

实验四 HBase 与 Hive

一、 小组信息

集群账号: 2017st15

组长 141220138 臧凤云 2714359840@qq.com

组员 141220132 银琦 141220132@smail.nju.edu.cn

二、 实验环境

Ubuntu 12.04 Java1.8.0_121 Maven3.3.9 IntelliJ 2017.1.2

HBase 1.2.5 Hive2.1.1

三、 HBase 安装与运行

1. 将压缩包拷贝至安装文件夹并解压

2. 修改 hbase-1.2.5/conf/hbase-env.sh

```
26 # The java implementation to use. Java 1.7+ required.
27 # export JAVA_HOME=/usr/java/jdk1.6.0/
28 export JAVA_HOME=/usr/local/java/jdk1.7.0_25
```

```
128 # Tell HBase whether it should manage it's own instance of Zookeeper or not.
129 export HBASE_MANAGES_ZK=true
```

3. 修改 hbase-1.2.5/conf/hbase-site.xml

```
23 <configuration>
24   <property>
25     <name>hbase.rootdir</name>
26     <value>hdfs://localhost:9000/hbase</value>
27   </property>
28
29   <property>
30     <name>hbase.cluster.distributed</name>
31     <value>true</value>
32   </property>
33 </configuration>
```

4. 修改 hadoop-2.7.1/etc/hadoop/hadoop-env.sh

```
24 # The java implementation to use.
25 export JAVA_HOME=${JAVA_HOME}
26 export JAVA_HOME=/usr/local/java/jdk1.7.0_25
27 #export HBASE_HOME=${HBASE_HOME}
28 #export HADOOP_CLASSPATH=${HBASE_HOME}/lib/*:$HADOOP_CLASSPATH
29 export HADOOP_HOME=/usr/local/hadoop/hadoop-2.7.1
30 export PATH=$PATH:/usr/local/hadoop/hadoop-2.7.1/bin
31
32 HBASE_LIBS=/usr/local/hadoop/hbase-1.2.5/lib/*
33 export HADOOP_CLASSPATH=${HADOOP_CLASSPATH}:${HBASE_LIBS}
```

