveSalary.class
-rw-r--r-- 1 shiyanlou shiyanlou 3292 Jun  6 13:46 Q6HigherThanAveSalary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1762 Jun  6 13:46 Q6HigherThanAveSalary$MapClass.class
-rw-rw-r-- 1 shiyanlou shiyanlou 2783 Jun  6 13:46 Q6HigherThanAveSalary$Reduce.class
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q6HigherThanAveSalary.jar ./Q6HigherThan*.class
added manifest
adding: Q6HigherThanAveSalary.class(in = 2374) (out= 1123)(deflated 52%)
adding: Q6HigherThanAveSalary$MapClass.class(in = 1762) (out= 689)(deflated 60%)
adding: Q6HigherThanAveSalary$Reduce.class(in = 2783) (out= 1263)(deflated 54%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q6HigherThan*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar            hadoop-tools-1.1.2.jar  Q2DeptNumberAveSalary.jar
bin                           hdfs                    Q3DeptEarliestEmp.jar
build.xml                     input                   Q4SumCitySalary.jar
c++                           ivy                     Q5EarnMoreThanManager.jar
CHANGES.txt                   ivy.xml                 Q6HigherThanAveSalary.jar
conf                          lib                     README.txt
contrib                       libexec                 sbin
```

3.6.5 运行并查看结果

运行 Q6HigherThanAveSalary 运行的员工数据路径和输出路径两个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out6

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q6HigherThanAveSalary.jar Q6HigherThanAveSalary

hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out6

```
[shiyanlou@b393a04554e1 class6]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q6HigherThanAveSalary.jar Q6HigherThanAv
eSalary hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out6
15/06/06 13:50:33 INFO input.FileInputFormat: Total input paths to process : 1
15/06/06 13:50:33 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/06 13:50:33 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/06 13:50:33 INFO mapred.JobClient: Running job: job_201506040132_0009
15/06/06 13:50:34 INFO mapred.JobClient:  map 0% reduce 0%
15/06/06 13:50:39 INFO mapred.JobClient:  map 100% reduce 0%
15/06/06 13:50:48 INFO mapred.JobClient:  map 100% reduce 33%
15/06/06 13:50:50 INFO mapred.JobClient:  map 100% reduce 100%
15/06/06 13:50:51 INFO mapred.JobClient: Job complete: job_201506040132_0009
15/06/06 13:50:51 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out6 目录

hadoop fs -ls /class6/out6

hadoop fs -cat /class6/out6/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

Average Salary = 2077

Following employees have salarys higher than Average:

FORD 3000

CLARK 2450

KING 5000

JONES 2975

BLAKE 2850

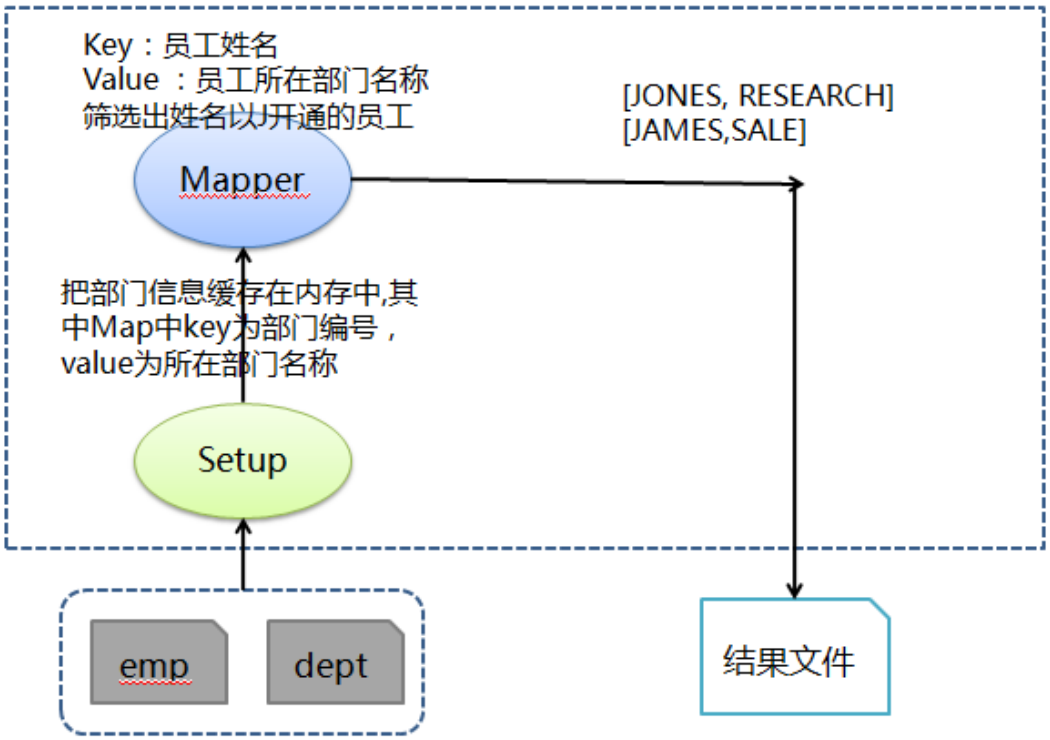
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out6
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-06 13:50 /class6/out6/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-06 13:50 /class6/out6/_logs
-rw-r--r--   1 shiyanlou supergroup        131 2015-06-06 13:50 /class6/out6/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out6/part-r-00000
Average Salary =        2077
Following employees have salarys higher than Average:
FORD    3000
CLARK   2450
KING    5000
JONES   2975
BLAKE   2850
```

3.7 测试例子 7：列出名字以 J 开头的员工姓名及其所属部门名称

3.7.1 问题分析

求名字以 J 开头的员工姓名机器所属部门名称，只需判断员工姓名是否以 J 开头。首先和问题 1 类似在 Mapper 的 Setup 阶段缓存部门数据，然后在 Mapper 阶段判断员工姓名是否以 J 开头，如果是抽取出员工姓名和员工所在部门编号，利用缓存部门数据把部门编号对应为部门名称，转换后输出结果。

3.7.2 处理流程图



3.7.3 编写代码

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q7NameDeptOfStartJ extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

        // 用于缓存 dept文件中的数据
        private Map<String, String> deptMap = new HashMap<String, String>();
        private String[] kv;

        // 此方法会在Map方法执行之前执行且执行一次
        @Override
        protected void setup(Context context) throws IOException, InterruptedException {
            BufferedReader in = null;
            try {

                // 从当前作业中获取要缓存的文件
                Path[] paths = DistributedCache.getLocalCacheFiles(context.getConfiguration());
                String deptIdName = null;
                for (Path path : paths) {

                    // 对部门文件字段进行拆分并缓存到deptMap中
                    if (path.toString().contains("dept")) {
                        in = new BufferedReader(new FileReader(path.toString()));
                        while (null != (deptIdName = in.readLine())) {
```



```
                        // 对部门文件字段进行拆分并缓存到deptMap中
                        // 其中Map中key为部门编号，value为所在部门名称
                        deptMap.put(deptIdName.split(",")[0], deptIdName.split(",")[1]);
                    }
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                if (in != null) {
                    in.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {

        // 对员工文件字段进行拆分
        kv = value.toString().split(",");

        // 输出员工姓名为J开头的员工信息，key为员工姓名和value为员工所在部门名称
        if (kv[1].toString().trim().startsWith("J")) {
            context.write(new Text(kv[1].trim()), new Text(deptMap.get(kv[7].trim())));
        }
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称
    Job job = new Job(getConf(), "Q7NameDeptOfStartJ");
    job.setJobName("Q7NameDeptOfStartJ");

    // 设置Mapper和Reduce类
    job.setJarByClass(Q7NameDeptOfStartJ.class);
    job.setMapperClass(MapClass.class);

    // 设置输入格式类
    job.setInputFormatClass(TextInputFormat.class);
```

```
        // 设置输出格式类
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        // 第1个参数为缓存的部门数据路径、第2个参数为员工数据路径和第3个参数为输出路径
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        DistributedCache.addCacheFile(new Path(otherArgs[0]).toUri(),
        job.getConfiguration());
        FileInputFormat.addInputPath(job, new Path(otherArgs[1]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[2]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q7NameDeptOfStartJ(), args);
        System.exit(res);
    }
}
```

