

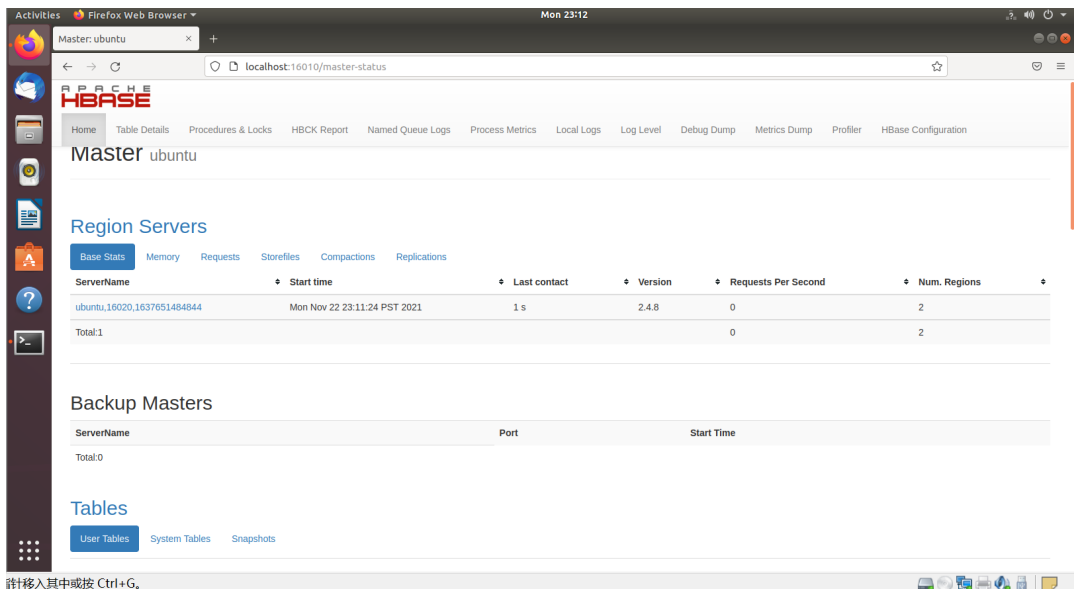
191098323 张峻琿 实验三

一. 环境配置

安装Hbase，成功运行伪分布模式

```
Activities Terminal Mon 19:38
severus@ubuntu: ~/hadoop_installs/hadoop-2.9.1

21/11/29 19:16:58 INFO namenode.FSDirectory: ACLs enabled? false
21/11/29 19:16:58 INFO namenode.FSDirectory: XAttrs enabled? true
21/11/29 19:16:58 INFO namenode.NameNode: Caching file names occurring more than 10 times
21/11/29 19:16:58 INFO SnapshotSnapshotManager: Loaded config captureOpenFiles: false skipCaptureAccessTimeOnlyChange: false
21/11/29 19:16:58 INFO util.GSet: Computing capacity for map cachedBlocks
21/11/29 19:16:58 INFO util.GSet: VM type = 64-bit
21/11/29 19:16:58 INFO util.GSet: 0.25% max memory 966.7 MB = 2.4 MB
21/11/29 19:16:58 INFO util.GSet: capacity = 2^18 = 262144 entries
21/11/29 19:16:58 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
21/11/29 19:16:58 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
21/11/29 19:16:58 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
21/11/29 19:16:58 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
21/11/29 19:16:58 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
21/11/29 19:16:58 INFO util.GSet: Computing capacity for map NameNodeRetryCache
21/11/29 19:16:58 INFO util.GSet: VM type = 64-bit
21/11/29 19:16:58 INFO util.GSet: 0.029999999932944774% max memory 966.7 MB = 297.0 KB
21/11/29 19:16:58 INFO util.GSet: capacity = 2^15 = 32768 entries
21/11/29 19:16:58 INFO namenode.FSImage: Allocated new BlockPoolId: BP-270534658-127.0.1.1-1638242218834
21/11/29 19:16:59 INFO common.Storage: Storage directory /tmp/hadoop-severus/dfs/name has been successfully formatted.
21/11/29 19:16:59 INFO namenode.FSImageFormatProtobuf: Saving image file /tmp/hadoop-severus/dfs/name/current/fsimage.ckpt_000000000000000000 using no compression
21/11/29 19:16:59 INFO namenode.FSImageFormatProtobuf: Image file /tmp/hadoop-severus/dfs/name/current/fsimage.ckpt_000000000000000000 of size 324 bytes saved in 0 seconds
21/11/29 19:16:59 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
21/11/29 19:17:00 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at ubuntu/127.0.1.1
*****/
severus@ubuntu:~/hadoop_installs/hadoop-2.9.1$ sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/severus/hadoop_installs/hadoop-2.9.1/logs/hadoop-severus-namenode-ubuntu.out
localhost: starting datanode, logging to /home/severus/hadoop_installs/hadoop-2.9.1/logs/hadoop-severus-datanode-ubuntu.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/severus/hadoop_installs/hadoop-2.9.1/logs/hadoop-severus-secondarynamenode-ubuntu.out
severus@ubuntu:~/hadoop_installs/hadoop-2.9.1$ jps
13360 NameNode
13856 Jps
13496 DataNode
13678 SecondaryNameNode
severus@ubuntu:~/hadoop_installs/hadoop-2.9.1$ ~/hbase/bin/start-hbase.sh
127.0.0.1: running zookeeper, logging to /home/severus/hbase/bin/./logs/hbase-severus-zookeeper-ubuntu.out
running master, logging to /home/severus/hbase/bin/./logs/hbase-severus-master-ubuntu.out
: running regionserver, logging to /home/severus/hbase/bin/./logs/hbase-severus-regionserver-ubuntu.out
severus@ubuntu:~/hadoop_installs/hadoop-2.9.1$
```



