

大数据实验3—HBase_Hive 实验报告

陈越琦

(121160005 Yueqichen.0x0@gmail.com)

刘威

(131220085 liuwei13cs@smail.nju.edu.cn)

杨杰才

(131220115 mark.grove@qq.com)

摘 要: 本次实验我们小组首先在本地上安装并配置了伪分布式HBase与Hive环境, 并且在实验2的基础上, 完成在HBase上的插入与遍历任务以及在Hive上的导入与查询任务。此外还完成了选做任务: 使用停词表处理倒排文档。

关键词: Hadoop、倒排索引、HBase、Hive、停词表

§1. 引 言

HBase是一个建立在HDFS之上的分布式数据库, 提供了结构化/半结构化数据的存储管理和访问能力。为了进一步方便开发人员开发大数据查询分析处理程序, 基于Hadoopde 数据仓库Hive提供了类似SQL的语言描述数据处理逻辑。在实验2的基础上, 本次实验进一步熟悉HBase和Hive的原理和使用方法, 完成在HBase与Hive上的相关任务: (1)将倒排索引结果部分内容导入到HBase中, (2)从HBase中读出数据到本地文件系统, (3)从本地文件系统中倒排结果导入到Hive中。同时还完成了选做任务: (4)将停词表导入HBase, 并在停词表的作用下重复(1)(2)任务。

实验报告的第2节简要介绍了实验环境和完成情况。第3节中将详细介绍实验各个部分的设计。测试与运行的结果留在第4节中展示。在第5节中总结实验内容和团队合作。

*

*陈越琦: 121160005 完成选做任务并编写相应实验报告和最后报告修改
刘威: 131220085 完成必做任务前四部分并编写相应实验报告
杨杰才: 121160005 完成必做任务Hive部分并编写相应实验报告

目录

§1. 引 言	1
§2. 实验环境与概述	3
2.1 HBase与Hive安装与配置	3
2.1.1 HBase的安装与配置	3
2.1.2 Hive的安装与配置	4
§3. 实验设计思路	5
3.1 倒排索引部分结果插入到HBase中	5
3.2 把HBase中”Wuxia”表保存到本地	5
3.3 将数据导入到Hive中并查询	6
3.4 从HBase中导入停词表并将倒排索引部分结果插入到HBase中	6
3.4.1 导入停词表	6
3.4.2 读入停词表并执行倒排索引	7
§4. 实验测试与运行结果	8
4.1 倒排索引部分结果插入到HBase中	8
4.2 把HBase中”Wuxia”表保存到本地	10
4.3 将数据导入到Hive中并查询	10
4.4 从HBase中导入停词表并将倒排索引部分结果插入到HBase中	16
§5. 实验总结	18
5.1 实验内容总结	18
5.2 团队合作总结	18

§2. 实验环境与概述

本次实验的本地开发与测试环境如下：

表 1 开发测试环境

软件	版本号
OS	ubuntu 15.04
kernel	4.2.0-36-generic
JDK	1.8.0.66
Hadoop	2.7.1
Hbase	1.2.1
Hive	1.2

本次实验依次完成了以下实验任务

1. 安装HBase和Hive.
2. 在HBase中创建“Wuxia”表并修改第2次实验中的MapReduce程序，在Reduce阶段将倒排索引的信息通过文件输出，同时把每个词语对应的“平均出现次数”信息写入到HBase的”Wuxia”表中.
3. 将HBase中“Wuxia”表的表格内容保存在本地文件中.
4. 通过Hive Shell命令行创建表”Wuxia”，导入平均出现次数和相应词语并查询.
5. 从HBase中读入事先导入的停词表，重复任务2.

2.1 HBase与Hive安装与配置

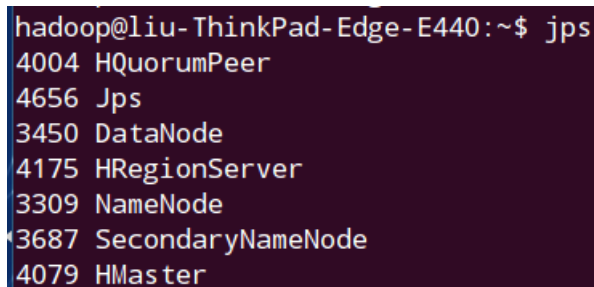
2.1.1 HBase的安装与配置

根据教程[2]安装与配置，具体过程如下：

1. 查找与Hadoop版本匹配的HBase版本，下载安装包.
2. 解压安装包并设置环境变量.
3. 修改配置文件：\$HBASE_HOME/conf/hbase-env.sh 和 \$HBASE_HOME/conf/hbase-site.xml
4. 启动HDFS，再通过start-hbase.sh脚本启动HBase，通过shell简单测试后，HBase正常运行，配置成功.

