

**计算机与网络空间安全学院学生实验报告**

**实验课程名称：** 大数据导论 **教师： 林鑫泓 \_\_**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | 熟悉常用的HBase操作 | | | **实验成绩** |  |
| **学生姓名** | **叶建行** | **学 号** | **116052020005** | **年级专业班级** | **2020级软件工程一班** |
| **小组成员** | **无** | | | **实验日期** | **2022年10月21** |

# 1 实验目的和要求

## 1.1 实验目的

* 理解HBase在Hadoop体系结构中的角色，
* 熟练使用HBase操作常用的Shell命令，
* 熟悉HBase操作常用的Java API。

## 1.2 实验软硬件环境

* 操作系统：Linux（建议Ubuntu18.04），
* Hadoop版本：建议3.2.1，
* HBase版本：建议2.2.6，
* JDK版本：openjdk-8。

## 1.3 实验要求

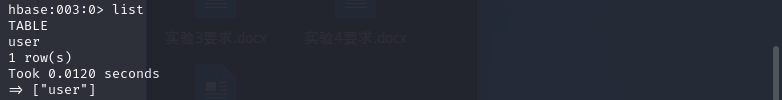
* 理解HBase在Hadoop体系结构中的角色，
* 熟练使用HBase操作常用的Shell命令，
* 熟悉HBase操作常用的Java API。

# 2 实验记录

## 2.1 Linux操作

（一）编程实现以下指定功能，并用Hadoop提供的HBase Shell命令完成相同任务：

1. 列出HBase所有的表的相关信息，例如表名；



**实验java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 java.io.IOException;

import java.util.List;

public class list {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void listTables() throws IOException {

        init();

        List<TableDescriptor> tableDescriptors = admin.listTableDescriptors();

        for(TableDescriptor tableDescriptor : tableDescriptors){

            TableName tableName = tableDescriptor.getTableName();

            System.out.println("表名: " + tableName);

        }

        close();

    }

    public static void main(String[] args) {

        try {

            listTables();

        } catch (IOException e) {

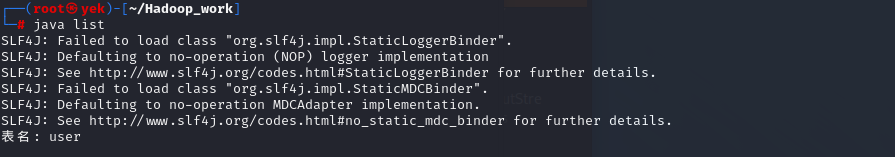
            throw new RuntimeException(e);

        }

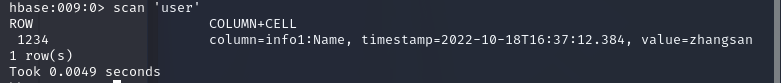
    }

}

**实验结果：**

****

1. 在终端打印出指定的表的所有记录数据；



**实验Java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 java.io.IOException;

import java.util.List;

public class scan {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

    public static void init() {

        configuration = HBaseConfiguration.create();

        configuration.set("hbase.rootdir", "hdfs://localhost:9000/hbase");

        try {

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    public static void close() {

        try {

            if (admin != null) {

                admin.close();

            }

            if (null != connection) {

                connection.close();

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    public static void getData(String tableName) throws IOException {

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Scan scan = new Scan();

        ResultScanner scanner = table.getScanner(scan);

        for (Result result : scanner) {

            printRecoder(result);

        }

        close();

    }

 public  static void printRecoder(Result result)throws IOException{

     for(Cell cell:result.rawCells()){

            System.out.print(行键: "+new String(Bytes.toString(cell.getRowArray(),cell.getRowOffset(), cell.getRowLength())));

            System.out.print(" 列簇: "+new String(Bytes.toString(cell.getFamilyArray(),cell.getFamilyOffset(), cell.getFamilyLength()) ));

            System.out.print(" 列: "+new String(Bytes.toString(cell.getQualifierArray(),cell.getQualifierOffset(), cell.getQualifierLength())));

            System.out.print(" 值: "+new String(Bytes.toString(cell.getValueArray(),cell.getValueOffset(), cell.getValueLength())));

            System.out.println(" 时间戳: "+cell.getTimestamp());

        }

    }

    public static void main(String[] args) {

        try {

            getData("user");

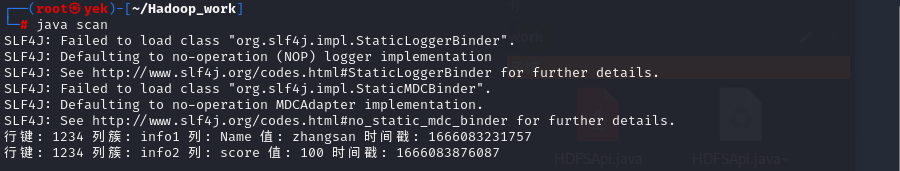
        } catch (IOException e) {

            throw new RuntimeException(e);