二. 设计Hbase表

以学生的学号为rowKey，每一行为一个学生的相关信息

初始表共4个列族，分别存储学生的个人信息和三门课程；Student列族共3列：S_Name，S_Sex，S_Age；每个课程列族各4列：C_No，C_Credit，C_Name，SC_Score

Student				Math				Computer Science				English			
rowKey(S_No)	S_Name	S_Sex	S_Age	C_No	C_Credit	C_Name	SC_Score	C_No	C_Credit	C_Name	SC_Score	C_No	C_Credit	C_Name	SC_Score
2015001	Li Lei	male	23	123001	2	Math	86	123002	5	Computer Science		123003	3	English	69
2015002	Han Meimei	female	22	123001	2	Math		123002	5	Computer Science	77	123003	3	English	99
2015003	Zhang San	male	24	123001	2	Math	98	123002	5	Computer Science	95	123003	3	English	

三. 编写java程序

1. 创建表格

```
public static void createTable(String tableName, String[] familys) throws Exception{
    HBaseAdmin admin = new HBaseAdmin(conf);
    if (admin.tableExists(tableName)) {
        System.out.println("table"+tableName+"already exists!");
    }else {
        HTableDescriptor tableDesc = new HTableDescriptor(tableName);
        for(int i=0;i < familys.length;i++)
        {
            tableDesc.addFamily(new HColumnDescriptor(familys[i]));
        }
        admin.createTable(tableDesc);
        System.out.println("create table" + tableName + " success!");
    }
}
```

2. 添加数据

```
public static void addData (String tableName, String rowKey, String family, String qualifier, String value)
throws Exception{
    try{
        HTable table = new HTable(conf, tableName);
        Put put = new Put(Bytes.toBytes(rowkey));
        put.add(Bytes.toBytes(family),Bytes.toBytes(qualifier),Bytes.toBytes(value));
        table.put(put);
        System.out.println("insert record success!");
    } catch (IOException e){
        e.printStackTrace();
    }
}
```