3.7.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q7NameDeptOfStartJ.java 程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q7NameDeptOfStartJ.java 文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q7NameDeptOfStartJ.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q7NameDeptOfStartJ.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q7NameDeptOfStartJ.jar ./Q7NameDept.class*

*mv *.jar ../..*

rm Q7NameDept.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q7NameDeptOfStartJ.java
[shiyanlou@b393a04554e1 class6]$ ll
total 24
-rw-r--r-- 1 shiyanlou shiyanlou 3616 Jun  5 15:15 Q4SumCitySalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3060 Jun  5 15:20 Q5EarnMoreThanManager.java
-rw-r--r-- 1 shiyanlou shiyanlou 3292 Jun  6 13:46 Q6HigherThanAveSalary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 2365 Jun  6 13:59 Q7NameDeptOfStartJ.class
-rw-r--r-- 1 shiyanlou shiyanlou 3098 Jun  6 13:55 Q7NameDeptOfStartJ.java
-rw-rw-r-- 1 shiyanlou shiyanlou 3354 Jun  6 13:59 Q7NameDeptOfStartJ$MapClass.class
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q7NameDeptOfStartJ.jar ./Q7NameDept*.class
added manifest
adding: Q7NameDeptOfStartJ.class(in = 2365) (out= 1105)(deflated 53%)
adding: Q7NameDeptOfStartJ$MapClass.class(in = 3354) (out= 1458)(deflated 56%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q7NameDept*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar            hadoop-tools-1.1.2.jar  Q2DeptNumberAveSalary.jar
bin                           hdfs                    Q3DeptEarliestEmp.jar
build.xml                     input                   Q4SumCitySalary.jar
c++                           ivy                     Q5EarnMoreThanManager.jar
CHANGES.txt                   ivy.xml                 Q6HigherThanAveSalary.jar
conf                          lib                     Q7NameDeptOfStartJ.jar
contrib                       libexec                 README.txt
hadoop-ant-1.1.2.jar          LICENSE.txt             sbin
```

3.7.5 运行并查看结果

运行 Q7NameDeptOfStartJ 时需要输入部门数据路径、员工数据路径和输出路径三个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 部门数据路径：hdfs://hadoop:9000/class6/input/dept，部门数据将缓存在各运行任务的节点内容中，可以提供处理的效率
- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out7

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q7NameDeptOfStartJ.jar Q7NameDeptOfStartJ

hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp

hdfs://hadoop:9000/class6/out7

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q7NameDeptOfStartJ.jar Q7NameDeptOfStart
J hdfs://hadoop:9000/class6/input/dept hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9
000/class6/out7
15/06/06 14:04:46 INFO input.FileInputFormat: Total input paths to process : 1
15/06/06 14:04:46 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/06 14:04:46 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/06 14:04:46 INFO mapred.JobClient: Running job: job_201506040132_0010
15/06/06 14:04:47 INFO mapred.JobClient:  map 0% reduce 0%
15/06/06 14:04:52 INFO mapred.JobClient:  map 100% reduce 0%
15/06/06 14:05:00 INFO mapred.JobClient:  map 100% reduce 33%
15/06/06 14:05:02 INFO mapred.JobClient:  map 100% reduce 100%
15/06/06 14:05:03 INFO mapred.JobClient: Job complete: job_201506040132_0010
15/06/06 14:05:03 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out7 目录

hadoop fs -ls /class6/out7

hadoop fs -cat /class6/out7/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

JAMES SALES

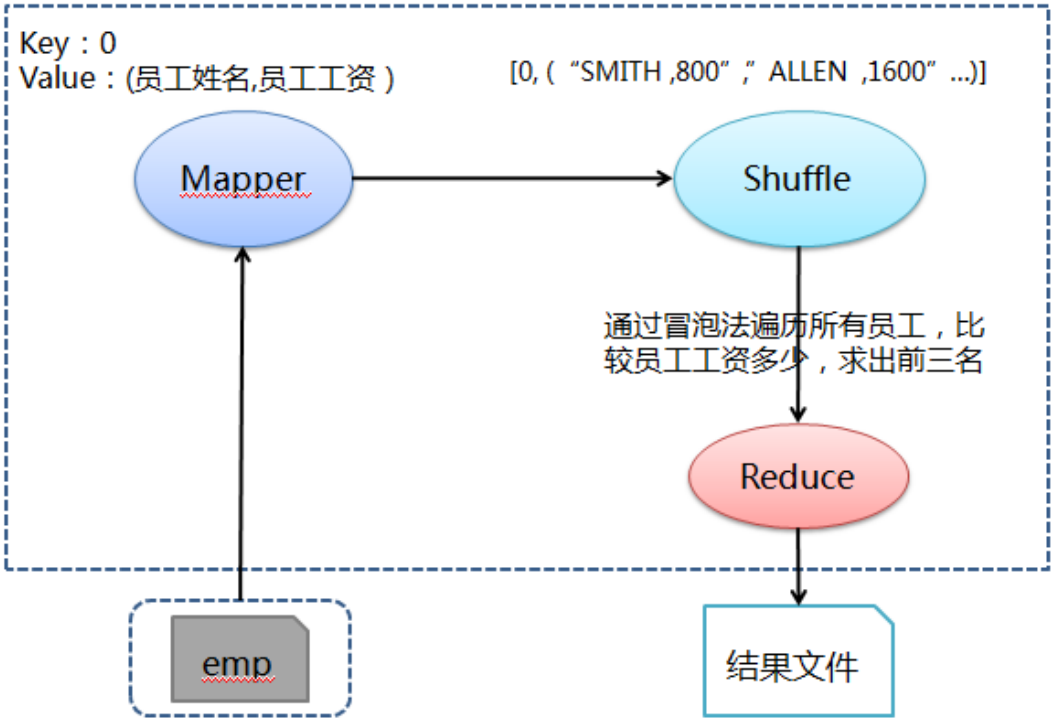
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out7
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-06 14:05 /class6/out7/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-06 14:04 /class6/out7/_logs
-rw-r--r--   1 shiyanlou supergroup         27 2015-06-06 14:04 /class6/out7/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out7/part-r-00000
JAMES   SALES
JONES   RESEARCH
```