        }

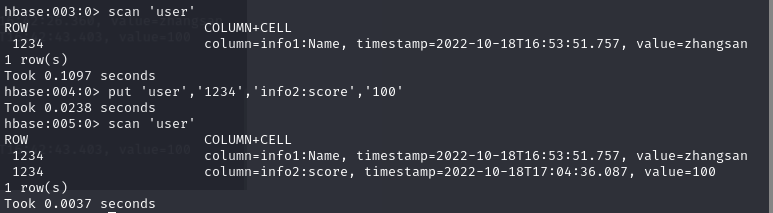
    }

}

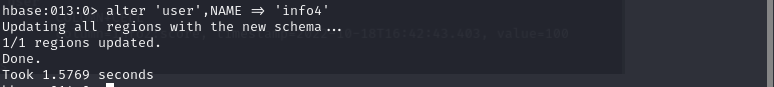
**实验结果：**

1. 向已经创建好的表添加和删除指定的列族或列；

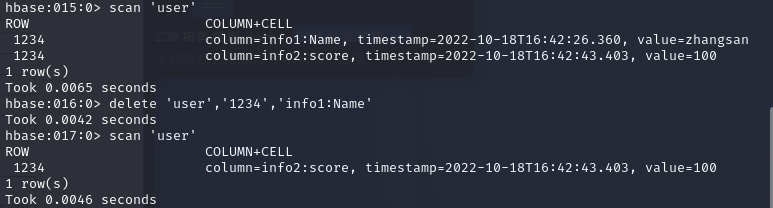
**添加指定列：**



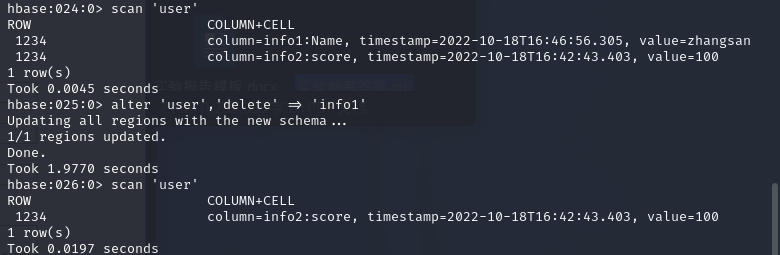
**添加指定列族：**

****

**删除指定列：**



**删除指定列族：**



**实验Java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 java.io.IOException;

import java.util.List;

public class addOrdelete {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

     public static void insertRow(String tableName,String rowKey,String colFamily,String col,String val) throws IOException {

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Put put = new Put(rowKey.getBytes());

        put.addColumn(colFamily.getBytes(), col.getBytes(), val.getBytes());

        table.put(put);

        table.close();

        close();

    }

    public static void deleteRow(String tableName,String rowKey,String colFamily,String col) throws IOException {

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Delete delete = new Delete(rowKey.getBytes());

        delete.addFamily(Bytes.toBytes(colFamily));

        delete.addColumn(Bytes.toBytes(colFamily),Bytes.toBytes(col));

        table.delete(delete);

        table.close();

        close();

    }

    public static void getData(String tableName)throws  IOException{

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Scan scan = new Scan();

        ResultScanner scanner = table.getScanner(scan);

        for (Result result:scanner){

            printRecoder(result);

        }

        close();

    }

    public  static void printRecoder(Result result)throws IOException{

        for(Cell cell:result.rawCells()){

            System.out.print("行键: "+new String(Bytes.toString(cell.getRowArray(),cell.getRowOffset(), cell.getRowLength())));

            System.out.print(" 列簇: "+new String( Bytes.toString(cell.getFamilyArray(),cell.getFamilyOffset(), cell.getFamilyLength()) ));

            System.out.print(" 列: "+new String(Bytes.toString(cell.getQualifierArray(),cell.getQualifierOffset(), cell.getQualifierLength())));

            System.out.print(" 值: "+new String(Bytes.toString(cell.getValueArray(),cell.getValueOffset(), cell.getValueLength())));

            System.out.println(" 时间戳: "+cell.getTimestamp());

        }

    }

    public static void main(String[] args) {

        try {

            getData("user");

            insertRow("user","lihua","info2","score","99");

            System.out.println("插入成功: ");

            getData("user");

            deleteRow("user","lihua","info2","score");

            System.out.println("删除成功: ");

            getData("user");