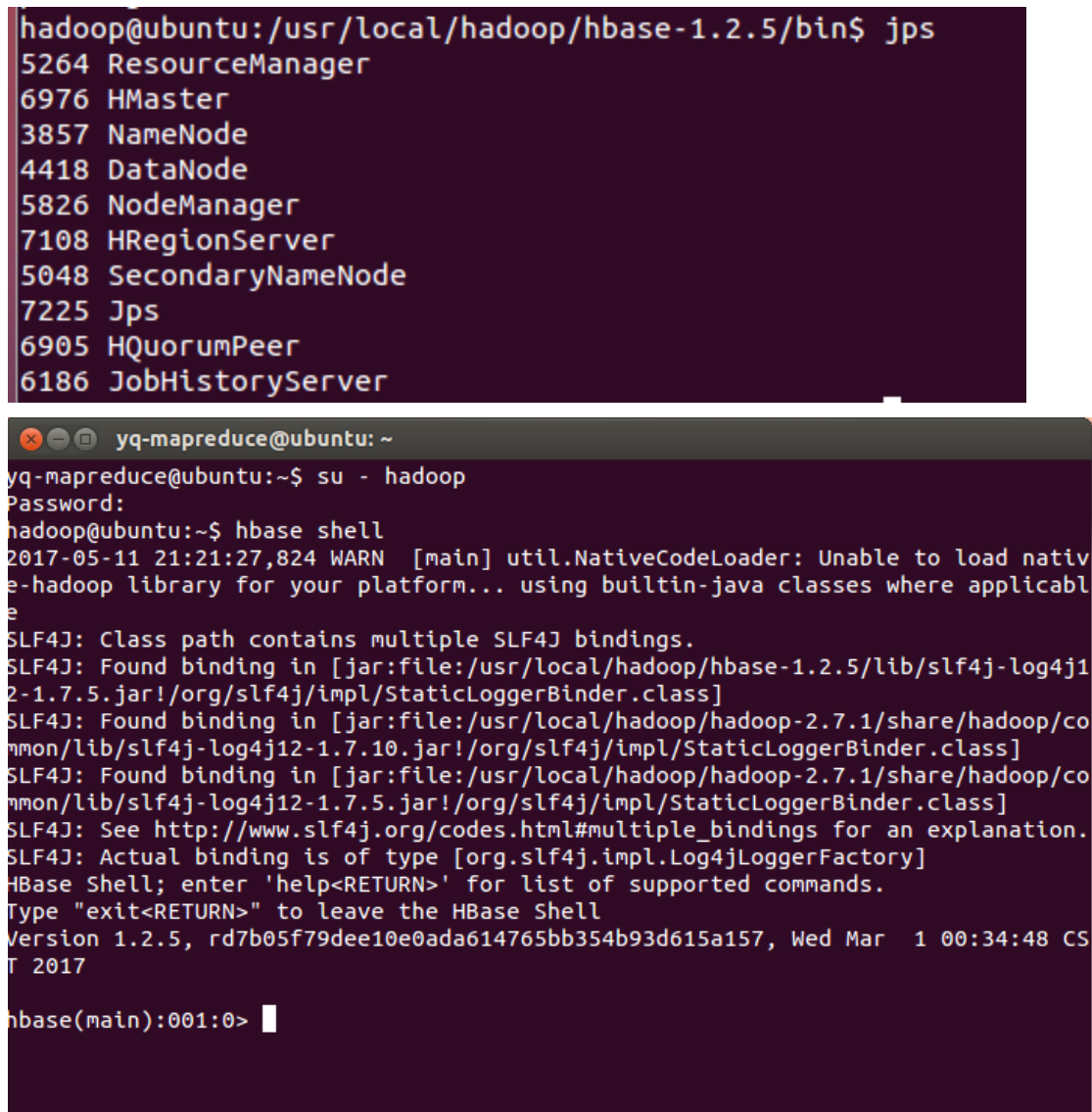
5. 替换 hbase 中 hadoop 的 jar 包，进行版本统一：

命令：find /usr/local/hadoop/hadoop-2.7.1/share/hadoop -name "hadoop*.jar" | xargs -i cp {} /usr/local/hadoop/hbase-1.2.5/lib/

6. 将 hbase-1.2.5/lib/下的 jar 包拷贝至 hadoop-2.7.1/share/Hadoop/common/lib/下

命令：cp /usr/local/hadoop/hbase-1.2.5/lib/*
/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/

7. 启动 hadoop, hbase



```
hadoop@ubuntu: /usr/local/hadoop/hbase-1.2.5/bin$ jps
5264 ResourceManager
6976 HMaster
3857 NameNode
4418 DataNode
5826 NodeManager
7108 HRegionServer
5048 SecondaryNameNode
7225 Jps
6905 HQuorumPeer
6186 JobHistoryServer

yq-mapreduce@ubuntu: ~
yq-mapreduce@ubuntu:~$ su - hadoop
Password:
hadoop@ubuntu:~$ hbase shell
2017-05-11 21:21:27,824 WARN [main] util.NativeCodeLoader: Unable to load native hbase-hadoop library for your platform... using builtin-java classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hbase-1.2.5/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.5, rd7b05f79dee10e0ada614765bb354b93d615a157, Wed Mar 1 00:34:48 CST 2017

hbase(main):001:0>
```

四、 在 Hbase 中建表

在代码 HBaseWuxia.java 中实现了每次运行前建表，所以不需要手动建表。

五、 修改 mapReduce 程序并填表

1. 登录集群，将文件下载至本地

```
hadoop fs -get /data/wuxia_novels/ wuxia
```

2. 将文件复制到 HDFS

```
hadoop fs -copyFromLocal /home/Hadoop/wuxia /
```

3. 代码的修改（仅展示与上次代码不同之处）

建立与 HBase 的连接

```
protected void setup(Context context) throws IOException, InterruptedException{
    con = ConnectionFactory.createConnection(configuration);
    table = con.getTable(TableName.valueOf("Wuxia"));
}
```