3.8 测试例子 8：列出工资最高的头三名员工姓名及其工资

3.8.1 问题分析

求工资最高的头三名员工姓名及工资，可以通过冒泡法得到。在 Mapper 阶段输出经理数据和员工对应经理表数据，其中经理数据 key 为 0 值、value 为"员工姓名，员工工资"；最后在 Reduce 中通过冒泡法遍历所有员工，比较员工工资多少，求出前三名。

3.8.2 处理流程图



3.8.3 编写代码

```
import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
```

```
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q8SalaryTop3Salary extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, IntWritable, Text> {

        public void map(LongWritable key, Text value, Context context) throws IOException,
    InterruptedException {

            // 对员工文件字段进行拆分
            String[] kv = value.toString().split(",");

            // 输出key为0和value为员工姓名+","+员工工资
            context.write(new IntWritable(0), new Text(kv[1].trim() + "," + kv[5].trim()));
        }
    }

    public static class Reduce extends Reducer<IntWritable, Text, Text, Text> {

        public void reduce(IntWritable key, Iterable<Text> values, Context context) throws
    IOException, InterruptedException {

            // 定义工资前三员工姓名
            String empName;
            String firstEmpName = "";
            String secondEmpName = "";
            String thirdEmpName = "";

            // 定义工资前三工资
            long empSalary = 0;
            long firstEmpSalary = 0;
            long secondEmpSalary = 0;
            long thirdEmpSalary = 0;

            // 通过冒泡法遍历所有员工，比较员工工资多少，求出前三名
            for (Text val : values) {
                empName = val.toString().split(",")[0];
                empSalary = Long.parseLong(val.toString().split(",")[1]);
```

```
            if(empSalary > firstEmpSalary) {
                thirdEmpName = secondEmpName;
                thirdEmpSalary = secondEmpSalary;
                secondEmpName = firstEmpName;
                secondEmpSalary = firstEmpSalary;
                firstEmpName = empName;
                firstEmpSalary = empSalary;
            } else if (empSalary > secondEmpSalary) {
                thirdEmpName = secondEmpName;
                thirdEmpSalary = secondEmpSalary;
                secondEmpName = empName;
                secondEmpSalary = empSalary;
            } else if (empSalary > thirdEmpSalary) {
                thirdEmpName = empName;
                thirdEmpSalary = empSalary;
            }
        }

        // 输出工资前三名信息
        context.write(new Text( "First employee name:" + firstEmpName), new Text("Salary:"
        + firstEmpSalary));
        context.write(new Text( "Second employee name:" + secondEmpName), new
        Text("Salary:" + secondEmpSalary));
        context.write(new Text( "Third employee name:" + thirdEmpName), new Text("Salary:"
        + thirdEmpSalary));
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称
    Job job = new Job(getConf(), "Q8SalaryTop3Salary");
    job.setJobName("Q8SalaryTop3Salary");

    // 设置Mapper和Reduce类
    job.setJarByClass(Q8SalaryTop3Salary.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(Reduce.class);
    job.setMapOutputKeyClass(IntWritable.class);
    job.setMapOutputValueClass(Text.class);

    // 设置输入格式类
    job.setInputFormatClass(TextInputFormat.class);

    // 设置输出格式类
    job.setOutputKeyClass(Text.class);
```

```
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setOutputValueClass(Text.class);

        // 第1个参数为员工数据路径和第2个参数为输出路径
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q8SalaryTop3Salary(), args);
        System.exit(res);
    }
}
```