        } catch (IOException e) {

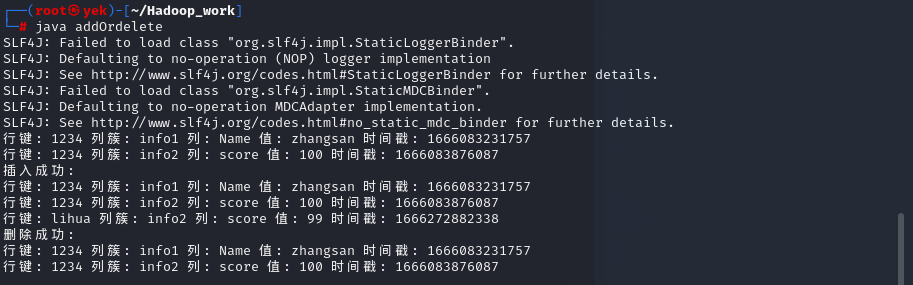
            throw new RuntimeException(e);

        }

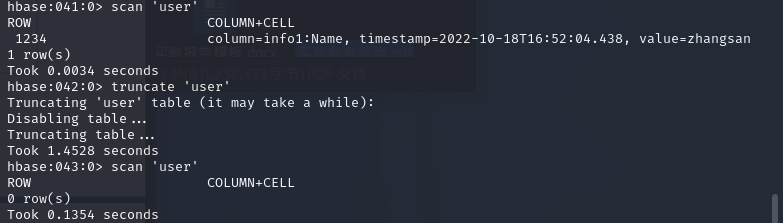
    }

}

**实验结果：**



1. 清空指定的表的所有记录数据；



**实验Java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 java.io.IOException;

import java.util.List;

import java.util.ArrayList;

public class truncate {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

      public static void clearRows(String tableName)throws IOException{

        init();

        TableName tablename = TableName.valueOf(tableName);

        Table table = connection.getTable(tablename);

        HTableDescriptor tableDescriptor1 = table.getTableDescriptor();

        HColumnDescriptor[] columnFamilies = tableDescriptor1.getColumnFamilies();

*// 创建集合用于存放ColumnFamilyDescriptor对象*

        List<ColumnFamilyDescriptor> families = new ArrayList<>();

*// 将每个familyName对应的ColumnFamilyDescriptor对象添加到families集合中保存*

        for (HColumnDescriptor columnFamily : columnFamilies){

              byte[] name = columnFamily.getName();

              String value = Bytes.toString(name);

              families.add(ColumnFamilyDescriptorBuilder.newBuilder(value.getBytes()).build());

        }

        admin.disableTable(tablename);

        admin.deleteTable(tablename);

        TableDescriptorBuilder tableDescriptor = TableDescriptorBuilder.newBuilder(tablename);

        admin.createTable(tableDescriptor.setColumnFamilies(families).build());

        close();

    }

    public static void main(String[] args) {

        try {

            clearRows("user");

            System.out.println("已全部删除");

        } catch (IOException e) {

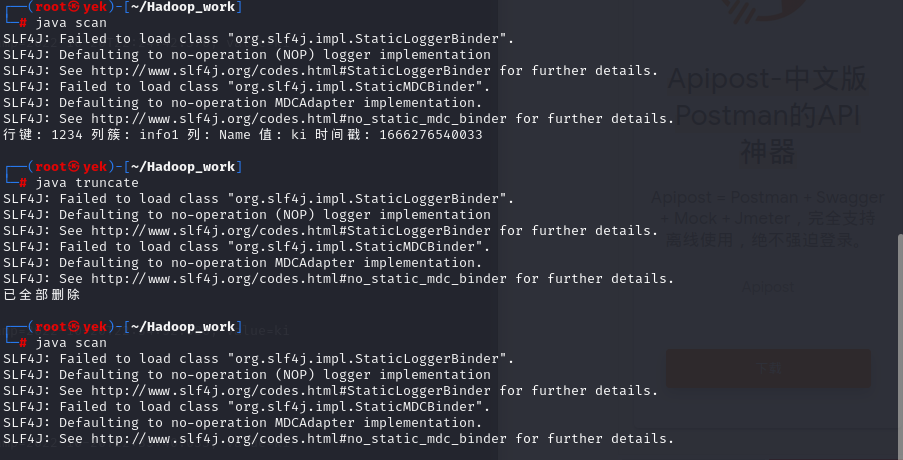
            throw new RuntimeException(e);

        }

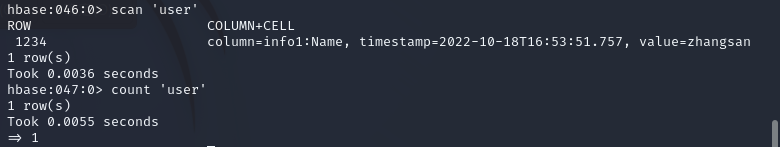
    }

}

**实验结果：**

****

1. 统计表的行数。



**实验Java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 java.io.IOException;

import java.util.List;

import java.util.ArrayList;

public class count {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

       public static void countRows(String tableName)throws IOException{

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Scan scan = new Scan();