在 HBase 中删除已有的表并建立新表

```
public static void createHBaseTable(String tableName) throws IOException{
    TableName name= TableName.valueOf(tableName.getBytes());
    HTableDescriptor tableDescriptor=new HTableDescriptor(tableName);
    HColumnDescriptor columnDescriptor=new HColumnDescriptor( familyName: "average");
    tableDescriptor.addFamily(columnDescriptor);
    //Configuration conf=HBaseConfiguration.create();
    configuration.set("hbase.zookeeper.quorum", "localhost");
    Connection connection = ConnectionFactory.createConnection(configuration);
    Admin admin= connection.getAdmin();
    if(admin.tableExists(name)){
        // System.out.println("表已存在, 正在尝试重新创建表!");
        admin.disableTable(name);
        admin.deleteTable(name);
    }
    // System.out.println("创建新表: "+tableName);
    admin.createTable(tableDescriptor);
    connection.close();
}
```

使用 put 写入表

```
Put put = new Put(Bytes.toBytes(key.toString()));
put.add(Bytes.toBytes( $ "average"), Bytes.toBytes( $ "average for each word"), Bytes.toBytes(avgf.toString()));
//putList.add(put);
table.put(put);
```

断开连接

```
protected void cleanup(Context context) throws IOException, InterruptedException{
    table.close();
    con.close();
}
```

4. 任务的执行

```
Hadoop jar HBaseWuxia.jar HBaseWuxia /wuxia /wuxia_out
```

5. 执行结果

mapreduce 后的结果

```
educe@ubuntu:~$  
17/05/11 19:06:12 INFO mapreduce.Job: map 100% reduce 94%  
17/05/11 19:06:15 INFO mapreduce.Job: map 100% reduce 95%  
17/05/11 19:06:18 INFO mapreduce.Job: map 100% reduce 96%  
17/05/11 19:06:21 INFO mapreduce.Job: map 100% reduce 97%  
17/05/11 19:06:24 INFO mapreduce.Job: map 100% reduce 98%  
17/05/11 19:06:27 INFO mapreduce.Job: map 100% reduce 99%  
17/05/11 19:06:30 INFO mapreduce.Job: map 100% reduce 100%  
17/05/11 19:06:33 INFO mapreduce.Job: Job job_1494499495005_0001 completed successfully  
17/05/11 19:06:33 INFO mapreduce.Job: Counters: 49  
    File System Counters  
        FILE: Number of bytes read=141494240  
        FILE: Number of bytes written=308336450  
        FILE: Number of read operations=0  
        FILE: Number of large read operations=0  
        FILE: Number of write operations=0  
        HDFS: Number of bytes read=268308546  
        HDFS: Number of bytes written=104956627  
        HDFS: Number of read operations=657  
        HDFS: Number of large read operations=0  
        HDFS: Number of write operations=2  
    Job Counters  
        Launched map tasks=218  
        Launched reduce tasks=1  
        Data-local map tasks=218  
        Total time spent by all maps in occupied slots (ms)=4853753  
        Total time spent by all reduces in occupied slots (ms)=857375  
        Total time spent by all map tasks (ms)=4853753  
        Total time spent by all reduce tasks (ms)=857375  
        Total vcore-seconds taken by all map tasks=4853753  
        Total vcore-seconds taken by all reduce tasks=857375  
        Total megabyte-seconds taken by all map tasks=4970243072  
        Total megabyte-seconds taken by all reduce tasks=877952000  
    Map-Reduce Framework  
        Map input records=1954746  
        Map output records=45567096  
        Map output bytes=1438646177  
        Map output materialized bytes=141495536  
        Input split bytes=28643  
        Combine input records=45567096  
        Combine output records=7077262
```