3. 根据列族和列查询信息

```
public static void scanByFilter(String tableName, String family, String qualifer, String value)
throws IOException{
    HTable table = new HTable(conf, tableName);
    Scan scan = new Scan();
    scan.addColumn(Bytes.toBytes(family), Byte.toBytes(qualifer));
    Filter filter = new SingleColumnValueFilter(Bytes.toBytes(family),Bytes.toBytes(qualifer),CompareOp.EQUAL, Bytes.toBytes(value));
    scan.setFilter(filter);
    ResultScanner result = table.getScanner(scan);
    for(Result r: result) {
        for(KeyValue kv : r.raw()) {
            if (new String(kv.getQualifier()).equals(SC_Score)){
                System.out.println("value:" + new String(kv.getValue()));
            }
        }
    }
}
```

查询选修Computer Science的学生的成绩

```
HbaseController.scanByFilter("Student_info","Computer Science","C_Name","Computer Science");
```

4. 新增列族

```
public static void addColumn(String tablename, String Columnname){
    Admin adminManger = conn.getAdmin();
    adminManger.disableTable(tablename);
    HTableDescriptor htd = adminManger.getTableDescriptor(tablename);
    HColumnDescriptor f = new HColumnDescriptor(Columnname); //新增列族
    f.setTimeToLive(TIME_TO_LIVE);
    htd.addFamily(f);
    adminManger.addColumn(tablename, f);
    adminManger.enableTableAsync(tablename);
}
```

增加新的列族Contact和新列S_Email, 并添加数据

```
HbaseController.addColumn("Student_info","Contact");
HbaseController.addData("Student_info","2015001","Contact","S_Email","lilie@qq.com");
HbaseController.addData("Student_info","2015002","Contact","S_Email","hmm@qq.com");
HbaseController.addData("Student_info","2015003","Contact","S_Email","zs@qq.com");
```

5. 删除信息

```
public static void deleteColumn(String tableName, String rowKey, String family){
    try{
        Table table = conn.getTable(TableName.valueOf(tableName));
        List<Delete> list = new ArrayList<Delete>();
        Delete del = new Delete(rowKey.getBytes());
        del.deleteFamily(Bytes.toBytes(family));
        list.add(del);
        table.delete(list);
        System.out.println("Del record:"+rowKey+"-"+family);
    }catch (IOException e){
        e.printStackTrace();
    }
}
```

删除学号为2015003的学生的选课记录

```
HbaseController.deleteColumn("Student_info","2015003","Math");
HbaseController.deleteColumn("Student_info","2015003","Computer Science");
HbaseController.deleteColumn("Student_info","2015003","English");
```

6. 删除表

```
public static void deleteTable(String tableName) throws IOException {
    init();
    TableName tn = TableName.valueOf(tableName);
    if (admin.tableExists(tn)) {
        admin.disableTable(tn);
        admin.deleteTable(tn);
    }
    close();
    System.out.println("删除了表："+tableName);
}
```

```
HbaseController.deleteTable("Student_info");
```

7. 完整代码

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.*;
import org.apache.hadoop.hbase.client.*;
import org.apache.hadoop.hbase.util.Bytes;
import org.apache.hadoop.conf.Configuration;
import java.io.IOException;
import java.util.Scanner;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class HbaseController {
    private static Configuration conf = null;
    private static Connection conn = null;
    static {
        conf = HbaseConfiguration.create();
        conf.set("hbase.zookeeper.quorum","localhost");
        conf.set("hbase.zookeeper.property.clientPort","2181");
    }

    public static void createTable(String tableName, String[] familys) throws Exception{
        HBaseAdmin admin = new HBaseAdmin(conf);
        if (admin.tableExists(tableName)) {
            System.out.println("table"+tableName+"already exists!");
        }else {
            HTableDescriptor tableDesc = new HTableDescriptor(tableName);
            for(int i=0;i < familys.length;i++)
            {
```

```

        tableDesc.addFamily(new HColumnDescriptor(familys[i]));
    }
    admin.createTable(tableDesc);
    System.out.println("create table" + tableName + " success!");
}