        ResultScanner scanner = table.getScanner(scan);

        int num = 0;

        for (Result result = scanner.next();result!=null;result=scanner.next()){

            num++;

        }

        System.out.println("行数:"+ num);

        scanner.close();

        close();

    }

    public static void main(String[] args) {

        try {

            countRows("user");

        } catch (IOException e) {

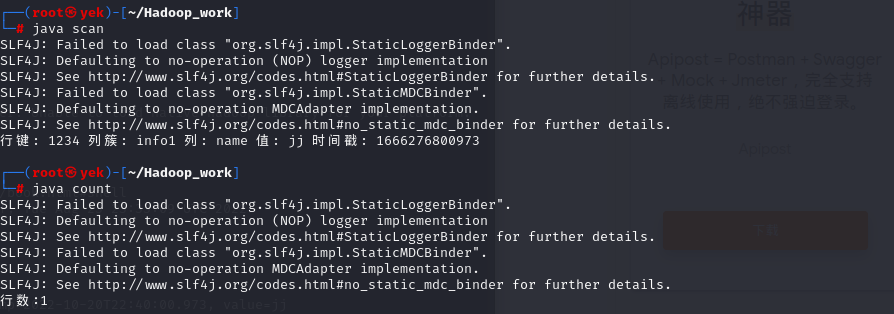
            throw new RuntimeException(e);

        }

    }

}

**实验结果：**



（二）HBase数据库操作

1. 现有以下关系型数据库中的表和数据，要求将其转换为适合于HBase存储的表并插入数据：

学生表（Student）

|  |  |  |  |
| --- | --- | --- | --- |
| 学号（S\_No） | 姓名（S\_Name） | 性别（S\_Sex） | 年龄（S\_Age） |
| 2015001 | Zhangsan | male | 23 |
| 2015003 | Mary | female | 22 |
| 2015003 | Lisi | male | 24 |

课程表（Course）

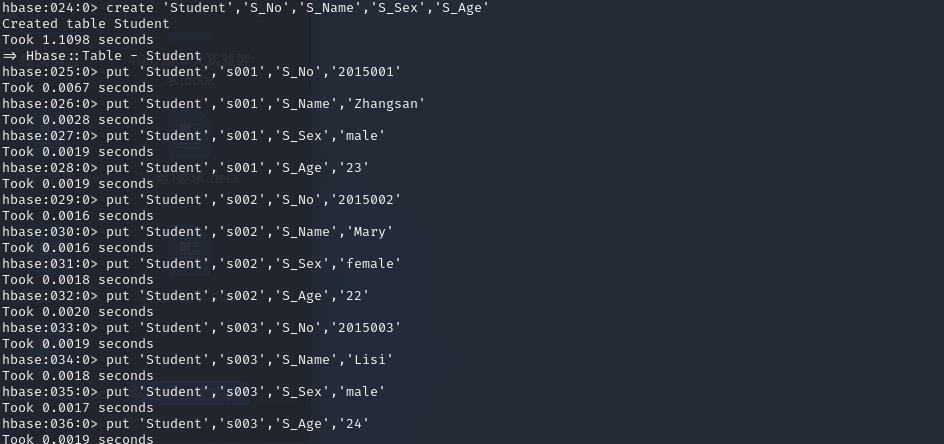
|  |  |  |
| --- | --- | --- |
| 课程号（C\_No） | 课程名（C\_Name） | 学分（C\_Credit） |
| 123001 | Math | 2.0 |
| 123002 | Computer Science | 5.0 |
| 123003 | English | 3.0 |

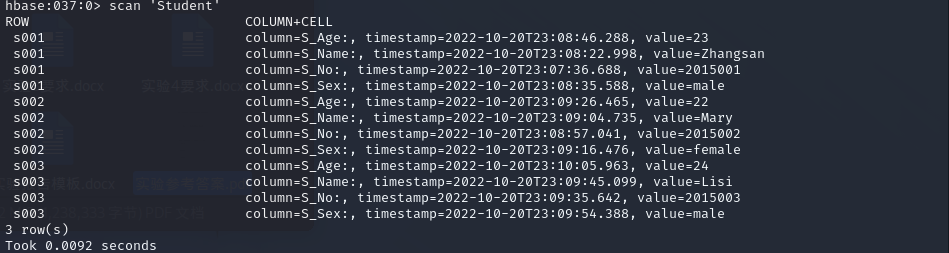
选课表（SC）

|  |  |  |
| --- | --- | --- |
| 学号（SC\_Sno） | 课程号（SC\_Cno） | 成绩（SC\_Score） |
| 2015001 | 123001 | 86 |
| 2015001 | 123003 | 69 |
| 2015002 | 123002 | 77 |
| 2015002 | 123003 | 99 |
| 2015003 | 123001 | 98 |
| 2015003 | 123002 | 95 |