检查表 Wuxia 中的内容，从表中信息可以看出共有 134882 行。

```
\xE9\xBE\x99\xE9\xA3\x9E\xE5\x87\xA4 column=average:average for each word, timestamp=1494493239307, value=1.33
\xE8\x88\x9E
\xE9\xBE\x99\xE9\xA3\x9E\xE8\x99\x8E column=average:average for each word, timestamp=1494493239308, value=1.00
\xE9\xBE\x99\xE9\xA6\x96 column=average:average for each word, timestamp=1494493239308, value=1.40
\xE9\xBE\x99\xE9\xA9\xAC column=average:average for each word, timestamp=1494493239309, value=5.09
\xE9\xBE\x99\xE9\xA9\xAC\xE7\xB2\xBE column=average:average for each word, timestamp=1494493239310, value=1.20
\xE7\xA5\x9E
\xE9\xBE\x99\xE9\xA9\xAD\xE4\xB8\x8A column=average:average for each word, timestamp=1494493239310, value=1.00
\xE5\xAE\xBE
\xE9\xBE\x99\xE9\xA9\xB9 column=average:average for each word, timestamp=1494493239311, value=10.04
\xE9\xBE\x99\xE9\xAA\x91\xE5\x85\xB5 column=average:average for each word, timestamp=1494493239311, value=6.67
\xE9\xBE\x99\xE9\xAA\x91\xE5\xA3\xAB column=average:average for each word, timestamp=1494493239312, value=1.00
\xE9\xBE\x99\xE9\xAA\xA8 column=average:average for each word, timestamp=1494493239314, value=1.29
\xE9\xBE\x99\xE9\xBD\xBF column=average:average for each word, timestamp=1494493239314, value=60.00
\xE9\xBE\x99\xE9\xBE\x99 column=average:average for each word, timestamp=1494493239315, value=3.25
\xE9\xBE\x9A column=average:average for each word, timestamp=1494493239316, value=21.43
\xE9\xBE\x9A\xE8\x87\xAA\xE7\x8F\x8D column=average:average for each word, timestamp=1494493239316, value=5.00
\xE9\xBE\x9B column=average:average for each word, timestamp=1494493239317, value=1.75
\xE9\xBE\x9F column=average:average for each word, timestamp=1494493239318, value=16.25
\xE9\xBE\x9F\xE5\x85\xB9 column=average:average for each word, timestamp=1494493239318, value=72.00
\xE9\xBE\x9F\xE5\x8D\x9C column=average:average for each word, timestamp=1494493239319, value=2.00
\xE9\xBE\x9F\xE5\xA3\xB3 column=average:average for each word, timestamp=1494493239320, value=2.15
\xE9\xBE\x9F\xE5\xA3\xB3\xE8\x8A\xB1 column=average:average for each word, timestamp=1494493239320, value=1.00
\xE9\xBE\x9F\xE5\xA4\xB4 column=average:average for each word, timestamp=1494493239321, value=2.33
\xE9\xBE\x9F\xE5\xB1\xB1 column=average:average for each word, timestamp=1494493239321, value=5.71
\xE9\xBE\x9F\xE7\x8E\x8B column=average:average for each word, timestamp=1494493239322, value=2.00
\xE9\xBE\x9F\xE7\x94\xB2 column=average:average for each word, timestamp=1494493239323, value=2.33
\xE9\xBE\x9F\xE7\x8A\xB9 column=average:average for each word, timestamp=1494493239323, value=3.00
\xE9\xBE\x9F\xE7\x8C\xA9 column=average:average for each word, timestamp=1494493239324, value=1.44
\xE9\xBE\x9F\xE8\x82\x89 column=average:average for each word, timestamp=1494493239325, value=1.33
\xE9\xBE\x9F\xE8\x83\x8C column=average:average for each word, timestamp=1494493239326, value=2.50
\xE9\xBE\x9F\xE8\xA3\x82 column=average:average for each word, timestamp=1494493239326, value=1.29
\xE9\xBE\x9F\xE9\xB3\x96 column=average:average for each word, timestamp=1494493239327, value=2.00
\xE9\xBE\x9F\xE9\xB9\xA4\xE9\x81\x90 column=average:average for each word, timestamp=1494493239327, value=1.00
\xE9\xBE\x84
\xEF\xA8\x8C column=average:average for each word, timestamp=1494493239328, value=5.00
\xEF\xBF\xA1 column=average:average for each word, timestamp=1494493239329, value=1.50
\xEF\xBF\xA5 column=average:average for each word, timestamp=1494493239346, value=3.33
134882 row(s) in 57.2660 seconds
hbase(main):002:0> █
```

六、 将表格内容保存至本地文件

1. 在 IntelliJ Idea 中运行 HbaseToFile.java, 将 hbase 中表 Wuxia 的内容写进 HBaseOut.txt 文件。
2. 代码:
与停词表写到本地共用一个文件, 用 Scan 不断从 HBase 的指定表中获取结果写入指定文件。

```
public static void main(String[] args){
    try {
        String filename = "/home/yq-mapreduce/Desktop/HBaseOut.txt";
        String tablename = "Wuxia";
        //String filename = "/home/yq-mapreduce/Desktop/WithStopWordsOut.txt";
        //String tablename = "WuxiaStopWords";
        PrintWriter output = new PrintWriter(new File(filename));
        Scan scan = new Scan();
        ResultScanner rs=null;
        HTable table = new HTable(conf, tablename);
        try {
            rs = table.getScanner(scan);
            for (Result r : rs) {
                for (KeyValue kv : r.list()) {
                    output.printf("%s\t%s\r\n", Bytes.toString(kv.getRow()), Bytes.toString(kv.getValue()));
                    output.flush();
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            rs.close();
        }
        output.close();
    } catch (IOException e){
        e.printStackTrace();
    }
}
```

