Hbase接口应用

1. Shell接口

即

```
hbase shell
```

1.1. 创建表

create命令

```
create 'user','uName','uPwd','sex','age'
```

'user'是表名，后面是属性名。注意Hbase是列式存储，不说列名。

1.2. 查看表详细信息

```
describe 'user'
```

1.3. 添加数据

在添加数据时，HBase会自动为添加的数据添加一个时间戳，故在需要修改数据时，只需直接添加数据，HBase即会生成一个新的版本，从而完成“改”操作，旧的版本依旧保留，系统会定时回收垃圾数据，只留下最新的几个版本，保存的版本数可以在创建表的时候指定。

添加数据的方式有点像KV存储的方式，put

```
put 'user','12138','uName:name1','lyh'
```

这个12138是行键，自己指定的。每个属性下可以再分子属性，例如12138用户的用户名中的第一个为’lyh’。

正统点的说法叫做uName:name1列下添加了一个名叫’lyh’的数据

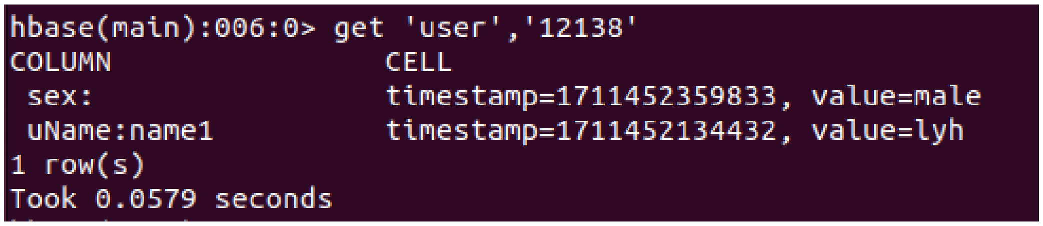
1.4. 删除数据

两种delete/deleteAll。前者删除某个行键下的指定属性的数据，后者删除具有同一行键的数据。

1.5. 查看数据

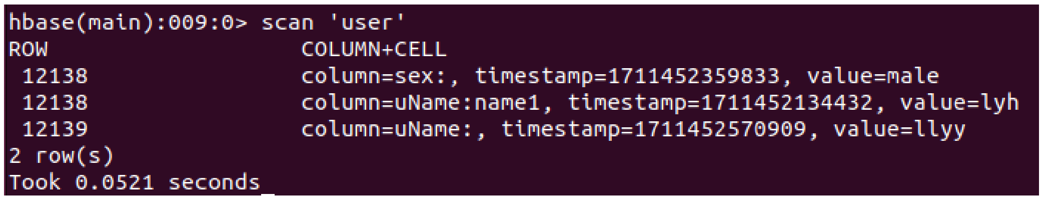
get/scan。前者获取具有同一行号的所有属性，后者返回表的所有属性。

```
get 'user','12138'
```



```
hbase(main):006:0> get 'user','12138'
COLUMN                    CELL
 sex:                     timestamp=1711452359833, value=male
 uName:name1              timestamp=1711452134432, value=lyh
1 row(s)
Took 0.0579 seconds
```

```
scan 'user'
```



```
hbase(main):009:0> scan 'user'
ROW                      COLUMN+CELL
 12138                   column=sex:, timestamp=1711452359833, value=male
 12138                   column=uName:name1, timestamp=1711452134432, value=lyh
 12139                   column=uName:, timestamp=1711452570909, value=llyy
2 row(s)
Took 0.0521 seconds
```

1.6. 删除表

先让表不可用，再删除表

```
disable 'dummy'
drop 'dummy'
```

1.7. 查询表历史数据

这需要在创建表的时候就指定一个保存数据的版本数。

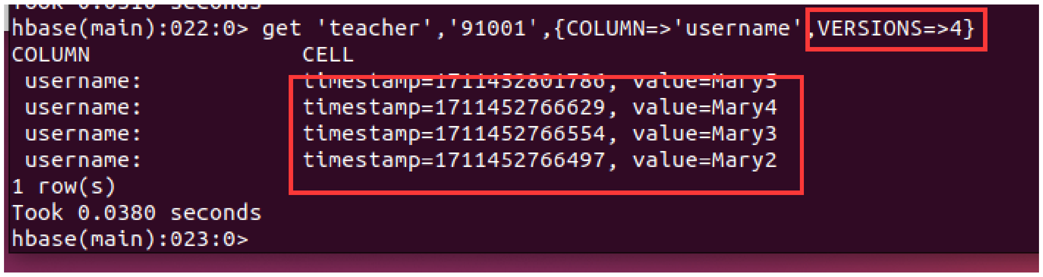
```
create 'teacher',{NAME=>'username',VERSIONS=>5}
```