3.8.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q8SalaryTop3Salary.java 程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q8SalaryTop3Salary.java 文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q8SalaryTop3Salary.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q8SalaryTop3Salary.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q8SalaryTop3Salary.jar ./Q8SalaryTop3.class*

*mv *.jar ../..*

rm Q8SalaryTop3.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q8SalaryTop3Salary.java
[shiyanlou@b393a04554e1 class6]$ ll
total 32
-rw-r--r-- 1 shiyanlou shiyanlou 3616 Jun  5 15:15 Q4SumCitySalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3060 Jun  5 15:20 Q5EarnMoreThanManager.java
-rw-r--r-- 1 shiyanlou shiyanlou 3292 Jun  6 13:46 Q6HigherThanAveSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3098 Jun  6 13:55 Q7NameDeptOfStartJ.java
-rw-rw-r-- 1 shiyanlou shiyanlou 2386 Jun  6 14:11 Q8SalaryTop3Salary.class
-rw-r--r-- 1 shiyanlou shiyanlou 3633 Jun  6 14:10 Q8SalaryTop3Salary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1749 Jun  6 14:11 Q8SalaryTop3Salary$MapClass.class
-rw-rw-r-- 1 shiyanlou shiyanlou 2584 Jun  6 14:11 Q8SalaryTop3Salary$Reduce.class
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q8SalaryTop3Salary.jar ./Q8SalaryTop3*.class
added manifest
adding: Q8SalaryTop3Salary.class(in = 2386) (out= 1109)(deflated 53%)
adding: Q8SalaryTop3Salary$MapClass.class(in = 1749) (out= 687)(deflated 60%)
adding: Q8SalaryTop3Salary$Reduce.class(in = 2584) (out= 1125)(deflated 56%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q8SalaryTop3*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar            hadoop-tools-1.1.2.jar  Q2DeptNumberAveSalary.jar
bin                           hdfs                    Q3DeptEarliestEmp.jar
build.xml                     input                   Q4SumCitySalary.jar
c++                           ivy                     Q5EarnMoreThanManager.jar
CHANGES.txt                   ivy.xml                 Q6HigherThanAveSalary.jar
conf                          lib                     Q7NameDeptOfStartJ.jar
contrib                       libexec                 Q8SalaryTop3Salary.jar
hadoop-ant-1.1.2.jar          LICENSE.txt             README.txt
hadoop-client-1.1.2.jar       logs                    sbin
```

3.8.5 运行并查看结果

运行 Q8SalaryTop3Salary 运行的员工数据路径和输出路径两个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out8

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q8SalaryTop3Salary.jar Q8SalaryTop3Salary

hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out8

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q8SalaryTop3Salary.jar Q8SalaryTop3Salar
y hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out8
15/06/06 14:13:30 INFO input.FileInputFormat: Total input paths to process : 1
15/06/06 14:13:30 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/06 14:13:30 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/06 14:13:30 INFO mapred.JobClient: Running job: job_201506040132_0011
15/06/06 14:13:31 INFO mapred.JobClient:  map 0% reduce 0%
15/06/06 14:13:36 INFO mapred.JobClient:  map 100% reduce 0%
15/06/06 14:13:44 INFO mapred.JobClient:  map 100% reduce 33%
15/06/06 14:13:46 INFO mapred.JobClient:  map 100% reduce 100%
15/06/06 14:13:47 INFO mapred.JobClient: Job complete: job_201506040132_0011
15/06/06 14:13:48 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out8 目录

hadoop fs -ls /class6/out8

hadoop fs -cat /class6/out8/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

First employee name:KING Salary:5000

Second employee name:FORD Salary:3000

Third employee name:JONES Salary:2975

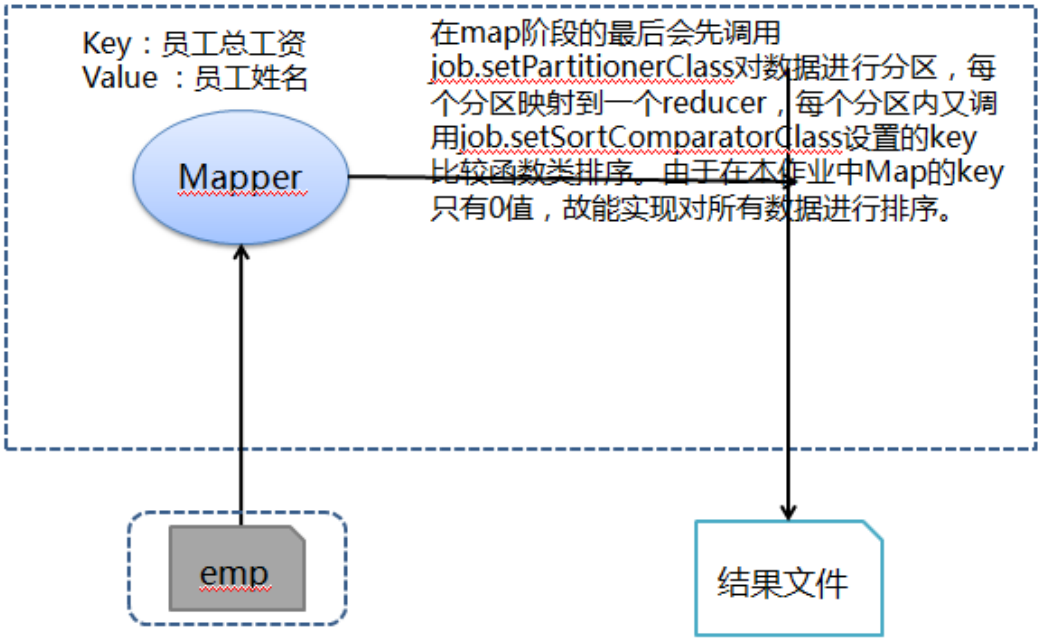
```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out8
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-06 14:13 /class6/out8/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-06 14:13 /class6/out8/_logs
-rw-r--r--   1 shiyanlou supergroup        113 2015-06-06 14:13 /class6/out8/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out8/part-r-00000
First employee name:KING        Salary:5000
Second employee name:FORD       Salary:3000
Third employee name:JONES       Salary:2975
[shiyanlou@b393a04554e1 hadoop-1.1.2]$
```

3.9 测试例子 9：将全体员工按照总收入（工资+提成）从高到低排列

3.9.1 问题分析

求全体员工总收入降序排列，获得所有员工总收入并降序排列即可。在 Mapper 阶段输出所有员工总工资数据，其中 key 为员工总工资、value 为员工姓名，在 Mapper 阶段的最后会先调用 job.setPartitionerClass 对数据进行分区，每个分区映射到一个 reducer，每个分区内又调用 job.setSortComparatorClass 设置的 key 比较函数类排序。由于在本作业中 Map 的 key 只有 0 值，故能实现对所有数据进行排序。

3.9.2 处理流程图



3.9.3 编写代码