}

public static void addData (String tableName, String rowKey, String family, String
qualifier, String value)
    throws Exception{
    try{
        HTable table = new HTable(conf, tableName);
        Put put = new Put(Bytes.toBytes(rowkey));
        put.add(Bytes.toBytes(family),Bytes.toBytes(qualifier),Bytes.toBytes(value));
        table.put(put);
        System.out.println("insert record success!");
    } catch (IOException e){
        e.printStackTrace();
    }
}

public static void scanByFilter(String tableName, String family, String qualifer, String
value)
    throws IOException{
    HTable table = new HTable(conf, tableName);
    Scan scan = new Scan();
    scan.addColumn(Bytes.toBytes(family), Byte.toBytes(qualifer));
    Filter filter = new
SingleColumnValueFilter(Bytes.toBytes(family),Bytes.toBytes(qualifer),CompareOp.EQUAL,
Bytes.toBytes(value));
    scan.setFilter(filter);
    ResultScanner result = table.getScanner(scan);
    for(Result r: result) {
        for(KeyValue kv : r.raw()) {
            if (new String(kv.getQualifier()).equals(SC_Score)){
                System.out.println("value:" + new String(kv.getValue()));
            }
        }
    }
}

public static void addColumn(String tablename, String Columnname){
    Admin adminManger = conn.getAdmin();
    adminManger.disableTable(tablename);
    HTableDescriptor htd = adminManger.getTableDescriptor(tablename);
    HColumnDescriptor f = new HColumnDescriptor(Columnname); //新增列族
    f.setTimeToLive(TIME_TO_LIVE);
    htd.addFamily(f);
    adminManger.addColumn(tablename, f);
    adminManger.enableTableAsync(tablename);
}

public static void deleteColumn(String tableName, String rowKey, String family){
    try{
        Table table = conn.getTable(TableName.valueOf(tableName));
        List<Delete> list = new ArrayList<Delete>();
        Delete del = new Delete(rowKey.getBytes());
        del.deleteFamily(Bytes.toBytes(family));
        list.add(del);
        table.delete(list);
        System.out.println("Del record:"+rowKey+"-"+family);
    }
}

```

```

        }catch (IOException e){
            e.printStackTrace();
        }
    }

    public static void deleteTable(String tableName) throws IOException {
        init();
        TableName tn = TableName.valueOf(tableName);
        if (admin.tableExists(tn)) {
            admin.disableTable(tn);
            admin.deleteTable(tn);
        }
        close();
        System.out.println("删除了表: "+tableName);
    }

    public static void main(String [] args)
    {
        try {
            String tablename = "Student_info";
            String[] familys = {"student", "Math", "Computer Science", "English"};
            HbaseController.createTable(tablename, familys);
            // 添加学生信息
            HbaseController.addData("Student_info", "2015001", "student", "S_No", "2015001");
            HbaseController.addData("Student_info", "2015001", "student", "S_Name", "Li Lei");
            HbaseController.addData("Student_info", "2015001", "student", "S_Sex", "male");
            HbaseController.addData("Student_info", "2015001", "student", "S_Age", "23");

            HbaseController.addData("Student_info", "2015002", "student", "S_No", "2015002");
            HbaseController.addData("Student_info", "2015002", "student", "S_Name", "Han
Meimei");
            HbaseController.addData("Student_info", "2015002", "student", "S_Sex", "female");
            HbaseController.addData("Student_info", "2015002", "student", "S_Age", "22");

            HbaseController.addData("Student_info", "2015003", "student", "S_No", "2015003");
            HbaseController.addData("Student_info", "2015003", "student", "S_Name", "Zhang San");
            HbaseController.addData("Student_info", "2015003", "student", "S_Sex", "male");
            HbaseController.addData("Student_info", "2015003", "student", "S_Age", "24");

            HbaseController.addData("Student_info", "2015001", "Math", "C_No", "123001");
            HbaseController.addData("Student_info", "2015001", "Math", "C_Credit", "2");
            HbaseController.addData("Student_info", "2015001", "Math", "SC_Score", "86");
            HbaseController.addData("Student_info", "2015001", "Math", "C_Name", "Math");