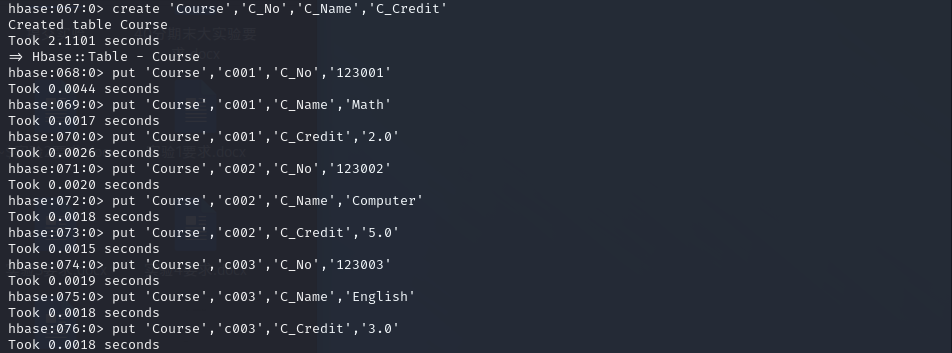
**实验代码及结果：**

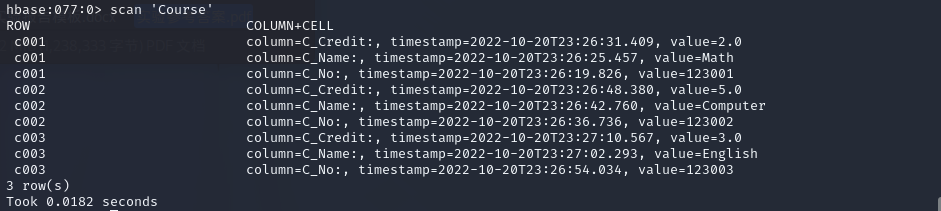
**学生表：**



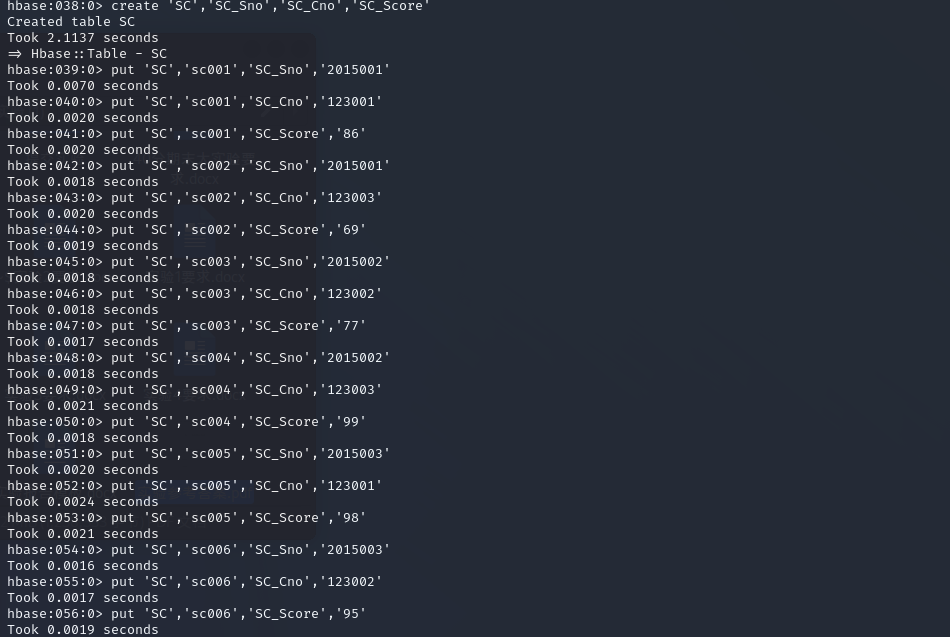


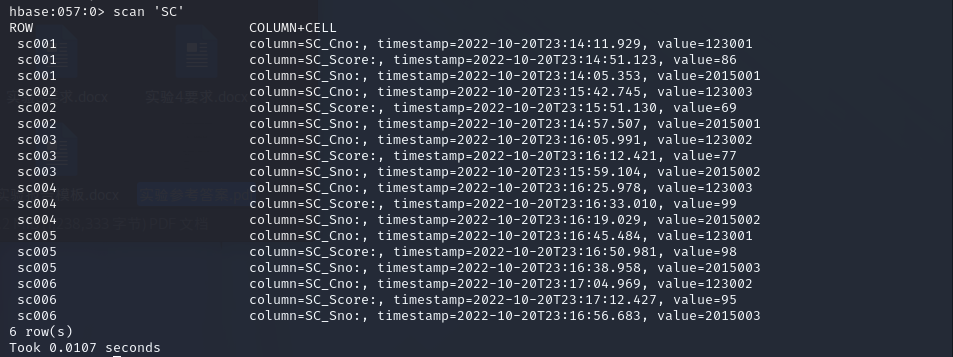
课程表





选课表：





2. 请编程实现以下功能：

（1）createTable(String tableName, String[] fields)