```
import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
```

```
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q9EmpSalarySort extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, IntWritable, Text> {

        public void map(LongWritable key, Text value, Context context) throws IOException,
    InterruptedException {

            // 对员工文件字段进行拆分
            String[] kv = value.toString().split(",");

            // 输出key为员工所有工资和value为员工姓名
            int empAllSalary = "".equals(kv[6]) ? Integer.parseInt(kv[5]) :
            Integer.parseInt(kv[5]) + Integer.parseInt(kv[6]);
            context.write(new IntWritable(empAllSalary), new Text(kv[1]));
        }
    }

    /**
     * 递减排序算法
     */
    public static class DecreaseComparator extends IntWritable.Comparator {
        public int compare(WritableComparable a, WritableComparable b) {
            return -super.compare(a, b);
        }

        public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {
            return -super.compare(b1, s1, l1, b2, s2, l2);
        }
    }

    @Override
    public int run(String[] args) throws Exception {

        // 实例化作业对象，设置作业名称
        Job job = new Job(getConf(), "Q9EmpSalarySort");
        job.setJobName("Q9EmpSalarySort");
```

```
        // 设置Mapper和Reduce类
        job.setJarByClass(Q9EmpSalarySort.class);
        job.setMapperClass(MapClass.class);

        // 设置输出格式类
        job.setMapOutputKeyClass(IntWritable.class);
        job.setMapOutputValueClass(Text.class);
        job.setSortComparatorClass(DecreaseComparator.class);

        // 第1个参数为员工数据路径和第2个参数为输出路径
        String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
        args).getRemainingArgs();
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q9EmpSalarySort(), args);
        System.exit(res);
    }
}
```

3.9.4 编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6 目录中新建 Q9EmpSalarySort.java 程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q9EmpSalarySort.java 文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q9EmpSalarySort.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q9EmpSalarySort.java

把编译好的代码打成 jar 包，如果不打成 jar 形式运行会提示 class 无法找到的错误

jar cvf ./Q9EmpSalarySort.jar ./Q9EmpSalary.class*

*mv *.jar ../..*

rm Q9EmpSalary.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q9EmpSalarySort.java
[shiyanlou@b393a04554e1 class6]$ ll
total 28
-rw-r--r-- 1 shiyanlou shiyanlou 3292 Jun  6 13:46 Q6HigherThanAveSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3098 Jun  6 13:55 Q7NameDeptOfStartJ.java
-rw-r--r-- 1 shiyanlou shiyanlou 3633 Jun  6 14:10 Q8SalaryTop3Salary.java
-rw-rw-r-- 1 shiyanlou shiyanlou 2114 Jun  6 14:17 Q9EmpSalarySort.class
-rw-rw-r-- 1 shiyanlou shiyanlou  613 Jun  6 14:17 Q9EmpSalarySort$DecreaseComparator.clas
s
-rw-r--r-- 1 shiyanlou shiyanlou 2306 Jun  6 14:17 Q9EmpSalarySort.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1806 Jun  6 14:17 Q9EmpSalarySort$MapClass.class
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q9EmpSalarySort.jar ./Q9EmpSalary*.class
added manifest
adding: Q9EmpSalarySort.class(in = 2114) (out= 1021)(deflated 51%)
adding: Q9EmpSalarySort$DecreaseComparator.class(in = 613) (out= 345)(deflated 43%)
adding: Q9EmpSalarySort$MapClass.class(in = 1806) (out= 733)(deflated 59%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q9EmpSalary*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar            hadoop-test-1.1.2.jar   NOTICE.txt
bin                           hadoop-tools-1.1.2.jar  Q5EarnMoreThanManager.jar
build.xml                     hdfs                    Q6HigherThanAveSalary.jar
c++                           input                   Q7NameDeptOfStartJ.jar
CHANGES.txt                   ivy                     Q8SalaryTop3Salary.jar
conf                          ivy.xml                 Q9EmpSalarySort.jar
contrib                       lib                     README.txt
hadoop-ant-1.1.2.jar          libexec                 sbin
```

3.9.5 运行并查看结果

运行 Q9EmpSalarySort 运行的员工数据路径和输出路径两个参数，需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out9