5. 配置\$HADOOP_HOME环境变量，使得依赖于HBase中jar包的MapReduce作业自动产生依赖，方便后续实验

使用jps命令查看当前运行的java进程如下：



```
hadoop@liu-ThinkPad-Edge-E440:~$ jps
4004 HQuorumPeer
4656 Jps
3450 DataNode
4175 HRegionServer
3309 NameNode
3687 SecondaryNameNode
4079 HMaster
```

图 1 jps命令查看java进程

2.1.2 Hive的安装与配置

根据教程[3]安装与配置，具体过程如下：本次使用的Hadoop版本是2.7.1，通过官方文档查询得知与之较好匹配的Hive版本是1.2.1，下载安装包后，按照如下步骤进行安装。

1. 查找与Hadoop版本匹配的Hive版本，下载安装包。
2. 使用apt-get命令安装SQL并设置环境变量
3. 重命名\$HIVE_HOME/conf/hive-default.xml为hive-site.xml，并配置hive-site.xml
4. 下载mysql-connector-java-5.1.27-bin.jar文件，并存储到\$HIVE_HOME/lib目录
5. 启动HDFS，再运行Hive，通过shell简单测试后，Hive正常运行，配置成功。

§3. 实验设计思路

3.1 倒排索引部分结果插入到HBase中

本部分使用HBase的JAVA API在MapReduce作业的Reduce阶段对HBase数据库进行操作：将词语与平均出现次数插入到‘Wuxia’表中，除平均出现次数之外的倒排索引信息仍输出到HDFS中。在HBase中使用词语作为‘Wuxia’表每一行的rowKey，相应的平均出现次数属于列‘content:fre’。具体设计思路如下：

首先在HBase中通过shell命令行建立“Wuxia”表：

```
hbase(main):xxx:x>create 'Wuxia', 'content'
```

在实验2的基础上对Reducer类部分代码进行修改：我们在每个Reduce任务中将每个词语的平均出现次数和词语构造为一个Put类的实例，将这个实例插入到全局Put类型的List中，关键语句如下：

```
static List<Put> putList = new ArrayList<Put>(); //全局变量
...
//在reduce方法中，插入以下语句构造相应的Put变量：
Put put = new Put(Bytes.toBytes(CurrentItem.toString()));
put.add(Bytes.toBytes(new String("content")), Bytes.toBytes(new String("frequency")),
Bytes.toBytes(Double.toString(fre)));
putList.add(put);
```

最后，在cleanup方法中，建立一个HBase Job，并通过zookeeper建立连接，调用HTable类的put方法将List中的内容插入到HBase中，关键语句如下：

```
//在HBase类的构造函数中，建立HBase Job，并通过zookeeper建立连接
conf = HBaseConfiguration.create();
conf.set("hbase.zookeeper.property.clientPort", "2181");
...
//在cleanup阶段将结果一次性输入到HBase中
HBase hbase = new HBase();
hbase.addDatas("Wuxia",putList);
```

3.2 把HBase中“Wuxia”表保存到本地

本部分通过调用HBase的JAVA API实现对‘Wuxia’表的遍历，将“Wuxia”表中数据按照<词语> TAB <平均出现次数> 的格式输出到输出文件output中。

我们使用io.File类完成文件的输出操作。实验的大体设计分为以下三部分：

1. 建立一个HBase Job，通过zookeeper建立连接，关键代码如下：

```
conf = HBaseConfiguration.create();
conf.set("hbase.zookeeper.quorum", "localhost");
conf.set("hbase.zookeeper.property.clientPort", "2181");
```

2. 建立输出文件并通过HTable提供的getScanner方法来进行批量查询，创建Scan变量时使用默认空参数从而实现对整个表的遍历，关键代码如下：

```
File writename = new File("output");
writename.createNewFile();
Table table = new HTable(conf, tableName);
Scan s = new Scan();
ResultScanner ss = table.getScanner(s);
```

3. 遍历批量查询结果，将相应的词语名rowKey与平均出现次数frequency输出到output文件，关键代码如下：

```
//对ResultScanner中每一个Result r的每一个KeyValue kv
out.write(new String(kv.getRow()));
out.write(new String(kv.getValue()));
```