创建表，参数tableName为表的名称，字符串数组fields为存储记录各个字段名称的数组。要求当HBase已经存在名为tableName的表的时候，先删除原有的表，然后再创建新的表。

**实验Java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 java.io.IOException;

import java.util.List;

import java.util.ArrayList;

public class createTable {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

      public static void clearRows(String tableName,String... familyNames)throws IOException{

        init();

        TableName tablename = TableName.valueOf(tableName);

*// 创建集合用于存放ColumnFamilyDescriptor对象*

        List<ColumnFamilyDescriptor> families = new ArrayList<>();

*// 将每个familyName对应的ColumnFamilyDescriptor对象添加到families集合中保存*

        for (String familyName : familyNames)

            families.add(ColumnFamilyDescriptorBuilder.newBuilder(familyName.getBytes()).build());

        if(admin.tableExists(tablename)){

        System.out.println("table is exists!");

        admin.disableTable(tablename);

        admin.deleteTable(tablename);*//删除原来的表*

    }

        TableDescriptorBuilder tableDescriptor = TableDescriptorBuilder.newBuilder(tablename); admin.createTable(tableDescriptor.setColumnFamilies(families).build());

        close();

    }

    public static void main(String[] args) {

        try {

            clearRows("user","info1","info2");

        } catch (IOException e) {

            throw new RuntimeException(e);

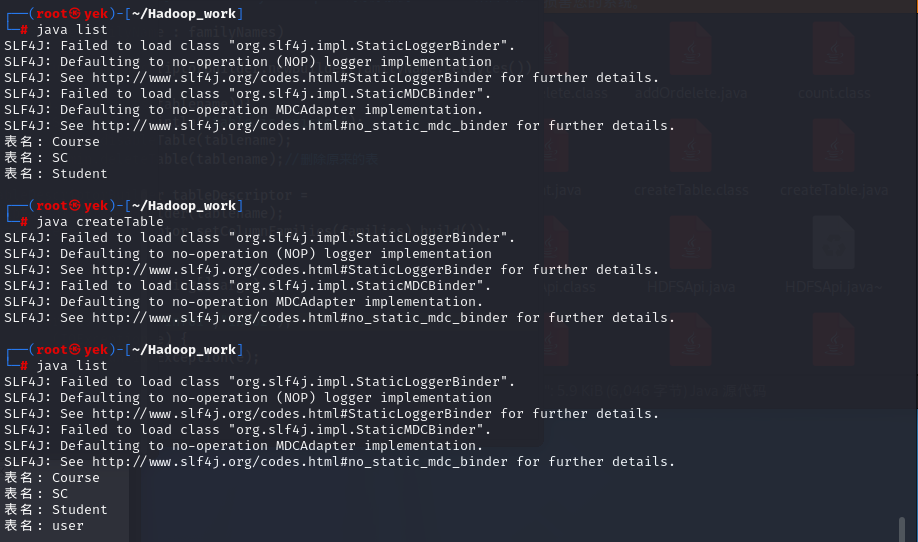
        }

    }

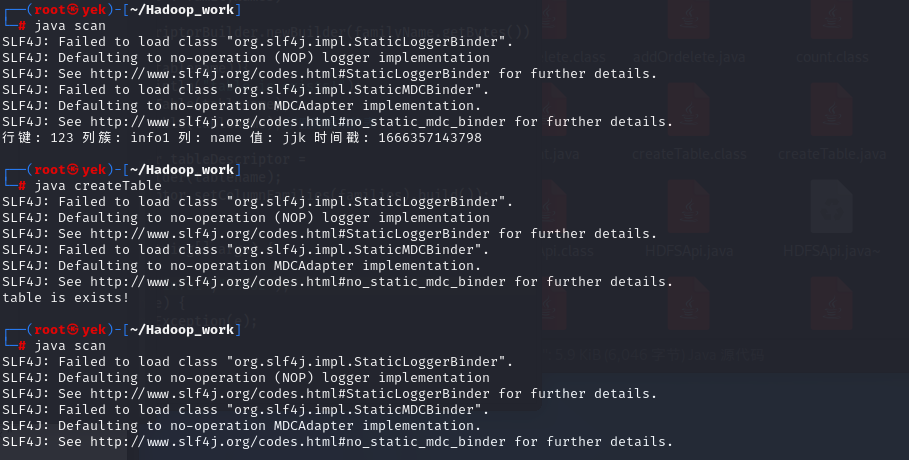
}

**实验结果：**

* + - 1. **表不存在时创建表**

****

**2. 表存在时删除原表，创建新表**

****（2）addRecord(String tableName, String row, String[] fields, String[] values)

向表tableName、行row（用S\_Name表示）和字符串数组fields指定的单元格中添加对应的数据values。其中，fields中每个元素如果对应的列族下还有相应的列限定符的话，用“columnFamily:column”表示。例如，同时向“Math”、“Computer Science”、“English”三列添加成绩时，字符串数组fields为{“Score:Math”, ”Score:Computer Science”, ”Score:English”}，数组values存储这三门课的成绩。

