Lab 4 - Hadoop

Handout: May 3, 2018

Deadline: June 5 23:59, 2018 (No extension)

Assignment overview:

In this assignment you should build a Hadoop cluster and make two MapReduce applications and deploy them on your Hadoop cluster to get the result. Finally, you are required to validate the fault tolerant mechanisms of Hadoop Runtime Environment.

Build a Hadoop cluster:

You can refer to resources about how to build a Hadoop cluster on the Internet. Here, your Hadoop cluster should be based on Hadoop-2.x.x. Your cluster should contain 3 or more nodes (at least 1 master and 2 slaves). You're required to record the details of every step on your lab report. Make sure it's well structured. You can build your cluster on Virtual Machines, Dockers, or Cloud Servers.

MapReduce Application:

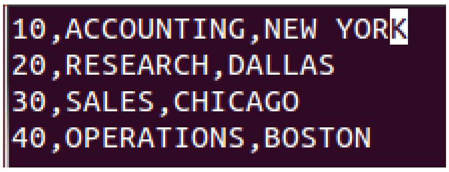
[Example]

Two tables are given as follows:

```
Department {                    Employee {
    id: int,                        id: int,
    name: string,                   name: string,
    location: string                salary: int,
}                                   department_id: int
                                }
```

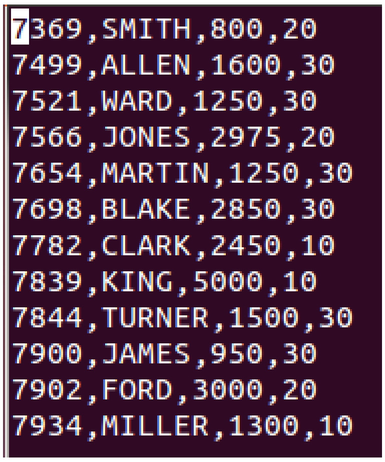
We export them from MYSQL to two files: dept and emp. Here are the context of them.

dept:



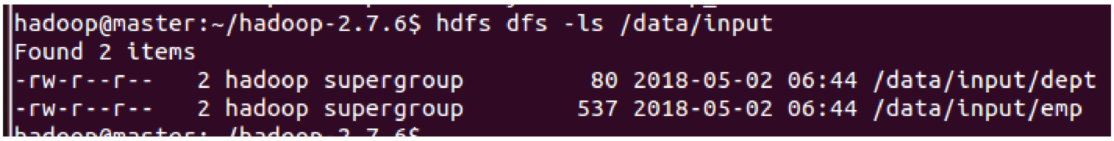
```
10,ACCOUNTING,NEW YORK
20,RESEARCH,DALLAS
30,SALES,CHICAGO
40,OPERATIONS,BOSTON
```

emp:



```
7369,SMITH,800,20
7499,ALLEN,1600,30
7521,WARD,1250,30
7566,JONES,2975,20
7654,MARTIN,1250,30
7698,BLAKE,2850,30
7782,CLARK,2450,10
7839,KING,5000,10
7844,TURNER,1500,30
7900,JAMES,950,30
7902,FORD,3000,20
7934,MILLER,1300,10
```

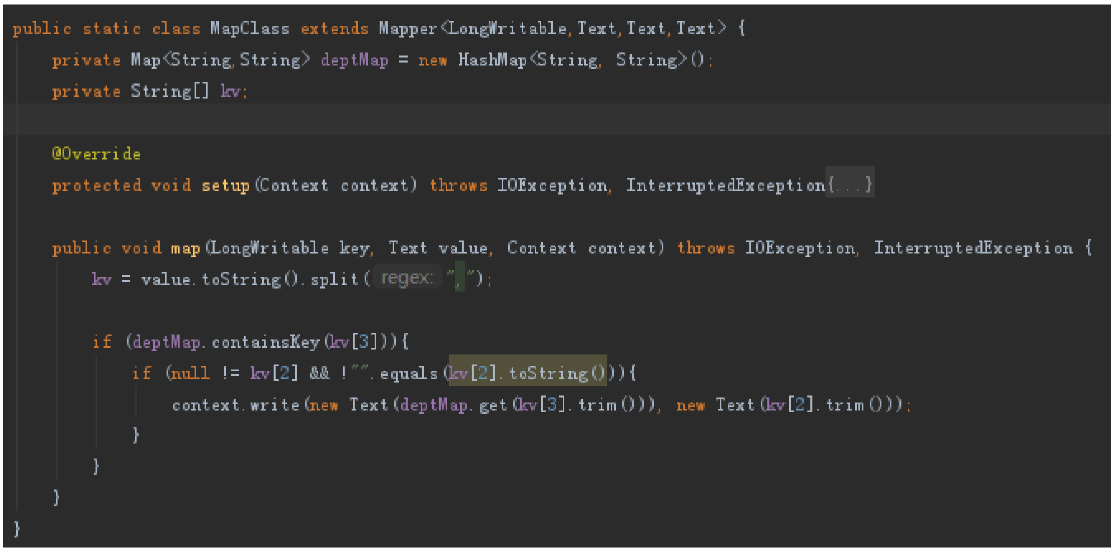
After uploading to HDFS, they are as follows:



```
hadoop@master:~/hadoop-2.7.6$ hdfs dfs -ls /data/input
Found 2 items
-rw-r--r--   2 hadoop supergroup         80 2018-05-02 06:44 /data/input/dept
-rw-r--r--   2 hadoop supergroup        537 2018-05-02 06:44 /data/input/emp
hadoop@master:~/hadoop-2.7.6$
```