Conf 配置

```
static Configuration conf = HBaseConfiguration.create();
static{
    conf.set("hbase.zookeeper.quorum", "localhost");
}
```

3. 结果:

展示了文件的结尾部分, 可以看出文件中词的总数与 HBase 表中的一致。

```
yq-mapreduce@ubuntu: ~
134860 龙龙      3.25
134861 龚      21.43
134862 龚自珍   5.00
134863 龔      1.75
134864 龟      16.25
134865 龟兹     72.00
134866 龟卜      2.00
134867 龟壳      2.15
134868 龟壳花    1.00
134869 龟头      2.33
134870 龟山      5.71
134871 龟王      2.00
134872 龟甲      2.33
134873 龟纹      3.00
134874 龟缩      1.44
134875 龟肉      1.33
134876 龟背      2.50
134877 龟裂      1.29
134878 龟鳖      2.00
134879 龟鹤遐龄   1.00
134880 兀       5.00
134881 𪚩       1.50
134882 𪚩       3.33

134882,1 Bot
```

七、 安装 Hive

1. 将压缩包拷贝至安装目录下并解压。
2. 配置系统环境变量 ~/.bash_profile

```
5 export HIVE_HOME=/usr/local/hadoop/hive-2.1.1
6 export PATH=$JAVA_HOME/bin:$HADOOP_HOME/bin:$HBASE_HOME/bin:$HIVE_HOME/bin:$
HIVE_HOME/conf:$PATH
```

3. 使环境变量生效: source ~/.bash_profile
4. 修改 hive-site.xml: cp hive-default.xml.template hive-site.xml
在模板的基础上作了四个修改:

(1) 设置 Hive 的存储目录

```
325 <property>
326   <name>hive.metastore.warehouse.dir</name>
327   <value>/usr/local/hadoop/hive-2.1.1/warehouse</value>
328   <description>location of default database for the warehouse</descriptio
n>
329 </property>
```


(2) JDBC 连接字符串

```
499 <property>
500   <name>javax.jdo.option.ConnectionURL</name>
501   <value>jdbc:derby:/usr/local/hadoop/hive-2.1.1/bin/metastore_db;databaseName=metastore_db;create=true</value>
502   <description>
503     JDBC connect string for a JDBC metastore.
504     To use SSL to encrypt/authenticate the connection, provide database-specific SSL flag in the connection URL.
505     For example, jdbc:postgresql://myhost/db?ssl=true for postgres database.
506   </description>
507 </property>
```

(3) hive 源码会检查导出的路径是否存在, 如果不存在则报错, 修改下列配置就不会报错。

```
3434 <property>
3435   <name>hive.insert.into.multilevel.dirs</name>
3436   <value>true</value>
3437   <description>
3438     Where to insert into multilevel directories like
3439     "insert directory '/HIVEFT25686/chinna/' from table"
3440   </description>
3441 </property>
```

(4) 修改 io.tmpdir 路径, 将所有包含 \${system:java.io.tmpdir} 字段的 value 即路径替换为 /usr/local/hadoop/hive-2.1.1/iotmp。Vim 下全局替换命令如下:
:%s/\${system:java.io.tmpdir}#/usr/local/hadoop/hive-2.1.1/iotmp#g

5. 修改 hive 配置文件: cp hive-env.sh.template hive-env.sh

```
47 # Set HADOOP_HOME to point to a specific hadoop install directory
48 # HADOOP_HOME=${bin}/../hadoop
49 HADOOP_HOME=/usr/local/hadoop/hadoop-2.7.1
50
51 # Hive Configuration Directory can be controlled by:
52 # export HIVE_CONF_DIR=
53 export HIVE_CONF_DIR=/usr/local/hadoop/hive-2.1.1/conf
54
55 # Folder containing extra libraries required for hive compilation/execution can be controlled by:
56 # export HIVE_AUX_JARS_PATH=
57 export HIVE_AUX_JARS_PATH=/usr/local/hadoop/hive-2.1.1/lib
```