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q9EmpSalarySort.jar Q9EmpSalarySort

hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out9

```
[shiyanlou@b393a04554e1 class6]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q9EmpSalarySort.jar Q9EmpSalarySort hdfs
://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out9
15/06/06 14:18:38 INFO input.FileInputFormat: Total input paths to process : 1
15/06/06 14:18:38 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/06 14:18:38 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/06 14:18:38 INFO mapred.JobClient: Running job: job_201506040132_0012
15/06/06 14:18:39 INFO mapred.JobClient:  map 0% reduce 0%
15/06/06 14:18:44 INFO mapred.JobClient:  map 100% reduce 0%
15/06/06 14:18:52 INFO mapred.JobClient:  map 100% reduce 33%
15/06/06 14:18:53 INFO mapred.JobClient:  map 100% reduce 100%
15/06/06 14:18:55 INFO mapred.JobClient: Job complete: job_201506040132_0012
15/06/06 14:18:55 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out9 目录

hadoop fs -ls /class6/out9

hadoop fs -cat /class6/out9/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

5000 KING

3000 FORD

2975 JONES

2850 BLAKE

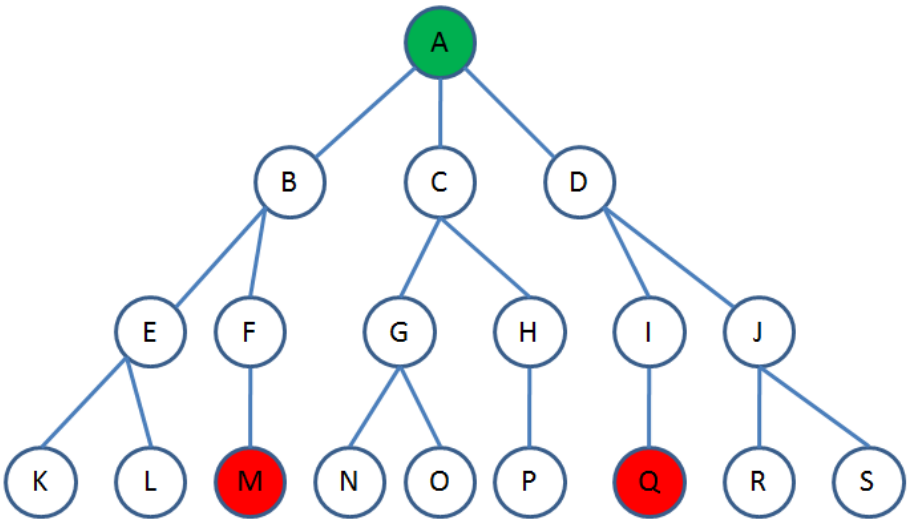
……