3.3 将数据导入到Hive中并查询

本部分，我们通过Hive Shell来完成。首先创建Table Wuxia，然后将上一任务中的输出文件output导入到Wuxia中，最后通过相应的查询指令进行查询。具体的过程在第四节中详细叙述。

3.4 从HBase中导入停词表并将倒排索引部分结果插入到HBase中

本部分任务分成两个阶段，一是将停词表导入HBase中，二是从HBase中读入停词表并执行倒排索引。

3.4.1 导入停词表

1. 在HBase中通过shell命令行建立"StopWords"表：

```
hbase(main):xxx:x>create 'StopWords', 'word'
```

2. 通过zookeeper建立连接，再通过InputStreamReader和BufferedReader读入停词表，以停词表中的词语作为RowKey插入到HBase中。

3.4.2 读入停词表并执行倒排索引

本部分在3.1的基础上，修改Map阶段的代码：在Map阶段的setup方法中，使用类似于3.2中的方法建立HBase连接并读入”StopWords”到全局停词表使得每个节点都能共享。同时在map方法中对于每个小说中读入的词语检查在停词表中是否存在匹配，若是则跳过这个词语并继续。关键语句如下：

```
//在setup方法中建立停词表
stopwords = new TreeSet<String>();
//读入HBase中”StopWords”中内容并添加进停词表
stopwords.add(new String(kv.getRow()));
...
//在map方法中检查是否存在匹配
if(!stopwords.contains(temp))
```

§4. 实验测试与运行结果

4.1 倒排索引部分结果插入到HBase中

本部分的测试与运行过程如下：

- (1) 将集群HDFS中的测试文件复制到本地，然后使用scp拷贝到我们的机器中。
- (2) 将修改后的代码编译得到.class文件并打包为InvertedIndex.jar包。
- (3) 在HBase中创建表'Wuxia'，如下图：

```
hbase(main):006:0> create 'Wuxia','content'
0 row(s) in 2.3060 seconds

=> Hbase::Table - Wuxia
hbase(main):007:0> list
TABLE
Wuxia
member
students
3 row(s) in 0.0050 seconds

=> ["Wuxia", "member", "students"]
hbase(main):008:0> describe 'Wuxia'
Table Wuxia is ENABLED
Wuxia
COLUMN FAMILIES DESCRIPTION
{NAME => 'content', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false',
KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER',
COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE =>
'65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.0350 seconds
```

图 2 建表与查看

- (4) 使用命令`hadoop jar InvertedIndex.jar InvertedIndex Lab3 Lab3out` 运行这一任务。

本任务的运行结果如下：


```

0      卧龙生07飞燕惊龙:1; 卧龙生37天涯情侣:1; 卧龙生42新仙鹤神针:1; 卧龙生45燕子传奇:1; 李凉07赌棍小狂侠:5; 李凉15江湖一担皮:1; 李凉21六宝江湖行:1; 李凉23妙贼丁小勾:3; 李凉27奇神杨小邪:1; 李凉38笑笑江湖:4; 梁羽生01白发魔女传:1; 梁羽生11广陵剑:1; 梁羽生12瀚海雄风:1; 梁羽生25牧野流星:3; 梁羽生34武当一剑:1; 金庸07鹿鼎记:1;
007    李凉12活宝小淘气:1;
01     卧龙生45燕子传奇:1; 李凉23妙贼丁小勾:1;
01章   古龙60神君别传:1;
02章   古龙60神君别传:1;
03章   古龙60神君别传:1;
04     李凉26奇神杨小邪续集:1;
04章   古龙60神君别传:1;
05     李凉23妙贼丁小勾:1;
05章   古龙60神君别传:1;
06     李凉23妙贼丁小勾:1;
06章   古龙60神君别传:1;
07章   古龙60神君别传:1;
08张   卧龙生47一代天骄:1;
08章   古龙60神君别传:1;
09章   古龙60神君别传:1;
0一    卧龙生46摇花放鹰传:1; 李凉34天下第一当:2;
0年    梁羽生01白发魔女传:4; 梁羽生33随笔集:三剑楼随笔:1;