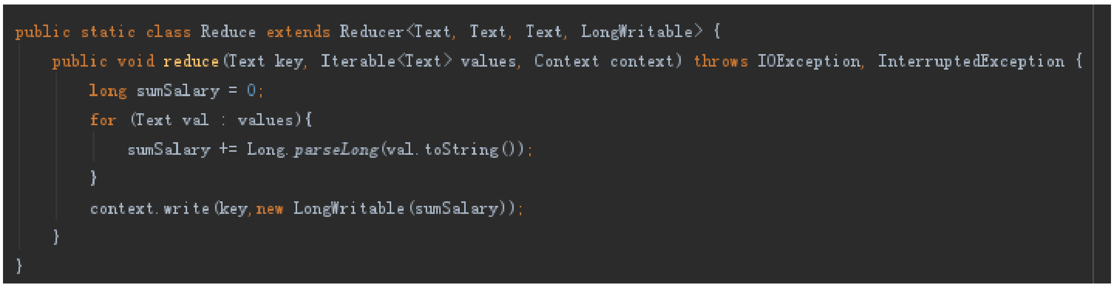
And then, we made a MapReduce to calculate the salary of every department.

Mapper:



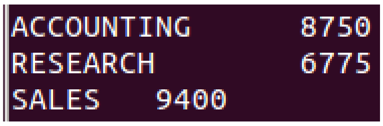
```java
public static class MapClass extends Mapper<LongWritable,Text,Text,Text> {
    private Map<String,String> deptMap = new HashMap<String, String>();
    private String[] kv;

    @Override
    protected void setup(Context context) throws IOException, InterruptedException{...}

    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
        kv = value.toString().split( regex: ",");

        if (deptMap.containsKey(kv[3])){
            if (null != kv[2] && !"".equals(kv[2].toString())){
                context.write(new Text(deptMap.get(kv[3].trim())), new Text(kv[2].trim()));
            }
        }
    }
}
```

Reducer:



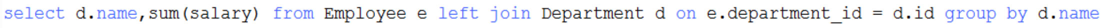
```java
public static class Reduce extends Reducer<Text, Text, Text, LongWritable> {
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
        long sumSalary = 0;
        for (Text val : values){
            sumSalary += Long.parseLong(val.toString());
        }
        context.write(key,new LongWritable(sumSalary));
    }
}
```

After deployed it on Hadoop cluster, we get the result:



```
ACCOUNTING      8750
RESEARCH        6775
SALES   9400
```

It’s same with the result of this SQL:



```sql
select d.name,sum(salary) from Employee e left join Department d on e.department_id = d.id group by d.name
```

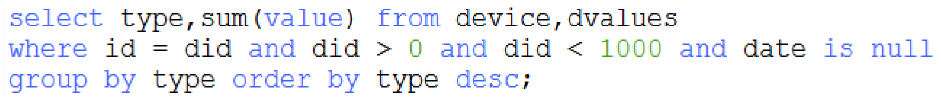
[Problem Description]

There are some IOT devices data stored in two tables:

Device{	**DValues {**
id: int,	**did: int,**
type: string,	**date: string,**
location: string	**value: double,**
}	**}**

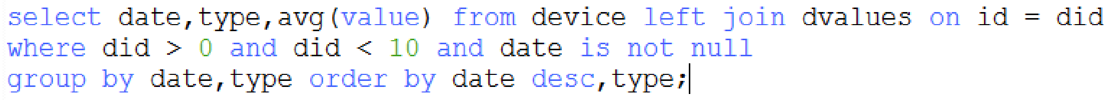
You need to make two MapReduce applications to get results of following two SQLs.

1.



```
select type,sum(value) from device,dvalues
where id = did and did > 0 and did < 1000 and date is null
group by type order by type desc;
```

2.



```
select date,type,avg(value) from device left join dvalues on id = did
where did > 0 and did < 10 and date is not null
group by date,type order by date desc,type;
```

Notes:

1. Test data are given in *.sql, you need to import table and export data to file.
2. You need to think about how to deal with null value in each stage.
3. You may need to do several Maps and Reduces.
4. You can download the Test data from https://pan.baidu.com/

[Validate the fault tolerant mechanisms]:

In this section, you need to validate the fault tolerant mechanisms of distributed computing. Surely, you've experienced using Hadoop cluster to run MapReduce in the above section. Now, considering what will happen if one node fails to execute a task during the execution of MapReduce jobs. You need to find a way to kill some sub-tasks of a MapReduce job, and try to know how Hadoop Runtime Environment deal with the failures. You can look for YARN's log to get some information. Please record exploration process and some critical logs in your lab reports.

[Lab report]:

Your lab report will include three parts. First, a document describing your Hadoop cluster building steps and your design of the MapReduce applications and your verification of the fault tolerant mechanisms. Second, your source code of the MapReduce applications. Third, the output text files of your MapReduce applications.

Upload your lab report as a gzipped tar file and name it as {Your student ID}.tar.gz to ftp://liaoruohuai:public@public.sjtu.edu.cn/upload/lab4

[Grading]:

60%: the quality of lab report.

40%: laboratory examinations, scheduled to July.