            HbaseController.addData("Student_info", "2015001", "English", "C_No", "123003");
            HbaseController.addData("Student_info", "2015001", "English", "C_Credit", "3");
            HbaseController.addData("Student_info", "2015001", "English", "SC_Score", "69");
            HbaseController.addData("Student_info", "2015001", "English", "C_Name", "English");

            HbaseController.addData("Student_info", "2015001", "English", "C_No", "123003");
            HbaseController.addData("Student_info", "2015001", "English", "C_Credit", "3");
            HbaseController.addData("Student_info", "2015001", "English", "SC_Score", "69");
            HbaseController.addData("Student_info", "2015001", "English", "C_Name", "English");

            HbaseController.addData("Student_info", "2015002", "Computer
Science", "C_No", "123002");
            HbaseController.addData("Student_info", "2015002", "Computer
Science", "C_Credit", "5");
            HbaseController.addData("Student_info", "2015002", "Computer
Science", "SC_Score", "77");

```

```

        HbaseController.addData("Student_info", "2015002", "Computer
Science", "C_Name", "Computer Science");

        HbaseController.addData("Student_info", "2015002", "English", "C_No", "123003");
        HbaseController.addData("Student_info", "2015002", "English", "C_Credit", "3");
        HbaseController.addData("Student_info", "2015002", "English", "SC_Score", "99");
        HbaseController.addData("Student_info", "2015002", "English", "C_Name", "English");

        HbaseController.addData("Student_info", "2015003", "Math", "C_No", "123001");
        HbaseController.addData("Student_info", "2015003", "Math", "C_Credit", "2");
        HbaseController.addData("Student_info", "2015003", "Math", "SC_Score", "98");
        HbaseController.addData("Student_info", "2015003", "Math", "C_Name", "Math");

        HbaseController.addData("Student_info", "2015003", "Computer
Science", "C_No", "123003");
        HbaseController.addData("Student_info", "2015003", "Computer
Science", "C_Credit", "5");
        HbaseController.addData("Student_info", "2015003", "Computer
Science", "SC_Score", "95");
        HbaseController.addData("Student_info", "2015003", "Computer
Science", "C_Name", "Computer Science");

        HbaseController.scanByFilter("Student_info", "Computer Science", "C_Name", "Computer
Science");

        HbaseController.addColumn("Student_info", "Contact");

        HbaseController.addData("Student_info", "2015001", "Contact", "S_Email", "lilie@qq.com");

        HbaseController.addData("Student_info", "2015002", "Contact", "S_Email", "hmm@qq.com");

        HbaseController.addData("Student_info", "2015003", "Contact", "S_Email", "zs@qq.com");

        HbaseController.deleteColumn("Student_info", "2015003", "Math");
        HbaseController.deleteColumn("Student_info", "2015003", "Computer Science");
        HbaseController.deleteColumn("Student_info", "2015003", "English");

        HbaseController.deleteTable("Student_info");
    }catch (Exception e){
        e.printStackTrace();
    }
}
}

```

四. Shell完成操作

这一部分在windows下安装的Hbase下进行操作

1. 初始化

```

C:\WINDOWS\system32\cmd.exe - hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/D:/hbase-1.2.0-bin/hbase-1.2.0/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/D:/Hadoop/hadoop-2.9.1/share/hadoop/common/lib/slf4j-log4j12-1.7.25.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 1.2.0, r25b281972df2f5b15c426c8963cbf77dd853a5ad, Thu Feb 18 23:01:49 CST 2016

hbase(main):001:0> list
TABLE

0 row(s) in 0.3410 seconds

=> []
hbase(main):002:0> create 'Student_info', 'Student', 'Math', 'Computer Science', 'English'
0 row(s) in 1.3450 seconds