```

图 3 HDFS中的输出倒排索引信息

可以看到，倒排索引中的信息已不再输出平均出现次数。

使用scan 'Wuxia' 命令查看HBase中'Wuxia'表的插入结果如下（rowKey为词语的UTF-8编码，frequency为该词语的平均出现次数）：

```

\xE9\xBE\x9F\xE7\xBA column=content:frequency, timestamp=1462887671554, value=3
\xB9 .0
\xE9\xBE\x9F\xE7\xBC column=content:frequency, timestamp=1462887671554, value=1
\xA9 .4375
\xE9\xBE\x9F\xE8\x82 column=content:frequency, timestamp=1462887671554, value=1
\x89 .3333333333333333
\xE9\xBE\x9F\xE8\x83 column=content:frequency, timestamp=1462887671554, value=2
\x8C .5
\xE9\xBE\x9F\xE8\xA3 column=content:frequency, timestamp=1462887671554, value=1
\x82 .2857142857142858
\xE9\xBE\x9F\xE9\xB3 column=content:frequency, timestamp=1462887671554, value=2
\x96 .0
\xE9\xBE\x9F\xE9\xB9 column=content:frequency, timestamp=1462887671554, value=1
\xA4\xE9\x81\x90\xE9 .0
\xBE\x84
\xEF\xA8\x8C column=content:frequency, timestamp=1462887671554, value=5
.0
\xEF\xBF\xA1 column=content:frequency, timestamp=1462887671554, value=1
.5
\xEF\xBF\xA5 column=content:frequency, timestamp=1462887671554, value=3
.3333333333333335
134881 row(s) in 17.1050 seconds

```

图 4 'Wuxia'表中信息

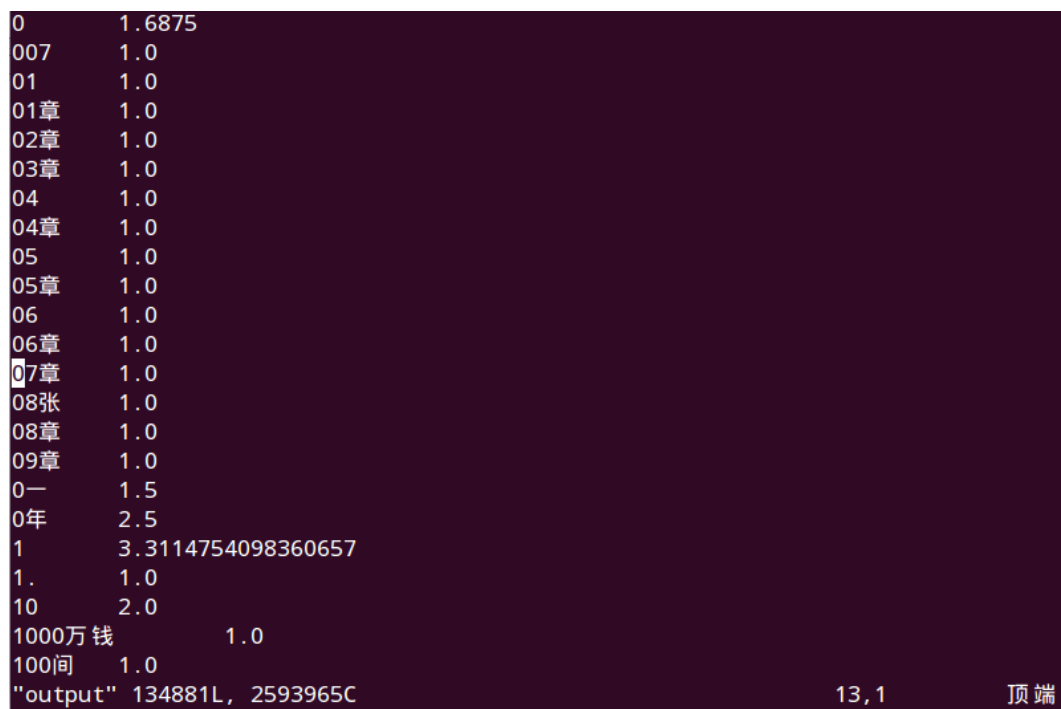
可以看到总共134881条记录已经全部成功插入到'Wuxia'表中。

4.2 把HBase中"Wuxia"表保存到本地

本任务的测试与运行过程如下：

- (1) 编译ReadHBase.java
- (2) 使用java -classpath <依赖文件> ReadHBase 命令执行任务
- (3) 执行结束后，在当前文件夹下即可看到输出

得到的输出文件output的部分内容的截图如下：



```
0      1.6875
007    1.0
01     1.0
01章   1.0
02章   1.0
03章   1.0
04     1.0
04章   1.0
05     1.0
05章   1.0
06     1.0
06章   1.0
07章   1.0
08张   1.0
08章   1.0
09章   1.0
0一    1.5
0年    2.5
1      3.3114754098360657
1.     1.0
10     2.0
1000万 钱      1.0
100间  1.0
"output" 134881L, 2593965C
```