**实验Java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 java.io.IOException;

public class addRecord {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

*//建立连接*

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

*//关闭连接*

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void addRecord(String tableName,String row,String[] fields,String[] values) throws IOException {

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        for(int i = 0;i != fields.length;i++){

            Put put = new Put(row.getBytes());

            String[] cols = fields[i].split(":");

            put.addColumn(cols[0].getBytes(), cols[1].getBytes(), values[i].getBytes());

            table.put(put);

        }

        table.close();

        close();

    }

    public static void main(String[] args) {

        try {

            String[] columns = {"info1:Math", "info1:Computer Science", "info1:English"};

            String[] values = {"90","80","100"};

            addRecord("user","lihua",columns,values);

        } catch (IOException e) {

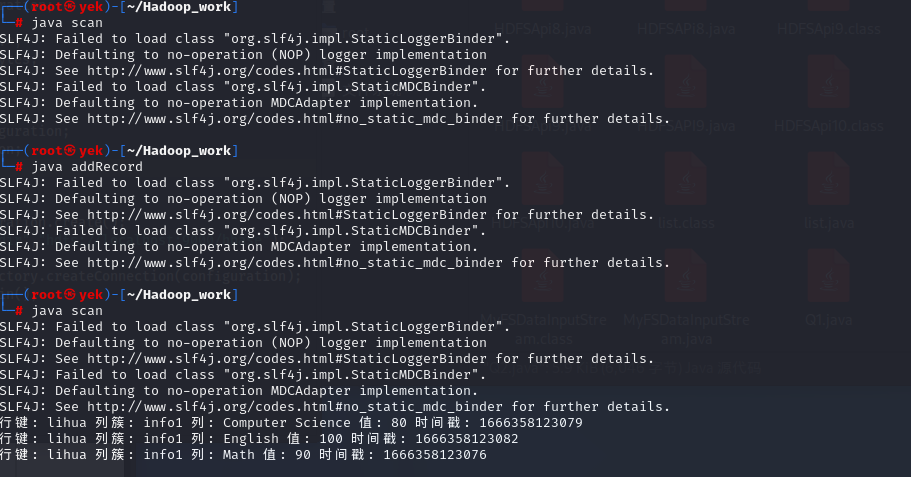
            throw new RuntimeException(e);

        }

    }

}

**实验结果：**



（3）scanColumn(String tableName, String column)

浏览表tableName某一列的数据，如果某一行记录中该列数据不存在，则返回null。要求当参数column为某一列族名称时，如果底下有若干个列限定符，则要列出每个列限定符代表的列的数据；当参数column为某一列具体名称（例如“Score:Math”）时，只需要列出该列的数据。

**实验Java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 java.io.IOException;

public class scanColumn {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

*//建立连接*

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

*//关闭连接*

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

     public static void scanColumn(String tableName,String column)throws  IOException{

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Scan scan = new Scan();

        scan.addFamily(Bytes.toBytes(column));

        ResultScanner scanner = table.getScanner(scan);

        for (Result result = scanner.next(); result != null; result = scanner.next()){

            showCell(result);

        }

        table.close();

        close();

    }

*//格式化输出*

    public static void showCell(Result result){

        Cell[] cells = result.rawCells();

        for(Cell cell:cells){

            System.out.println("RowName:"+new String(Bytes.toString(cell.getRowArray(),cell.getRowOffset(), cell.getRowLength()))+" ");

            System.out.println("Timetamp:"+cell.getTimestamp()+" ");

            System.out.println("column Family:"+new String(Bytes.toString(cell.getFamilyArray(),cell.getFamilyOffset(), cell.getFamilyLength()))+" ");

            System.out.println("row Name:"+new String(Bytes.toString(cell.getQualifierArray(),cell.getQualifierOffset(), cell.getQualifierLength()))+" ");

            System.out.println("value:"+new String(Bytes.toString(cell.getValueArray(),cell.getValueOffset(), cell.getValueLength()))+" ");

        }

    }

    public static void main(String[] args) {

        try {

          scanColumn("user","info1");