6. 创建相关文件夹并赋予权限

```
mkdir -p warehouse
chmod a+rwX warehouse
mkdir iotmp
chmod 777 iotmp
```

7. 修改 hive-config.sh

```
19 export JAVA_HOME=/usr/local/java/jdk1.7.0_25
20 export HADOOP_HOME=/usr/local/hadoop/hadoop-2.7.1
21 export HIVE_HOME=/usr/local/hadoop/hive-2.1.1
```

8. 初始化数据库

```
hadoop@ubuntu:/usr/local/hadoop/hive-2.1.1/bin$ ./schematool -initSchema -dbType
derby
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hive-2.1.1/lib/log4j-slf4j-impl-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hbase-1.2.5/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL: jdbc:derby:/usr/local/hadoop/hive-2.1.1/bin/metastore_db;databaseName=metastore_db;create=true
Metastore Connection Driver : org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User: APP
Starting metastore schema initialization to 2.1.0
Initialization script hive-schema-2.1.0.derby.sql
Initialization script completed
schemaTool completed
```

八、 在 Hive 中执行操作

1. 创建 Wuxia 表

```
yq-mapreduce@ubuntu: ~
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hive-2.1.1/lib/log4j-slf4j-impl-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hbase-1.2.5/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/usr/local/hadoop/hive-2.1.1/lib/hive-common-2.1.1.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive> create table Wuxia
  > (word STRING, count DOUBLE)
  > ROW FORMAT DELIMITED
  > FIELDS TERMINATED BY '\t'
  > STORED AS TEXTFILE;
OK
Time taken: 1.778 seconds
hive> █
```

2. 将从 HBase 中导出的数据导入 Wuxia 表

```
hive> LOAD DATA LOCAL INPATH '/home/hadoop/HBaseOut.txt' OVERWRITE INTO TABLE Wuxia
  > ;
Loading data to table default.wuxia
OK
Time taken: 0.943 seconds
hive> █
```


3. 查询出现次数大于 300 的词:

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/home/hadoop/HiveOut' SELECT * FROM Wuxi
a WHERE count >= 300;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futu
re versions. Consider using a different execution engine (i.e. spark, tez) or us
ing Hive 1.X releases.
Query ID = hadoop_20170511180908_ae651319-0e5f-4577-9ab5-1c4769d32fbd
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1494496061333_0001, Tracking URL = http://ubuntu:8088/proxy/a
pplication_1494496061333_0001/
Kill Command = /usr/local/hadoop/hadoop-2.7.1/bin/hadoop job -kill job_14944960
61333_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-05-11 18:11:07,652 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 4.31
sec
MapReduce Total cumulative CPU time: 4 seconds 310 msec
Ended Job = job_1494496061333_0001
Moving data to local directory /home/hadoop/HiveOut
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Cumulative CPU: 4.31 sec HDFS Read: 0 HDFS Write: 0 SU
CCESS
Total MapReduce CPU Time Spent: 4 seconds 310 msec
OK
Time taken: 119.271 seconds
```

```

yq-mapreduce@ubuntu: ~
1 一^A753.2
2 一个^A448.27
3 一声^A327.49
4 丁典^A364.0
5 丁玲^A586.5
6 万成^A962.5
7 万震山^A333.0
8 不^A494.76
9 东方龙^A544.0
10 两利^A1471.89
11 中^A541.02
12 之^A391.53
13 乌老大^A302.0
14 乐圣^A889.5
15 也^A1345.75
16 了^A1574.48
17 人^A340.51
18 什么^A332.74
19 他^A2614.89
20 他们^A568.61
21 令狐冲^A1905.0
22 仪琳^A729.0
23 伍元^A934.0
24 但^A597.12
25 余沧海^A378.0