```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out9
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-06 14:18 /class6/out9/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-06 14:18 /class6/out9/_logs
-rw-r--r--   1 shiyanlou supergroup        130 2015-06-06 14:18 /class6/out9/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out9/part-r-00000
5000    KING
3000    FORD
2975    JONES
2850    BLAKE
2650    MARTIN
2450    CLARK
1900    ALLEN
1750    WARD
1500    TURNER
1300    MILLER
950     JAMES
800     SMITH
```

3.10 测试例子 10：求任何两名员工信息传递所需要经过的中间节点数

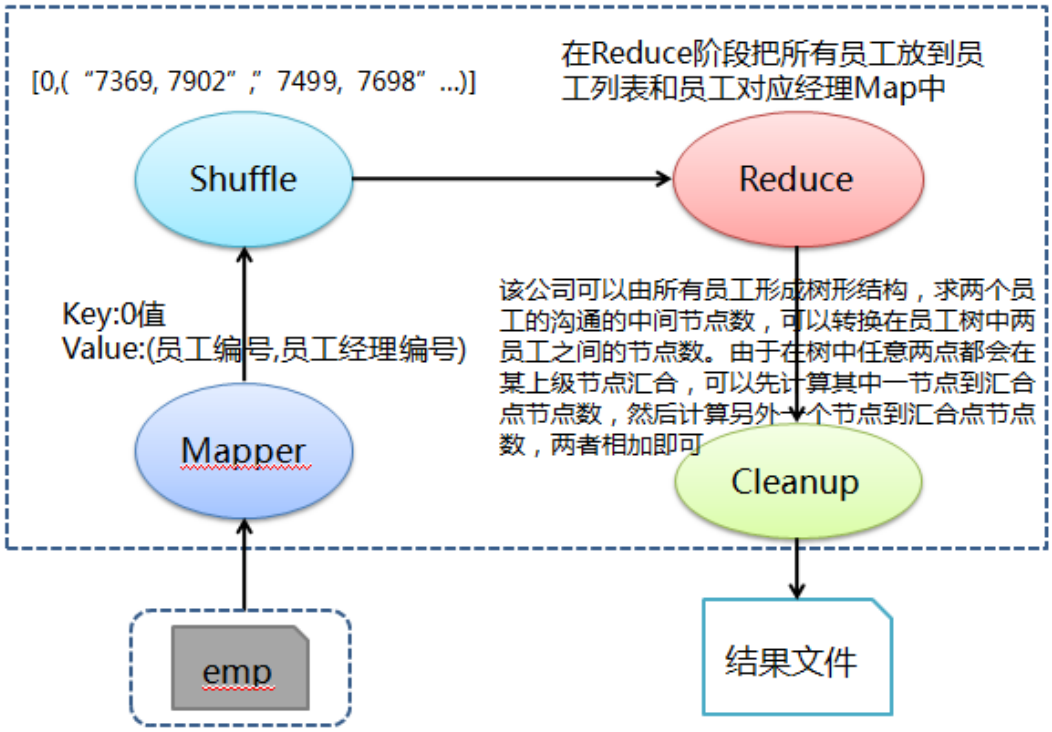
3.10.1问题分析

该公司所有员工可以形成入下图的树形结构，求两个员工的沟通的中间节点数，可转换在员工树中求两个节点连通所经过的节点数 ,即从其中一节点到汇合节点经过节点数加上另一节点到汇合节点经过节点数。例如求 M 到 Q 所需节点数，可以先找出 M 到 A 经过的节点数，然后找出 Q 到 A 经过的节点数，两者相加得到 M 到 Q 所需节点数。



在作业中首先在 Mapper 阶段所有员工数据，其中经理数据 key 为 0 值、value 为"员工编号，员工经理编号"，然后在 Reduce 阶段把所有员工放到员工列表和员工对应经理链表 Map 中，最后在 Reduce 的 Cleanup 中按照上面说所算法对任意两个员工计算出沟通的路径长度并输出。

3.10.2处理流程图



3.10.3编写代码

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class Q10MiddlePersonsCountForComm extends Configured implements Tool {
```

```
public static class MapClass extends Mapper<LongWritable, Text, IntWritable, Text> {

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {

        // 对员工文件字段进行拆分
        String[] kv = value.toString().split(",");

        // 输出key为0和value为员工编号+","+员工经理编号
        context.write(new IntWritable(0), new Text(kv[0] + "," + ("".equals(kv[3]) ? " " : kv[3])));
    }
}

public static class Reduce extends Reducer<IntWritable, Text, NullWritable, Text> {

    // 定义员工列表和员工对应经理Map
    List<String> employeeList = new ArrayList<String>();
    Map<String, String> employeeToManagerMap = new HashMap<String, String>();

    public void reduce(IntWritable key, Iterable<Text> values, Context context) throws
IOException, InterruptedException {

        // 在reduce阶段把所有员工放到员工列表和员工对应经理Map中
        for (Text value : values) {
            employeeList.add(value.toString().split(",")[0].trim());
            employeeToManagerMap.put(value.toString().split(",")[0].trim(),
            value.toString().split(",")[1].trim());
        }
    }

    @Override
    protected void cleanup(Context context) throws IOException, InterruptedException {
        int totalEmployee = employeeList.size();
        int i, j;
        int distance;
        System.out.println(employeeList);
        System.out.println(employeeToManagerMap);

        // 对任意两个员工计算出沟通的路径长度并输出
        for (i = 0; i < (totalEmployee - 1); i++) {
            for (j = (i + 1); j < totalEmployee; j++) {
                distance = calculateDistance(i, j);
                String value = employeeList.get(i) + " and " + employeeList.get(j) + " =
                " + distance;
                context.write(NullWritable.get(), new Text(value));
            }
        }
```

```
}

/**
 * 该公司可以由所有员工形成树形结构，求两个员工的沟通的中间节点数，可以转换在员工树中两员工之间的距离
 * 由于在树中任意两点都会在某上级节点汇合，根据该情况设计了如下算法
 */
private int calculateDistance(int i, int j) {
    String employeeA = employeeList.get(i);
    String employeeB = employeeList.get(j);
    int distance = 0;

    // 如果A是B的经理，反之亦然
    if (employeeToManagerMap.get(employeeA).equals(employeeB) ||
        employeeToManagerMap.get(employeeB).equals(employeeA)) {
        distance = 0;
    }
    // A和B在同一经理下
    else if  (employeeToManagerMap.get(employeeA).equals(
            employeeToManagerMap.get(employeeB))) {
        distance = 0;
    } else {
        // 定义A和B对应经理链表
        List<String> employeeA_ManagerList = new ArrayList<String>();
        List<String> employeeB_ManagerList = new ArrayList<String>();

        // 获取从A开始经理链表
        employeeA_ManagerList.add(employeeA);
        String current = employeeA;
        while (false == employeeToManagerMap.get(current).isEmpty()) {
            current = employeeToManagerMap.get(current);
            employeeA_ManagerList.add(current);
        }

        // 获取从B开始经理链表
        employeeB_ManagerList.add(employeeB);
        current = employeeB;
        while (false == employeeToManagerMap.get(current).isEmpty()) {
            current = employeeToManagerMap.get(current);
            employeeB_ManagerList.add(current);
        }

        int ii = 0, jj = 0;
        String currentA_manager, currentB_manager;
        boolean found = false;

        // 遍历A与B开始经理链表，找出汇合点计算
        for (ii = 0; ii < employeeA_ManagerList.size(); ii++) {
```

```
                currentA_manager = employeeA_ManagerList.get(ii);
                for (jj = 0; jj < employeeB_ManagerList.size(); jj++) {
                    currentB_manager = employeeB_ManagerList.get(jj);
                    if (currentA_manager.equals(currentB_manager)) {
                        found = true;
                        break;
                    }
                }

                if (found) {
                    break;
                }
            }

            // 最后获取两只之前的路径
            distance = ii + jj - 1;
        }

        return distance;
    }
}

@Override
public int run(String[] args) throws Exception {

    // 实例化作业对象，设置作业名称
    Job job = new Job(getConf(), "Q10MiddlePersonsCountForComm");
    job.setJobName("Q10MiddlePersonsCountForComm");

    // 设置Mapper和Reduce类
    job.setJarByClass(Q10MiddlePersonsCountForComm.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(Reduce.class);

    // 设置Mapper输出格式类
    job.setMapOutputKeyClass(IntWritable.class);
    job.setMapOutputValueClass(Text.class);

    // 设置Reduce输出键和值类型
    job.setOutputFormatClass(TextOutputFormat.class);
    job.setOutputKeyClass(NullWritable.class);
    job.setOutputValueClass(Text.class);

    // 第1个参数为员工数据路径和第2个参数为输出路径
    String[] otherArgs = new GenericOptionsParser(job.getConfiguration(),
    args).getRemainingArgs();
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
```

```
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        job.waitForCompletion(true);
        return job.isSuccessful() ? 0 : 1;
    }

    /**
     * 主方法，执行入口
     * @param args 输入参数
     */
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Q10MiddlePersonsCountForComm(), args);
        System.exit(res);
    }
}
```

3.10.4编译并打包代码

进入/app/hadoop-1.1.2/myclass/class6目录中新建Q10MiddlePersonsCountForComm.java程序代码（代码页可以使用/home/shiyanlou/install-pack/class6/Q10MiddlePersonsCountForComm.java文件）

cd /app/hadoop-1.1.2/myclass/class6

vi Q10MiddlePersonsCountForComm.java

编译代码

javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/commons-cli-1.2.jar Q10MiddlePersonsCountForComm.java