图 5 'Wuxia'表保存到本地的文件output

4.3 将数据导入到Hive中并查询

本部分全部在Hive Shell下完成，首先建表，然后将上面的输出文件output导入表中，最后查询。相应的命令与查询结果见下面的截图：

图6-7显示了建表和把数据导入Hive的过程

```
hive> create table Wuxia(word STRING,count DOUBLE) row format delimited fields
terminated by '\t' stored as textfile;
OK
Time taken: 0.141 seconds
hive> show tables;
OK
wuxia
Time taken: 0.048 seconds, Fetched: 1 row(s)
hive> █
```

图 6 创建表

```
hive> load data inpath"hdfs:///test/output" into table wuxia;
Loading data to table default.wuxia
Table default.wuxia stats: [numFiles=1, totalSize=2593965]
OK
Time taken: 1.112 seconds
```

图 7 将数据导入Hive

图8显示了平均出现次数大于300的词语

```
hive> select * from wuxia where count>300;
OK
一个 753.2018348623853
一声 448.27064220183485
丁典 327.4908256880734
丁玲 364.0
万成 586.5
万震山 962.5
不 333.0
东方龙 494.75688073394497
两利 544.0
中之 1471.888888888889
乌老大 541.0183486238532
乐圣 391.5321100917431
也 302.0
了 889.5
人 1345.7522935779816
什么 1574.4770642201836
他 340.5091743119266
他们 332.74056603773585
令狐冲 2614.8899082568805
仪琳 568.6146788990826
伍元 1905.0
但 729.0
余沧海 934.0
余鱼同 597.1244239631336
你 378.0
你们 304.0
信 2517.532110091743
俞 302.7649769585253
侗 345.6
凌云凤 383.962962962963
凤梧 329.0
刀儿 415.6666666666667
剑 322.0
十一郎 368.0
南宮 381.0281690140845
南江 406.0
892.6
```

却	405.5321100917431
去	343.8394495412844
又	532.954128440367
只	338.07339449541286
叶开道	975.0
叶长青	724.6666666666666
向问天	516.0
周仲英	405.0
周伯通	412.6666666666667
周绮	556.0
周芷若	413.5
和	505.7522935779817
咱们	343.4720812182741
虞花	334.0
在	745.6192660550458
大	396.54128440366975
她	805.0
婉丽	472.0
完颜	344.35714285714283
完颜洪烈	346.0
宝元	360.0
宫锦	646.5
寒秋	995.1428571428571
寒衣	324.0
晁	575.3125
对	327.5229357798165
小	678.683486238532
小凤	374.84375
小明	505.6
小熊	358.5
小赌	474.6
小鱼儿	1756.25
小龙女	650.0
就	441.36238532110093
岳不群	1184.0
岳灵珊	919.0
左冷禅	482.0
已	627.3761467889908
张无忌	2338.0
张翠山	573.5
後	329.27272727272725
徐天宏	343.5
思南	629.8
思思	317.6666666666667
慕容	354.28205128205127
慕容复	459.5
我	2373.6284403669724
我们	323.91705069124424
戚芳	390.0
方怡	404.0
方证	340.0
易天行	1229.0
是	1336.1009174311926
有	615.1330275229358
木婉清	370.5
朱七七	800.75
朱文	355.0
李文秀	441.0
李莫愁	973.0
李青	381.6666666666667
杜英豪	4230.0
来	401.3211009174312
杨凡	592.0
杨过	1021.6
杨逍	515.0
林平之	929.0
楚天舒	1110.5
楚留香	418.22727272727275
欧阳克	306.5
段正淳	741.0
段誉	1688.0
水笙	439.0
洪七公	312.25
海瑞	455.0
熊猫儿	625.3333333333334
燕南天	362.5
狄云	1408.0
玉	335.5962441314554