=> Hbase::Table - Student_info
hbase(main):003:0> list
TABLE

Student_info

1 row(s) in 0.0160 seconds

=> ["Student_info"]
hbase(main):004:0>

```

```

C:\WINDOWS\system32\cmd.exe - hbase shell
hbase(main):003:0> list
TABLE

Student_info

1 row(s) in 0.0160 seconds

=> ["Student_info"]
hbase(main):004:0> put 'Student_info', '2015001', 'Student:S_Name', 'Li Lei'
0 row(s) in 0.1890 seconds

hbase(main):005:0> put 'Student_info', '2015001', 'Student:S_Sex', 'male'
0 row(s) in 0.0160 seconds

hbase(main):006:0> put 'Student_info', '2015001', 'Student:S_Age', '23'
0 row(s) in 0.0000 seconds

hbase(main):007:0> put 'Student_info', '2015002', 'Student:S_Name', 'Han Meimei'
0 row(s) in 0.0000 seconds

hbase(main):008:0> put 'Student_info', '2015002', 'Student:S_Sex', 'female'
0 row(s) in 0.0000 seconds

hbase(main):009:0> describe 'Student_info'
Table Student_info is ENABLED

Student_info

COLUMN FAMILIES DESCRIPTION

```

(不知为何这边用时变成了0，有点害怕，不过查看了一下没问题，于是继续)


```
C:\WINDOWS\system32\cmd.exe - hbase shell
4 row(s) in 0.0940 seconds
hbase(main):010:0> scan 'Student_info'
ROW COLUMN+CELL
2015001 column=Student:S_Age, timestamp=1638287123854, value=23
2015001 column=Student:S_Name, timestamp=1638287057495, value=Li Lei
2015001 column=Student:S_Sex, timestamp=1638287106337, value=male
2015002 column=Student:S_Name, timestamp=1638287155233, value=Han Meimei
2015002 column=Student:S_Sex, timestamp=1638287171996, value=female
2 row(s) in 0.0470 seconds
hbase(main):011:0> put 'Student_info', '2015003', 'Student:S_Age', '22'
0 row(s) in 0.0000 seconds
hbase(main):012:0> put 'Student_info', '2015002', 'Student:S_Age', '22'
0 row(s) in 0.0160 seconds
hbase(main):013:0> put 'Student_info', '2015003', 'Student:S_Name', 'Zhang San'
0 row(s) in 0.0000 seconds
hbase(main):014:0> put 'Student_info', '2015003', 'Student:S_Sex', 'male'
0 row(s) in 0.0000 seconds
hbase(main):015:0> put 'Student_info', '2015003', 'Student:S_Age', '24'
```

完成

```
C:\WINDOWS\system32\cmd.exe - hbase shell
hbase(main):049:0> scan 'Student_info'
ROW COLUMN+CELL
2015001 column=Computer Science:C_Credit, timestamp=1638288057011, value=5
2015001 column=Computer Science:C_Name, timestamp=1638288101023, value=Computer Science
2015001 column=Computer Science:C_No, timestamp=1638288022200, value=123002
2015001 column=English:C_Credit, timestamp=1638288287207, value=3
2015001 column=English:C_Name, timestamp=1638288324475, value=English
2015001 column=English:C_No, timestamp=1638288206974, value=123003
2015001 column=English:SC_Score, timestamp=1638288377745, value=69
2015001 column=Math:C_Credit, timestamp=1638287820982, value=2
2015001 column=Math:C_Name, timestamp=1638287863552, value=Math
2015001 column=Math:C_No, timestamp=1638287779943, value=123001
2015001 column=Math:SC_Score, timestamp=1638287907276, value=86
2015001 column=Student:S_Age, timestamp=1638287123854, value=23
2015001 column=Student:S_Name, timestamp=1638287057495, value=Li Lei
2015001 column=Student:S_Sex, timestamp=1638287106337, value=male
2015002 column=Computer Science:C_Credit, timestamp=1638288068058, value=5
2015002 column=Computer Science:C_Name, timestamp=1638288113151, value=Computer Science
2015002 column=Computer Science:C_No, timestamp=1638288012030, value=123002
2015002 column=Computer Science:SC_Score, timestamp=1638288155747, value=77
2015002 column=English:C_Credit, timestamp=1638288298857, value=3
2015002 column=English:C_Name, timestamp=163828833882, value=English
2015002 column=English:C_No, timestamp=1638288216039, value=123003
2015002 column=English:SC_Score, timestamp=1638288390559, value=99
2015002 column=Math:C_Credit, timestamp=1638287831879, value=2
2015002 column=Math:C_Name, timestamp=1638287876289, value=Math
2015002 column=Math:C_No, timestamp=1638287790195, value=123001
2015002 column=Student:S_Age, timestamp=1638287526904, value=22
2015002 column=Student:S_Name, timestamp=1638287155233, value=Han Meimei
2015002 column=Student:S_Sex, timestamp=1638287171996, value=female
2015003 column=Computer Science:C_Credit, timestamp=1638288074527, value=5
2015003 column=Computer Science:C_Name, timestamp=1638288120027, value=Computer Science
2015003 column=Computer Science:C_No, timestamp=1638287995380, value=123002
2015003 column=Computer Science:SC_Score, timestamp=1638288175414, value=95
2015003 column=English:C_Credit, timestamp=1638288304966, value=3
2015003 column=English:C_Name, timestamp=1638288338945, value=English
2015003 column=English:C_No, timestamp=163828822462, value=123003
2015003 column=Math:C_Credit, timestamp=1638287883915, value=2
2015003 column=Math:C_Name, timestamp=1638287883915, value=Math
2015003 column=Math:C_No, timestamp=1638287797789, value=123001
2015003 column=Math:SC_Score, timestamp=1638287952012, value=98
```

2. 查询Computer Science的学生的成绩