把编译好的代码打成jar包，如果不打成jar形式运行会提示class无法找到的错误

jar cvf ./Q10MiddlePersonsCountForComm.jar ./Q10MiddlePersons.class*

*mv *.jar ../..*

rm Q10MiddlePersons.class*

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2/myclass/class6
[shiyanlou@b393a04554e1 class6]$ javac -classpath ../../hadoop-core-1.1.2.jar:../../lib/co
mmons-cli-1.2.jar Q10MiddlePersonsCountForComm.java
[shiyanlou@b393a04554e1 class6]$ ll
total 40
-rw-rw-r-- 1 shiyanlou shiyanlou 2363 Jun  6 14:22 Q10MiddlePersonsCountForComm.class
-rw-r--r-- 1 shiyanlou shiyanlou 4969 Jun  6 14:21 Q10MiddlePersonsCountForComm.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1976 Jun  6 14:22 Q10MiddlePersonsCountForComm$MapClass.c
lass
-rw-rw-r-- 1 shiyanlou shiyanlou 4201 Jun  6 14:22 Q10MiddlePersonsCountForComm$Reduce.cla
ss
-rw-r--r-- 1 shiyanlou shiyanlou 3292 Jun  6 13:46 Q6HigherThanAveSalary.java
-rw-r--r-- 1 shiyanlou shiyanlou 3098 Jun  6 13:55 Q7NameDeptOfStartJ.java
-rw-r--r-- 1 shiyanlou shiyanlou 3633 Jun  6 14:10 Q8SalaryTop3Salary.java
-rw-r--r-- 1 shiyanlou shiyanlou 2306 Jun  6 14:17 Q9EmpSalarySort.java
[shiyanlou@b393a04554e1 class6]$
[shiyanlou@b393a04554e1 class6]$ jar cvf ./Q10MiddlePersonsCountForComm.jar ./Q10MiddlePer
sons*.class
added manifest
adding: Q10MiddlePersonsCountForComm.class(in = 2363) (out= 1109)(deflated 53%)
adding: Q10MiddlePersonsCountForComm$MapClass.class(in = 1976) (out= 791)(deflated 59%)
adding: Q10MiddlePersonsCountForComm$Reduce.class(in = 4201) (out= 1810)(deflated 56%)
[shiyanlou@b393a04554e1 class6]$ mv *.jar ../..
[shiyanlou@b393a04554e1 class6]$ rm Q10MiddlePersons*.class
[shiyanlou@b393a04554e1 class6]$ ls ../..
AvgTemperature.jar            hadoop-tools-1.1.2.jar          Q5EarnMoreThanManager.jar
bin                           hdfs                            Q6HigherThanAveSalary.jar
build.xml                     input                           Q7NameDeptOfStartJ.jar
c++                           ivy                             Q8SalaryTop3Salary.jar
CHANGES.txt                   ivy.xml                         Q9EmpSalarySort.jar
conf                          lib                             README.txt
contrib                       libexec                         sbin
hadoop-ant-1.1.2.jar          LICENSE.txt                     share
hadoop-client-1.1.2.jar       logs                            src
hadoop-core-1.1.2.jar         MinTemperature.jar              tmp
hadoop-examples-1.1.2.jar     myclass                         webapps
hadoop-minicluster-1.1.2.jar  NOTICE.txt
hadoop-test-1.1.2.jar         Q10MiddlePersonsCountForComm.jar
```

3.10.5运行并查看结果

运行 Q10MiddlePersonsCountForComm 运行的员工数据路径和输出路径两个参数 ,需要注意的是 hdfs 的路径参数路径需要全路径，否则运行会报错：

- 员工数据路径：hdfs://hadoop:9000/class6/input/emp
- 输出路径：hdfs://hadoop:9000/class6/out10

运行如下命令：

cd /app/hadoop-1.1.2

hadoop jar Q10MiddlePersonsCountForComm.jar Q10MiddlePersonsCountForComm hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out10

```
[shiyanlou@b393a04554e1 class6]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar Q10MiddlePersonsCountForComm.jar Q10Midd
lePersonsCountForComm hdfs://hadoop:9000/class6/input/emp hdfs://hadoop:9000/class6/out10
15/06/06 14:23:50 INFO input.FileInputFormat: Total input paths to process : 1
15/06/06 14:23:50 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/06 14:23:50 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/06 14:23:51 INFO mapred.JobClient: Running job: job_201506040132_0013
15/06/06 14:23:52 INFO mapred.JobClient:  map 0% reduce 0%
15/06/06 14:23:57 INFO mapred.JobClient:  map 100% reduce 0%
15/06/06 14:24:05 INFO mapred.JobClient:  map 100% reduce 33%
15/06/06 14:24:07 INFO mapred.JobClient:  map 100% reduce 100%
15/06/06 14:24:08 INFO mapred.JobClient: Job complete: job_201506040132_0013
15/06/06 14:24:08 INFO mapred.JobClient: Counters: 29
```

运行成功后，刷新 CentOS HDFS 中的输出路径/class6/out10 目录

hadoop fs -ls /class6/out10

hadoop fs -cat /class6/out10/part-r-00000

打开 part-r-00000 文件，可以看到运行结果：

7369 and 7499 = 4

7369 and 7521 = 4

7369 and 7566 = 1

7369 and 7654 = 4

7369 and 7698 = 3

……

```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class6/out10
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-06 14:24 /class6/out10/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-06 14:23 /class6/out10/_logs
-rw-r--r--   1 shiyanlou supergroup       1188 2015-06-06 14:24 /class6/out10/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class6/out10/part-r-00000
7369 and 7499 = 4
7369 and 7521 = 4
7369 and 7566 = 1
7369 and 7654 = 4
7369 and 7698 = 3
7369 and 7782 = 3
```