王语嫣	430.5
玮	2438.6666666666665
珪	338.8
珪	505.5
田伯光	721.0
白万剑	443.0
白云飞	726.0
白飞飞	626.0
白维	1416.0
的	7168.192660550459
知道	331.7889908256881
石破天	598.0
程小蝶	1898.5
程灵素	383.0
笑道	488.3824884792627
罗刚	341.0
罗烈	305.0
胡斐	997.3333333333334
自己	375.77522935779814
丙	1528.4
花无缺	764.5
茅十八	396.0
蓬莱	319.2608695652174
虚竹	1606.0
袁承志	1013.3333333333334
婁	402.05045871559633
说	419.6238532110092
说道	505.11961722488036
谢晓峰	407.5
谢烟客	335.0
谢逊	607.0
赵敏	626.5
这	932.0
邀	316.5
道	3272.6146788990827
那	983.2660550458716
郑克爽	408.0
郭襄	346.6666666666667
郭靖	1073.0
都	336.38532110091745
金贵	705.6666666666666
铁心兰	887.0
阿三	420.6666666666667
阿朱	493.0
阿碧	305.0
阿紫	567.0
陆天	570.0
陆文	447.4
陈家洛	708.0
陈近南	471.0
非子	459.6
韦小宝	3277.0
马春花	325.0
鸠摩智	563.0
麼	383.3
黄药师	370.3333333333333
黄蓉	1262.5
齐金蝉	1744.0
Time taken: 2.689 seconds, Fetched: 174 row(s)	

图 8 查询平均出现次数大于300的词语

图9显示了查询平均出现次数前100的词语

```

hive> select * from wuxia sort by count desc limit 100;
Query ID = yjc_20160511194806_0bac016a-6c13-4907-a3a0-2e31238b8eda
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 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>
Job running in-process (local Hadoop)
2016-05-11 19:48:08,391 Stage-1 map = 0%, reduce = 0%
2016-05-11 19:48:09,410 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local186372307_0003
Launching Job 2 out of 2
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 hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2016-05-11 19:48:11,179 Stage-2 map = 100%, reduce = 100%
Ended Job = job_local1570581327_0004
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 10375860 HDFS Write: 0 SUCCESS
Stage-Stage-2: HDFS Read: 10375860 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
的 7168.192660550459
杜英豪 4230.0
韦小宝 3277.0
道 3272.6146788990827
他 2614.8899082568805
你 2517.532110091743
玮 2438.6666666666665
我 2373.6284403669724
张无忌 2338.0
令狐冲 1905.0
程小蝶 1898.5
小鱼儿 1756.25
齐金蝉 1744.0
段誉 1688.0
虚竹 1606.0
了 1574.4770642201836
芮 1528.4
两利 1471.888888888889
白维 1416.0
狄云 1408.0
也 1345.7522935779816
是 1336.1009174311926
黄蓉 1262.5
易天行 1229.0
岳不群 1184.0
楚天舒 1110.5
郭靖 1073.0
杨过 1021.6
袁承志 1013.3333333333334
胡斐 997.3333333333334
寒秋 995.1428571428571
那 983.2660550458716
叶开道 975.0
李莫愁 973.0
万成 962.5
伍元 934.0
这 932.0
林平之 929.0
岳灵珊 919.0
南江 892.6
乐圣 889.5
铁心兰 887.0
她 805.0
朱七七 800.75
花无缺 764.5
一 753.2018348623853
在 745.6192660550458
段正淳 741.0
仪琳 729.0
白云飞 726.0
叶长青 724.6666666666666

```

田伯光	721.0
陈家洛	708.0
金贵	705.6666666666666
小	678.683486238532
小龙女	650.0
宫锦	646.5
思南	629.8
已	627.3761467889908
赵敏	626.5
白飞飞	626.0
熊猫儿	625.3333333333334
有	615.1330275229358
谢逊	607.0
石破天	598.0
但	597.1244239631336
杨凡	592.0
丁玲	586.5
袁	575.3125
张翠山	573.5
陆天	570.0
他们	568.6146788990826
阿紫	567.0
鸠摩智	563.0
周绮	556.0
东方龙	544.0
中	541.0183486238532
又	532.954128440367
向问天	516.0
杨逍	515.0
和	505.7522935779817
小明	505.6
挂	505.5
说道	505.11961722488036
不	494.75688073394497
阿朱	493.0
笑道	488.3824884792627
左冷禅	482.0
小赌	474.6
婉丽	472.0
陈近南	471.0
非子	459.6
慕容复	459.5
海瑞	455.0
一个	448.27064220183485
陆文	447.4
白万剑	443.0
就	441.36238532110093
李文秀	441.0
水笙	439.0

Time taken: 4.54 seconds, Fetched: 100 row(s)