```
hbase(main):050:0> scan 'Student_info', {COLUMN=>'Computer Science:SC_Score'}
ROW COLUMN+CELL
2015002 column=Computer Science:SC_Score, timestamp=1638288155747, value=77
2015003 column=Computer Science:SC_Score, timestamp=1638288175414, value=95
2 row(s) in 0.0160 seconds
```

3. 增加新的列族和新列Contact:Email, 并添加数据

遇到报错, 发现自己是输错了, 卡在那里, 通过>指令退出重来


```

hbase(main):051:0> alter 'Student_info', Contact'
hbase(main):052:0'
hbase(main):053:0' >
hbase(main):054:0' >'
SyntaxError: (hbase):51: syntax error, unexpected tSTRING_BEG
alter 'Student_info', Contact'
hbase(main):055:0>

```

完成

```

hbase(main):056:0> alter 'Student_info', 'Contact'
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9730 seconds

hbase(main):057:0> put 'Student_info', '2015001', 'Contact:S_Email', 'lilie@qq.com'
0 row(s) in 0.0000 seconds

hbase(main):058:0> put 'Student_info', '2015002', 'Contact:S_Email', 'hmm@qq.com'
0 row(s) in 0.0000 seconds

hbase(main):059:0> put 'Student_info', '2015003', 'Contact:S_Email', 'zs@qq.com'
0 row(s) in 0.0000 seconds

```

(遇到奇奇怪怪的问题，重新打开)

```

hbase(main):003:0> scan 'Student_info', {COLUMN=>'Contact:S_Email'}
ROW
COLUMN+CELL
2015001      column=Contact:S_Email, timestamp=1638289199213, value=lilie@qq.com
2015002      column=Contact:S_Email, timestamp=1638289225835, value=hmm@qq.com
2015003      column=Contact:S_Email, timestamp=1638289259321, value=zs@qq.com
3 row(s) in 0.0470 seconds

```

4. 删除学号为2015003的学生的选课记录

```

hbase(main):013:0> delete 'Student_info', '2015003', 'English:C_No'
0 row(s) in 0.0000 seconds

hbase(main):014:0> delete 'Student_info', '2015003', 'English:C_Name'
0 row(s) in 0.0000 seconds

hbase(main):015:0> delete 'Student_info', '2015003', 'English:C_Credit'
0 row(s) in 0.0150 seconds

hbase(main):016:0> get 'Student_info', '2015003'
COLUMN      CELL
Contact:S_Email      timestamp=1638289259321, value=zs@qq.com
Student:S_Age        timestamp=1638287695841, value=24
Student:S_Name       timestamp=1638287557973, value=Zhang San
Student:S_Sex        timestamp=1638287593435, value=male
4 row(s) in 0.0160 seconds

```

5. 删除所创建的表