创建几个历史数据

```
put 'teacher','91001','username','Mary'
put 'teacher','91001','username','Mary1'
put 'teacher','91001','username','Mary2'
put 'teacher','91001','username','Mary3'
put 'teacher','91001','username','Mary4'
put 'teacher','91001','username','Mary5'
```

指定一个版本号查询：

```
get 'teacher','91001',{COLUMN=>'username',VERSIONS=>5}
```



```
hbase(main):022:0> get 'teacher','91001',{COLUMN=>'username',VERSIONS=>4}
COLUMN                CELL
 username:            timestamp=1711452801786, value=Mary5
 username:            timestamp=1711452766629, value=Mary4
 username:            timestamp=1711452766554, value=Mary3
 username:            timestamp=1711452766497, value=Mary2
1 row(s)
Took 0.0380 seconds
hbase(main):023:0>
```

1.7. 推出hbase shell

```
exit
```

2. JAVA API

2.1. 导入依赖库

还是先导JAR包。

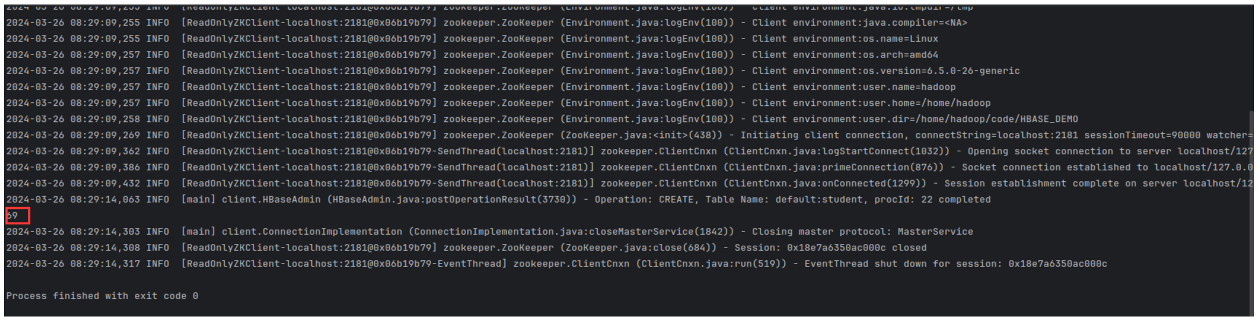
1. /usr/local/hbase/lib
2. /usr/local/hbase/lib/client-facing-thirdparty

2.2. 编写应用程序

```
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 HBASE_OPS {
    public static Configuration configuration;
    public static Connection connection;
    public static Admin admin;
    public static void main(String[] args)throws IOException{
        init();
        createTable("student",new String[]{"score"});
        insertData("student","zhangsan","score","English","69");
        insertData("student","zhangsan","score","Math","86");
        insertData("student","zhangsan","score","Computer","77")
        getData("student", "zhangsan", "score","English");
        close();
    }

    public static void init(){
        configuration  = HBaseConfiguration.create();
        configuration.set("hbase.rootdir","hdfs://localhost:9000
        try{
            connection = ConnectionFactory.createConnection(conf
            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 createTable(String myTableName,String[] c
        TableName tableName = TableName.valueOf(myTableName);
        if(admin.tableExists(tableName)){
            System.out.println("talbe is exists!");
        }else {
            TableDescriptorBuilder tableDescriptor = TableDescri
            for(String str:colFamily){
                ColumnFamilyDescriptor family =
                        ColumnFamilyDescriptorBuilder.newBuilder
                tableDescriptor.setColumnFamily(family);
            }
            admin.createTable(tableDescriptor.build());
        }
    }

    public static void insertData(String tableName,String rowKey
        Table table = connection.getTable(TableName.valueOf(tabl
        Put put = new Put(rowKey.getBytes());
        put.addColumn(colFamily.getBytes(),col.getBytes(), val.g
        table.put(put);
        table.close();
    }

    public static void getData(String tableName,String rowKey,St
        Table table = connection.getTable(TableName.valueOf(tabl
        Get get = new Get(rowKey.getBytes());
        get.addColumn(colFamily.getBytes(),col.getBytes());
        Result result = table.get(get);
        System.out.println(new String(result.getValue(colFamily.
        table.close();
    }
}
```

