

实验名称

计算HBase上的学生成绩写入HDFS

实验目的

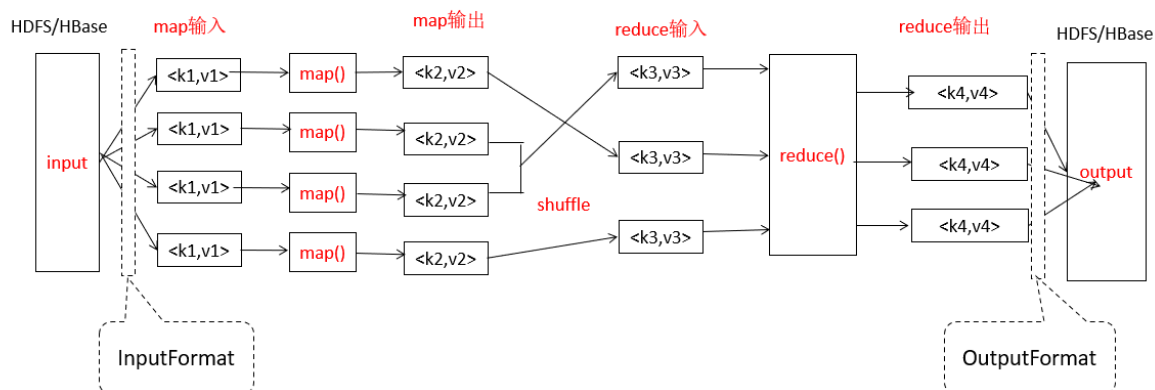
熟练掌握MapReduce的编程模型

熟练开发基于HBase和HDFS的MapReduce程序

实验背景

对于海量的离线数据的处理，我们一般采用的是MapReduce来进行计算。而数据一般存储在HDFS或HBase表中，如何来完成HDFS与HBase表中数据的计算呢？本节实验将HBase表中的数据通过MapReduce编程计算写入至HDFS。

实验原理



从MapReduce自身的命名特点可以看出，MapReduce由两个阶段组成：Map和Reduce。用户只需要编写map()和reduce()两个函数，即可完成分布式程序的设计。

实验环境

ubuntu 16.04

hadoop 2.7.3

jdk 1.8

hbase 1.2.6

maven 3.6.1

实验课时

1课时

实验步骤

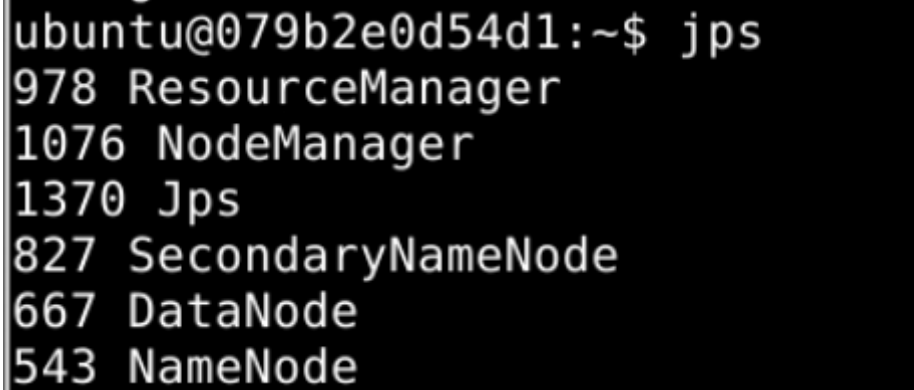
一、环境准备

本实验在eclipse进行开发。

首先启动Hadoop环境：

```
start-all.sh  
jps
```

当看下以下进程，则Hadoop启动成功



```
ubuntu@079b2e0d54d1:~$ jps  
978 ResourceManager  
1076 NodeManager  
1370 Jps  
827 SecondaryNameNode  
667 DataNode  
543 NameNode
```

启动HBase

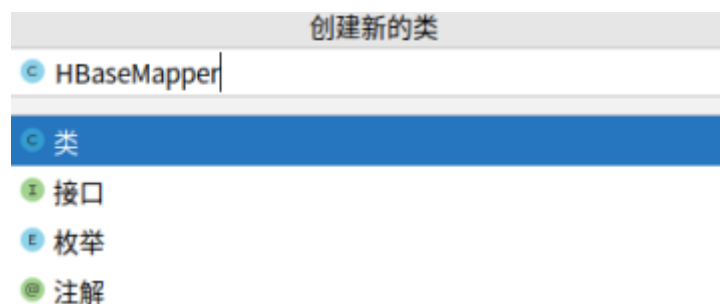
另打开一个终端

```
start-hbase.sh
```

二、代码编写

3、计算HBase表中的数据，将结果写入HDFS

首先创建HBaseMapper类，继承TableMapper，然后重写map()方法。



完整代码如下：

```
import java.io.IOException;  
  
import org.apache.hadoop.hbase.client.Result;  
import org.apache.hadoop.hbase.io.ImmutableBytesWritable;  
import org.apache.hadoop.hbase.mapreduce.TableMapper;  
import org.apache.hadoop.hbase.util.Bytes;  
import org.apache.hadoop.io.IntWritable;
```

```

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class HBaseMapper extends TableMapper<Text, IntWritable> {

    @Override
    protected void map(ImmutableBytesWritable key, Result value,
                      Mapper<ImmutableBytesWritable, Result, Text,
IntWritable>.Context context)
        throws IOException, InterruptedException {

        //获取班级信息
        String clas = Bytes.toString(value.getValue(Bytes.toBytes("info"),
Bytes.toBytes("clas")));
        //获取课程分数数据
        String course = Bytes.toString(value.getValue(Bytes.toBytes("info"),
Bytes.toBytes("course")));
        //班级为Key值，课程分数数据为Value值
        context.write(new Text(clas), new
IntWritable(Integer.parseInt(course)));
    }

}