1,1 Top
```

4. 查询前 100 次数最多的词

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/home/hadoop/HiveOut' SELECT * FROM Wuxia ORDER BY count desc LIMIT 100;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e.
park, tez) or using Hive 1.X releases.
Query ID = hadoop_20170511181434_920613cf-3109-419a-8c32-7578dae51636
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1494496061333_0002, Tracking URL = http://ubuntu:8088/proxy/application_1494496061333_0002/
Kill Command = /usr/local/hadoop/hadoop-2.7.1/bin/hadoop job -kill job_1494496061333_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-05-11 18:15:42,493 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 4.74 sec
MapReduce Total cumulative CPU time: 4 seconds 740 msec
Ended Job = job_1494496061333_0002
Moving data to local directory /home/hadoop/HiveOut
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 4.74 sec HDFS Read: 0 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 740 msec
OK
Time taken: 68.614 seconds
hive>
```

```
yq-mapreduce@ubuntu: ~
1 的^A7168.19
2 杜英豪^A4230.0
3 韦小宝^A3277.0
4 道^A3272.61
5 他^A2614.89
6 你^A2517.53
7 玮^A2438.67
8 我^A2373.63
9 张无忌^A2338.0
10 令狐冲^A1905.0
11 程小蝶^A1898.5
12 小鱼儿^A1756.25
13 齐金蝉^A1744.0
14 段誉^A1688.0
15 虚竹^A1606.0
16 了^A1574.48
17 芮^A1528.4
18 两利^A1471.89
19 百维^A1416.0
20 狄云^A1408.0
21 也^A1345.75
22 是^A1336.1
23 黄蓉^A1262.5
24 易天行^A1229.0
25 岳不群^A1184.0
"Top100" 100L, 1392C 1,1 Top
```

```
yq-mapreduce@ubuntu: ~
78 又^A532.95
79 向问天^A516.0
80 杨道^A515.0
81 和^A505.75
82 小明^A505.6
83 珪^A505.5
84 说道^A505.12
85 不^A494.76
86 阿朱^A493.0
87 笑道^A488.38
88 左冷禅^A482.0
89 小赌^A474.6
90 婉丽^A472.0
91 陈近南^A471.0
92 非子^A459.6
93 慕容复^A459.5
94 海瑞^A455.0
95 一个^A448.27
96 陆文^A447.4
97 白万剑^A443.0
98 就^A441.36
99 李文秀^A441.0
100 水笙^A439.0

95,1 Bot
```

九、 附加内容：停词表

1. 代码说明：

包括两个文件，其中 StopwordToHBase.java 是将停词表中的词存入 HBase 中。
主要介绍 WuxiaWithStopWords.java（仅展示与不含停词表的代码中的不同之处）
在 map 时连接 HBase

```
protected void cleanup(Mapper.Context context) throws IOException, InterruptedException{
    table.close();
    con.close();
}
```

查询是否在停词表中，若不在则写入

```
while (itr.hasMoreTokens()) {
    String keystr = itr.nextToken();

    Get get = new Get(keystr.getBytes());
    Result r = table.get(get);
    if(r.isEmpty()) {
        Text keyinfo = new Text( string: keystr + ':' + filename);
        context.write(keyinfo, new Text( string: "1"));
    }
}
```

断开与停词表的连接

```
protected void setup(Mapper.Context context) throws IOException, InterruptedException{
    con = ConnectionFactory.createConnection(configuration);
    table = con.getTable(TableName.valueOf("StopWords"));
}
```

2. 由于 wuxia 数据量太大，而停词表运行很慢，所以缩减了数据量，只选取了 wuxia 中的前 10 个文件，并且每个文件只截取了一部分，为了保证这 10 个文件中有词在停词表中，手动在第二个文件中添加了停词表中的词“上面”，在第四个文件中添加了停词表中的词“不仅仅是”、“不但”。