图 9 平均次数前100的词语

4.4 从HBase中导入停用词表并将倒排索引部分结果插入到HBase中

图10显示了把停用词表导入HBase中之后，查看”StopWords”表的部分结果。

```
hbase(main):001:0> scan 'StopWords'
COLUMN+CELL
! column=word:content, timestamp=1463065774902, value=value
" column=word:content, timestamp=1463065774914, value=value
# column=word:content, timestamp=1463065774925, value=value
$ column=word:content, timestamp=1463065774928, value=value
% column=word:content, timestamp=1463065774931, value=value
& column=word:content, timestamp=1463065774933, value=value
' column=word:content, timestamp=1463065774937, value=value
( column=word:content, timestamp=1463065774939, value=value
) column=word:content, timestamp=1463065774942, value=value
* column=word:content, timestamp=1463065774944, value=value
+ column=word:content, timestamp=1463065774946, value=value
, column=word:content, timestamp=1463065774949, value=value
- column=word:content, timestamp=1463065774951, value=value
-- column=word:content, timestamp=1463065774953, value=value
. column=word:content, timestamp=1463065774956, value=value
.. column=word:content, timestamp=1463065774958, value=value
... column=word:content, timestamp=1463065774961, value=value
..... column=word:content, timestamp=1463065774963, value=value
..... column=word:content, timestamp=1463065774965, value=value
\xEF\xBD\x83\xEF\xBC column=word:content, timestamp=1463065778032, value=value
\xBD
\xEF\xBD\x85\xEF\xBC column=word:content, timestamp=1463065778033, value=value
\xBD
\xEF\xBD\x86\xEF\xBC column=word:content, timestamp=1463065778034, value=value
\xBD
\xEF\xBD\x8E\xEF\xBD column=word:content, timestamp=1463065778035, value=value
\x87\xE6\x98\x89
\xEF\xBD\x9B column=word:content, timestamp=1463065778037, value=value
\xEF\xBD\x9B\xEF\xBC column=word:content, timestamp=1463065778038, value=value
\x8D
\xEF\xBD\x9C column=word:content, timestamp=1463065778039, value=value
\xEF\xBD\x9D column=word:content, timestamp=1463065778040, value=value
\xEF\xBD\x9D\xEF\xBC column=word:content, timestamp=1463065778041, value=value
\x9E
\xEF\xBD\x9E column=word:content, timestamp=1463065778042, value=value
\xEF\xBD\x9E\xC2\xB1 column=word:content, timestamp=1463065778043, value=value
\xEF\xBD\x9E\xEF\xBC column=word:content, timestamp=1463065778044, value=value
\x8B
\xEF\xBF\xA5 column=word:content, timestamp=1463065778045, value=value
1892 row(s) in 1.3800 seconds
```