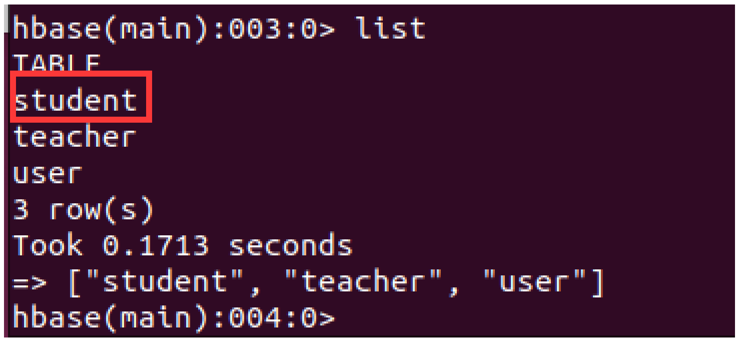
大致就是初始化好数据库，然后建一个student表，插入几条数据，然后查询一下。

查询 那个也打印出来了

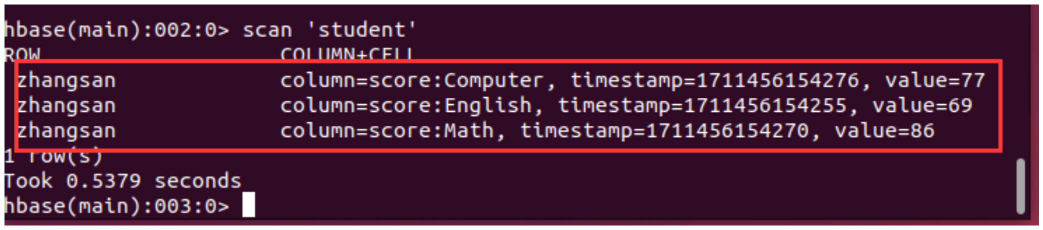


```
2024-03-26 08:29:09,255 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:java.io.tmpdir=/tmp
2024-03-26 08:29:09,255 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:java.compiler=<NA>
2024-03-26 08:29:09,255 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:os.name=Linux
2024-03-26 08:29:09,257 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:os.arch=amd64
2024-03-26 08:29:09,257 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:os.version=6.5.0-26-generic
2024-03-26 08:29:09,257 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:user.name=hadoop
2024-03-26 08:29:09,257 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:user.home=/home/hadoop
2024-03-26 08:29:09,258 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (Environment.java:logEnv(100)) - Client environment:user.dir=/home/hadoop/code/HBASE_DEMO
2024-03-26 08:29:09,269 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (ZooKeeper.java:<init>(438)) - Initiating client connection, connectString=localhost:2181 sessionTimeout=90000 watcher=
2024-03-26 08:29:09,362 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79-SendThread(localhost:2181)] zookeeper.ClientCnxn (ClientCnxn.java:logStartConnect(1032)) - Opening socket connection to server localhost/127
2024-03-26 08:29:09,386 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79-SendThread(localhost:2181)] zookeeper.ClientCnxn (ClientCnxn.java:primeConnection(876)) - Socket connection established to localhost/127.0.0
2024-03-26 08:29:09,432 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79-SendThread(localhost:2181)] zookeeper.ClientCnxn (ClientCnxn.java:onConnected(1299)) - Session establishment complete on server localhost/12
2024-03-26 08:29:14,063 INFO  [main] client.HBaseAdmin (HBaseAdmin.java:postOperationResult(3730)) - Operation: CREATE, Table Name: default:student, procId: 22 completed
69
2024-03-26 08:29:14,303 INFO  [main] client.ConnectionImplementation (ConnectionImplementation.java:closeMasterService(1842)) - Closing master protocol: MasterService
2024-03-26 08:29:14,308 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79] zookeeper.ZooKeeper (ZooKeeper.java:close(684)) - Session: 0x18e7a6350ac000c closed
2024-03-26 08:29:14,317 INFO  [ReadOnlyZKClient-localhost:2181@0x06b19b79-EventThread] zookeeper.ClientCnxn (ClientCnxn.java:run(519)) - EventThread shut down for session: 0x18e7a6350ac000c

Process finished with exit code 0
```

启动hbase shell



```
hbase(main):003:0> list
TABLE
student
teacher
user
3 row(s)
Took 0.1713 seconds
=> ["student", "teacher", "user"]
hbase(main):004:0>
```



```
hbase(main):002:0> scan 'student'
ROW                    COLUMN+CELL
 zhangsan              column=score:Computer, timestamp=1711456154276, value=77
 zhangsan              column=score:English, timestamp=1711456154255, value=69
 zhangsan              column=score:Math, timestamp=1711456154270, value=86
1 row(s)
Took 0.5379 seconds
hbase(main):003:0>
```

可以看到应用程序创建的表和插入的数据