卧龙生02.春秋笔.txt.segmented ✕

TXT 合集 网 独家 整理
www.txthj.com
卧龙生
春秋 笔
第一回
奇人 传 绝学
江山代有才人出 各领风骚 二十年

上面

武林中 的人事 变幻 更为 快速 除了 像 少林 武当 四大 世家 那种 基础 雄厚 弟子 众多 的
大 门户 享 名 久远 之外 江湖 上 的 新 陈 替代 都不会 太久 多则 十年 少则 三五年 都会
有 一个 转变
其实 少林 武当 四大 世家 那样 基础 雄厚 的 门户 也有 黯淡 的 时间
现在 正是 这样 一个 时刻
少林 武当 两大 门户 的 弟子 人才 凋零 很少 在 江湖 上 走动
武林 四大 世家 也 由 极盛 转 入 了 衰退
江湖 上 的 盛 外 得 之 不易 维护 更难 那要 付出 无与伦比 的 代价 血泪斑斑 往事 可考
因此 从没有 一个 门户 世家 能够 永享 江湖 盛名
现 面 如日中天 在 江湖 上 最 享 声誉 的 是 无极 门 主 青萍 剑宗 领 刚
立 窑 在 襄阳 府 隆中 山下 的 无极 门 占地 不过 数十 亩 谈不上 什么 宏伟 连 仆 徒 门
人 算上 也 不过 百来 号 比起 江湖 上 那些 壮大 的 门派 实在 算不得 什么
代表 无极 门 威望 的 是 大 门上 那块 横匾
那只是 上好 松木 做成 的 横匾 黑漆 金字 写着 无极 门 三个 大字
价值 在 那块 木匾 下面 的 署名 包括 了 少林 武当 掌门人 东方 世家 主人 丐帮 帮主 排 教

卧龙生04.地狱门.txt.segmented ✕

这天 就在这 大雪纷飞 的 这一天 万物 皆 晶莹 的 大千世界 九华山 冷 谷 中 的 地 狱门 却
是 一片 的 红
何谓 地 狱门
不仅仅是

地 狱门 乃 是 由 武林 中一 群 正气 磅礴 得 悲天悯人 的 前辈 高人 所 发起 所 组织
他们 按照 明哲 地府 的 条 律 次序 设置 森 罗 主殿 城隍 上地 、 文武 判官 、 黑白 无常
牛头马面 、
以及 日 夜游神 等等 的 职司 和 责任
这是 干什么 呀
当然是 执 起 法外 之法 武林 之法 以 遏阻 江湖 上 凶杀 之风 暴戾 之气
凡 有人 越礼 犯 分 过分 的 跋扈 恣 凡 有人 杀人 害命 过分 的 猖 獗 扬厉 经 土地 具报
经 城隍
查证
再 经过 阎罗
不但

们 审议 裁决
若 该 入 阳寿 已 终 应 登 鬼 录 则 呈请 菩萨 降旨 下 谕 遴选 适任 之 地 抵 阴 兵 拘提
归 阴
那 后来 呢
在 地 狱中 没有 刑罚 没有 血腥
神道 们 和 光 问 尘 地 抵 们 待 鬼 以 诚 予以 开导 予以 教化 一 但 犯 鬼 混 却 凶 念
滤去 恶性 然后 再 遣 送出 投生 使 对方 再度 为 人 成为 良民
普 萨 准 已 普 萨

3. 输出结果:

将运行结果存到 HBase 中的 WuxiaStopWords 表中，并保存至本地文件中。可在文件中查看没有“上面”、“不仅仅是”、“不但”等词。

WithStopWordsOut.txt ✕		WithStopWordsOut.txt ✕	
下药	1.00	二寸	1.00
下落	1.00	三尺	1.00
不乏	1.00	三年	1.00
不便	1.00	三更	1.00
不停地	1.00	三月	1.00
不出	2.00	三环	1.00
不到	1.00	三百里	1.00
不剩	1.00	三老	1.00
不动	1.00	三面	1.00
不及	1.00	上一个	1.00
不啻	1.00	上万	1.00
不在	1.00	上了	1.00
不多	1.00	上交	1.00
不太	1.00	上地	1.00
不好受	1.00	上好	1.00
不待	1.00	上流	1.00
不愿	1.00	上车	1.00
不改	1.00	上钢	1.00
不放心	1.00	上高	1.00
不易	1.00	下了	1.00
不用	1.00	下山	1.00
不知	1.00	下有	1.00
不禁	1.00	下药	1.00
不耐	1.00	下落	1.00
不自禁	1.00		
不要紧	1.00		
不辞	1.00		
不远处	2.00		
不迟	1.00		
不错	1.50		
不高不矮	1.00		
与世无争	1.00		

十、 实验体会

1. 配置文件很重要！实验的很大一部分时间花费在了修改配置文件上，所以需要先明白配置文件的作用再进行修改，可以减少错误。
2. 在编译时成功，但是运行时遇到了 ClassNotFound 的问题，后来将在 Hbase 的 lib 文件夹中相关的 jar 包拷贝至 hadoop 放 jar 包的地方就没有报错了。