图 10 停用词表导入HBase

图11显示了添加停用词表前后的输出比较，可以看到添加停用词：“打开天窗说亮话”之后，输出文件的倒排索引中就不存在这一词条对应的索引。

61366	打开天窗	卧龙生51玉手点将录:1; 梁羽生07弹指惊雷:1; 梁羽生10风云雷电:2; 梁羽生11广陵剑:2; 梁羽生16剑网尘丝:1; 梁羽生18绝塞传烽录:1; 梁羽生19狂侠天娇魔女:2; 梁羽生25牧野流星:2; 梁羽生34武当一剑:1; 梁羽生35武林天骄:1;
61367	打开天窗说亮话	卧龙生01镖旗:2; 卧龙生02春秋笔:1; 卧龙生03翠袖玉环:1; 卧龙生05飞花逐月:1; 卧龙生10黑白剑:1; 卧龙生18金剑雕翎:1; 卧龙生28神州豪侠:1; 卧龙生29双凤旗:1; 卧龙生38铁剑玉佩:1; 卧龙生44烟锁江湖:1; 卧龙生45燕子传奇:1; 古龙07残金缺玉:1; 古龙32剑玄录:1; 李凉15江湖一担皮:3; 李凉16江湖一品郎:1; 李凉25魔手邪怪:1; 李凉33天齐大帝:1; 梁羽生01白发魔女传:1; 梁羽生07弹指惊雷:2; 梁羽生09风雷震九洲:1; 梁羽生10风云雷电:3; 梁羽生11广陵剑:3; 梁羽生12瀚海雄风:3; 梁羽生15慧剑心魔:1; 梁羽生16剑网尘丝:4; 梁羽生18绝塞传烽录:1; 梁羽生19狂侠天娇魔女:2; 梁羽生24鸣镝风云录:9; 梁羽生25牧野流星:6; 梁羽生27萍踪侠影录:1; 梁羽生30散花女侠:1; 梁羽生34武当一剑:2; 梁羽生35武林天骄:2; 梁羽生37游剑江湖:7; 金庸07鹿鼎记:1; 金庸08笑傲江湖:3; 金庸11侠客行:3; 金庸12倚天屠龙记:1; 金庸14鸳鸯刀:1;
61368	打开窗户说亮话	卧龙生19惊鸿一剑震江湖:1; 卧龙生36天香飘:1;
60444	打开天窗	卧龙生51玉手点将录:1; 梁羽生07弹指惊雷:1; 梁羽生10风云雷电:2; 梁羽生11广陵剑:2; 梁羽生16剑网尘丝:1; 梁羽生18绝塞传烽录:1; 梁羽生19狂侠天娇魔女:2; 梁羽生25牧野流星:2; 梁羽生34武当一剑:1; 梁羽生35武林天骄:1;
60445	打开窗户说亮话	卧龙生19惊鸿一剑震江湖:1; 卧龙生36天香飘:1;
60446	打开门	卧龙生07飞燕惊龙:1; 卧龙生08风尘侠隐:1; 卧龙生14剑仙列传:5; 卧龙生19惊鸿一剑震江湖:1; 卧龙生21妙绝天香:3; 卧龙生22女捕头:1; 卧龙生25七绝剑II还情剑:3; 卧龙生26情剑无刃:1; 卧龙生31桃花血令:1; 卧龙生34天龙甲:2; 卧龙生35天马霜衣:1; 卧龙生42新仙鹤神针:1; 卧龙生45燕子传奇:2; 卧龙生46摇花放鹰传:2; 卧龙生49幽灵四艳:1; 古龙02白玉雕龙:1; 古龙04边城刀声:1; 古龙37流星蝴蝶剑:1; 古龙39陆小凤02绣花大盗:1; 古龙43陆小凤06凤舞九天:2; 古龙45名剑风流:3; 古龙46那一剑的风情:1; 古龙47怒剑狂花:2; 古龙55七种武器05霸王枪:1; 李凉06超级邪侠:1; 李凉08公孙小刀:1; 李凉15江湖一担皮:1; 李凉16江湖一品郎:1; 李凉22矛盾天师:1; 李凉25魔手邪怪:1; 李凉26奇神扬小邪续集:1; 李凉27奇神杨小邪:2; 李凉38笑笑江湖:1; 梁羽生03冰河洗剑录:1; 梁羽生10风云雷电:1; 梁羽生11广陵剑:1; 梁羽生36侠骨丹心:1; 金庸05射雕英雄传:2; 金庸07鹿鼎记:1; 金庸08笑傲江湖:1; 金庸09书剑恩仇录:1; 金庸10神雕侠侣:2;

图 11 添加停词表前后比较

§5. 实验总结

5.1 实验内容总结

本次实验是大数据处理综合实验的第3次实验。在实验2的基础上，本次实验完成在HBase与Hive上的相关任务：(1)将倒排索引结果部分内容导入到HBase中，(2)从HBase中读出数据到本地文件系统，(3)从本地文件系统中倒排结果导入到Hive中。同时还完成了选做任务：(4)将停词表导入HBase，并在停词表的作用下重复(1)(2)任务。在进一步熟悉MapReduce编程基础上，学习了HBase和Hive的使用方法。

本次实验的核心内容是HBase和Hive的安装和使用，实验过程中比较多的时间花费在安装上但是好在最后顺利解决了。

代码链接：<https://github.com/chenyueqi/BPLab/tree/master/Lab3/src>

5.2 团队合作总结

在本次实验中，我们使用了“包产到户”的合作模式。即在动手实验之前，通过集体交流确定实验内容和内容之间的依赖关系，然后将实验内容划分成不同部分，分别交由几个人完成。每位同学在完成自己的代码部分的同时完成了对应的实验报告部分。这一合作模式在完成实验3的过程中发挥了一定的作用，但是实验报告的撰写很难取得统一，最后依然需要一位同学检查修改。在今后实验中要注意这方面的配合。

参 考 文 献

- [1] 黄宜华. 深入理解大数据 大数据处理与编程实践[M]. 北京: 机械工业出版社, 2014.7.
- [2] HBase安装配置之伪分布式模式-pdw2009的专栏-博客频道-CSDN.NET
<http://blog.csdn.net/pdw2009/article/details/21261417>
- [3] hadoop2.2 学习5 ubuntu在伪分布式环境下安装hive-幸运声的日志-网易博客
http://blog.163.com/gibby_l/blog/static/83003161201402410204518/