        } catch (IOException e) {

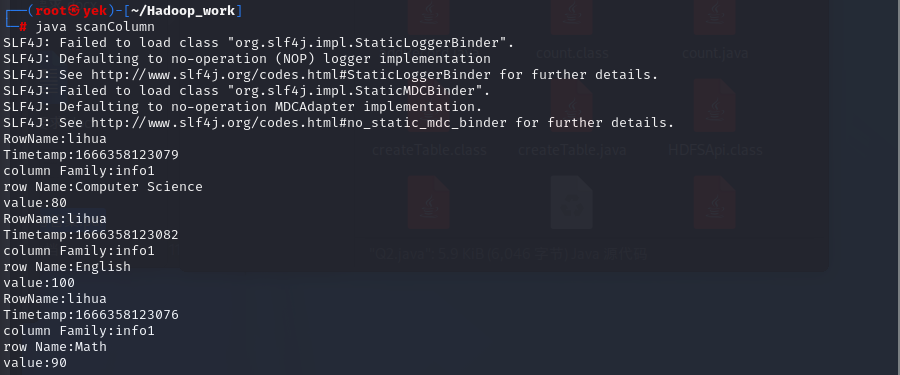
            throw new RuntimeException(e);

        }

    }

}

**实验结果：**

****

（4）modifyData(String tableName, String row, String column)

修改表tableName，行row（可以用学生姓名S\_Name表示），列column指定的单元格的数据。

**实验Java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 java.io.IOException;

public class modifyData {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

*//建立连接*

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

*//关闭连接*

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void modifyData(String tableName,String row,String columnfamily,String column,String val)throws IOException{

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Put put = new Put(row.getBytes());

        put.addColumn(columnfamily.getBytes(),column.getBytes(),val.getBytes());

        table.put(put);

        table.close();

        close();

    }

    public static void main(String[] args) {

        try {

            modifyData("user","lihua","info1","Math","90");

        } catch (IOException e) {

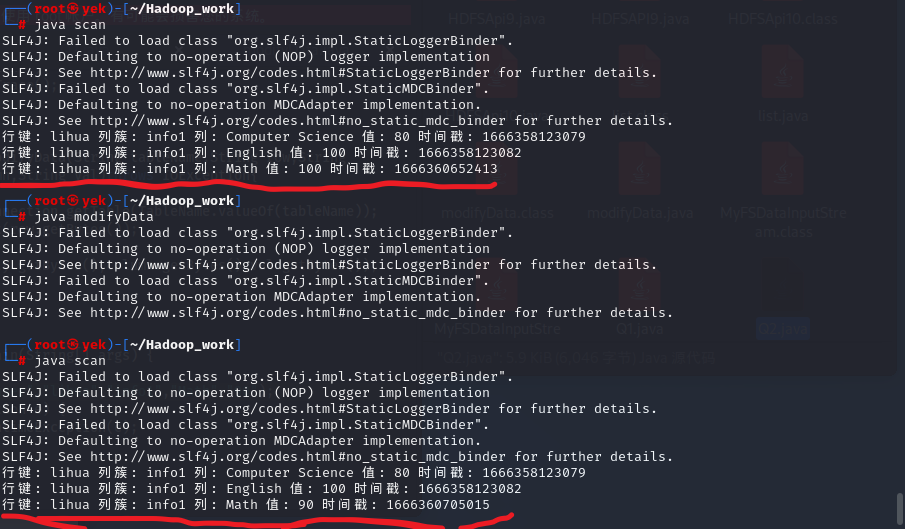
            throw new RuntimeException(e);

        }

    }

}

**实验结果：**



（5）deleteRow(String tableName, String row)

删除表tableName中row指定的行的记录。

**实验Java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 java.io.IOException;

public class deleteRow {

    public static Configuration configuration;

    public static Connection connection;

    public static Admin admin;

*//建立连接*

    public static void init(){

        configuration  = HBaseConfiguration.create();

        configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");

        try{

            connection = ConnectionFactory.createConnection(configuration);

            admin = connection.getAdmin();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

*//关闭连接*

    public static void close(){

        try{

            if(admin != null){

                admin.close();

            }

            if(null != connection){

                connection.close();

            }

        }catch (IOException e){

            e.printStackTrace();

        }

    }

    public static void deleteRow(String tableName,String row)throws IOException{

        init();

        Table table = connection.getTable(TableName.valueOf(tableName));

        Delete delete = new Delete(row.getBytes());

        table.delete(delete);

        table.close();

        close();

    }

    public static void main(String[] args) {

        try {

            deleteRow("user","lihua");

        } catch (IOException e) {

            throw new RuntimeException(e);

        }

    }

}

**实验结果：**



# 3 实验总结

练习使用利用hbase shell 脚本编写 hbase的常用命令，加深了我对 hbase的理解，以及对 hbase 操作的熟练度，能更快的使用这些命令。通过 hbase Api 进行编程操作，加深 了我对 hbase的理解