```

hbase(main):017:0> disable 'Student_info'
0 row(s) in 2.3300 seconds

hbase(main):018:0> drop 'Student_info'
0 row(s) in 1.2980 seconds

hbase(main):019:0> exists 'Student_info'
Table Student_info does not exist

0 row(s) in 0.0150 seconds

```

五. 遇到问题

```

hbase(main):060:0> scan 'Student_info', 'COLUMN=>' Contact:S_Email'
hbase(main):061:0> >
SyntaxError: (hbase):60: syntax error, unexpected tCONSTANT

scan 'Student_info', 'COLUMN=>' Contact:S_Email'

hbase(main):062:0> scan 'Student_info', {'COLUMN=>' Contact:S_Email'}
hbase(main):063:1> >
hbase(main):064:1> scan 'Student_info', {COLUMN=>' Contact:S_Email'}
hbase(main):065:1> >
hbase(main):066:1> >
hbase(main):067:1> scan 'Student_info', {COLUMN=>' Contact:S_Email'}
hbase(main):068:1> scan 'Student_info'
hbase(main):069:1> >+

```

这边输错了一次，然后就进入了1，之前做的都没有了，不知为何会这样，网上没有找到解决方案，只好重来；所幸重新进来之后，表格还在，于是继续操作

```

C:\WINDOWS\system32\cmd.exe - hbase shell
HBase Shell: enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0, r25b281972df2f5b15c426c8963cbf77dd853a5ad, Thu Feb 18 23:01:49 CST 2016

hbase(main):001:0> list
TABLE
Student_info
1 row(s) in 0.2500 seconds

=> ["Student_info"]
hbase(main):002:0> scan 'Student_info'
ROW COLUMN+CELL
2015001 column=Computer Science:C_Credit, timestamp=1638288057011, value=5
2015001 column=Computer Science:C_Name, timestamp=1638288101023, value=Computer Science
2015001 column=Computer Science:C_No, timestamp=1638288022200, value=123002
2015001 column=Contact:S_Email, timestamp=1638289199213, value=lilie@qq.com
2015001 column=English:C_Credit, timestamp=1638288287207, value=3
2015001 column=English:C_Name, timestamp=1638288324475, value=English
2015001 column=English:C_No, timestamp=1638288206974, value=123003
2015001 column=English:SC_Score, timestamp=1638288377745, value=69
2015001 column=Math:C_Credit, timestamp=1638287820982, value=2
2015001 column=Math:C_Name, timestamp=1638287863552, value=Math
2015001 column=Math:C_No, timestamp=1638287779943, value=123001
2015001 column=Math:SC_Score, timestamp=1638287907276, value=86
2015001 column=Student:S_Age, timestamp=1638287123854, value=23
2015001 column=Student:S_Name, timestamp=1638287057495, value=Li Lei
2015001 column=Student:S_Sex, timestamp=1638287106337, value=male
2015002 column=Computer Science:C_Credit, timestamp=1638288068058, value=5
2015002 column=Computer Science:C_Name, timestamp=1638288113151, value=Computer Science
2015002 column=Computer Science:C_No, timestamp=1638288012030, value=123002
2015002 column=Computer Science:SC_Score, timestamp=1638288155747, value=77
2015002 column=Contact:S_Email, timestamp=1638289225835, value=hmm@qq.com
2015002 column=English:C_Credit, timestamp=1638288298857, value=3
2015002 column=English:C_Name, timestamp=1638288333882, value=English
2015002 column=English:C_No, timestamp=1638288216039, value=123003
2015002 column=English:SC_Score, timestamp=1638288390559, value=99
2015002 column=Math:C_Credit, timestamp=1638287831879, value=2
2015002 column=Math:C_Name, timestamp=1638287876289, value=Math
2015002 column=Math:C_No, timestamp=1638287790195, value=123001
2015002 column=Student:S_Age, timestamp=1638287526904, value=22
2015002 column=Student:S_Name, timestamp=1638287155233, value=Han Meimei

```