```

接着创建HBaseReducer类，继承Reducer，过程请参考上节实验

接着重写reduce()方法

完整代码如下：

```

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class HBaseReduce extends Reducer<Text, IntWritable, Text, NullWritable>
{

    @Override
    protected void reduce(Text k3, Iterable<IntWritable> v3,
                      Reducer<Text, IntWritable, Text, NullWritable>.Context
context) throws IOException, InterruptedException {

        int count = 0; //定义同学人数变量
        int sum = 0; //定义总分数

        //统计总人数与总分数
        for (IntWritable v : v3) {
            count += v.get();
            sum++;
        }
        //计算平均分数
        int avg = count / sum;
        //格式化输出
    }

}

```

```

        String result = k3.toString() + "共" + sum + "人;总分数是: " + count + "; 平均分是: " + avg;

        //将reduce结果输出到文件
        context.write(new Text(result), NullWritable.get());

    }
}

```

最后创建HBaseMain类，完成main()代码的编写

完成代码如下：

```

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.hbase.client.Scan;
import org.apache.hadoop.hbase.mapreduce.TableMapReduceUtil;
import org.apache.hadoop.hbase.util.Bytes;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class HBaseMain {
    private static final String ZK_QUORUM = "localhost";    //设置ZK的IP
    private static final String INPUT_TABLE_NAME = "mytable";    //获取数据的表格名称
    public static void main(String[] args) throws Exception {
        //hadoop设置
        Configuration conf = new Configuration();
        conf.set("hbase.zookeeper.quorum", ZK_QUORUM);

        //创建任务，并配置运行驱动类
        Job job = Job.getInstance(conf);
        job.setJarByClass(HBaseMain.class);

        //定义scan对象
        Scan scan = new Scan();
        //甚至获取对应的列信息
        scan.addColumn(Bytes.toBytes("info"), Bytes.toBytes("clas"));
        scan.addColumn(Bytes.toBytes("info"), Bytes.toBytes("course"));

        //配置Mapper端，对接HBase数据
        TableMapReduceUtil.initTableMapperJob(INPUT_TABLE_NAME, scan,
        HBaseMapper.class,
            Text.class, IntWritable.class, job);

        //配置reduce端运行方法
        job.setReducerClass(HBaseReducer.class);
        //reduce端输出格式设置
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(NullWritable.class);
        //配置输出路径
        FileOutputFormat.setOutputPath(job, new
        Path("hdfs://localhost:9000/001/output"));
    }
}

```

```

        job.waitForCompletion(true);

    }

}

```

4、执行程序并查看结果

```

7  import org.apache.hadoop.io.NullWritable;
8  import org.apache.hadoop.io.Text;
9  import org.apache.hadoop.mapreduce.Job;
10 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
11
12 1个用法
13 public class HBaseMain {
14     1个用法
15     private static final String ZK_QUORUM = "localhost"; //设置ZK的IP
16     1个用法
17     private static final String INPUT_TABLE_NAME = "mytable"; //获取数据的表格名称
18     public static void main(String[] args) throws Exception {
19         //hadoop设置
20         Configuration conf = new Configuration();
21         conf.set("hbase.zookeeper.quorum", ZK_QUORUM);
22
23         //创建任务，并配置运行驱动类
24         Job job = Job.getInstance(conf);
25         job.setJarByClass(HBaseMain.class);
26
27         //定义scan对象
28         Scan scan = new Scan();
29         //甚至获取对应的列信息
30         scan.addColumn(Bytes.toBytes("info"), Bytes.toBytes("clas"));
31         scan.addColumn(Bytes.toBytes("info"), Bytes.toBytes("course"));
32
33         //配置Mapper端，对接HBase数据

```

打开命令行窗口，使用hadoop shell命令查看结果

```

ubuntu@210975724be0:~$ hdfs dfs -ls /output
Found 2 items
-rw-r--r--  3 ubuntu supergroup          0 2022-02-18 02:53 /output/_SUCCESS
-rw-r--r--  3 ubuntu supergroup    106 2022-02-18 02:53 /output/part-r-00000
ubuntu@210975724be0:~$ hdfs dfs -cat /output/part-r-00000
16-1共81人;总分数是:6271;平均分是:77
17-2共54人;总分数是:4134;平均分是:76

```

到此实验结束。

实验总结

该实验主要是练习基于HDFS和HBase的MapReduce的编写，其中，源数据存储在HBase上，故在编写HBaseMapper方法时，需要继承的是TableMapper类，即重写的map()是TableMapper类中的方法，我们最终将结果写入HDFS上，即处理完成的reduce()结果最终由outputFormat类写入HDFS的文件中，故在编写HBaseReducer类时，需要继承的是Reducer类，即重写的reduce()是Reducer类中的方法，请结合实验2和实验3理解基于HDFS和HBase的MapReduce的代码编写。