**Hbase Java API**

Exercise:

Write a java code to do the following.

1. Write a function to Check if table **scores** exist. If yes, then drop the table and create it with column family **grade** and **course**.
2. Write a function to insert data.

Mona, grade, Average,5

Mona, course, phil, 90

Mona, course, math, 97

Joe, grade, Average, 4

Joe, course, math, 89

Joe, course, sci, 99

1. Write a function to display the contents of the table.
2. Rerun the program by adding this record in the code.

Mona, course, art, 87

(Note: when first time the program is run, the table won’t be available. Rerun of the code will perform the delete operation)

Solution

**Note: Please find the java files required for this exercise in this path: /home/mapr/Desktop/mapr\_training/java\_files**

Step 1: Write and save the java code as HBaseExe.java. In this case it is already saved.

import java.io.IOException;

import java.util.ArrayList;

import java.util.List;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.hbase.HBaseConfiguration;

import org.apache.hadoop.hbase.HColumnDescriptor;

import org.apache.hadoop.hbase.HTableDescriptor;

import org.apache.hadoop.hbase.KeyValue;

import org.apache.hadoop.hbase.MasterNotRunningException;

import org.apache.hadoop.hbase.ZooKeeperConnectionException;

import org.apache.hadoop.hbase.client.Delete;

import org.apache.hadoop.hbase.client.Get;

import org.apache.hadoop.hbase.client.HBaseAdmin;

import org.apache.hadoop.hbase.client.HTable;

import org.apache.hadoop.hbase.client.Result;

import org.apache.hadoop.hbase.client.ResultScanner;

import org.apache.hadoop.hbase.client.Scan;

import org.apache.hadoop.hbase.client.Put;

import org.apache.hadoop.hbase.util.Bytes;

public class HbaseExe {

private static Configuration conf = null;

/\*\*

\* Initialization

\*/

static {

conf = HBaseConfiguration.create();

}

/\*\*

\* Create a table

\*/

public static void creatTable(String tableName, String[] familys)

throws Exception {

HBaseAdmin admin = new HBaseAdmin(conf);

if (admin.tableExists(tableName)) {

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

admin.disableTable(tableName);

admin.deleteTable(tableName);

System.out.println("delete table " + tableName + " ok.");

}

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 + " ok.");

}

/\*\*

\* Put (or insert) a row

\*/

public static void putRecord(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 recored " + rowKey + " to table "

+ tableName + " ok.");

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* Get a row

\*/

public static void getRecord (String tableName, String rowKey) throws IOException{

HTable table = new HTable(conf, tableName);

Get get = new Get(rowKey.getBytes());

Result rs = table.get(get);

for(KeyValue kv : rs.raw()){

System.out.print(new String(kv.getRow()) + " " );

System.out.print(new String(kv.getFamily()) + ":" );

System.out.print(new String(kv.getQualifier()) + " " );

System.out.print(kv.getTimestamp() + " " );

System.out.println(new String(kv.getValue()));

}

}

/\*\*

\* Scan (or list) a table

\*/

public static void ScanTable(String tableName) {

try{

HTable table = new HTable(conf, tableName);

Scan s = new Scan();

ResultScanner ss = table.getScanner(s);

for(Result r:ss){

for(KeyValue kv : r.raw()){

System.out.print(new String(kv.getRow()) + " ");

System.out.print(new String(kv.getFamily()) + ":");

System.out.print(new String(kv.getQualifier()) + " ");

System.out.print(kv.getTimestamp() + " ");

System.out.println(new String(kv.getValue()));

}

}

} catch (IOException e){

e.printStackTrace();

}

}

public static void main(String[] agrs) {

try {

String tablename = "scores";

String[] familys = { "grade", "course" };

HbaseExe.creatTable(tablename, familys);

/\* add record Mona \*/

HbaseExe.putRecord(tablename, "Mona", "grade", "Average", "5");

HbaseExe.putRecord(tablename, "Mona", "course", "phil", "90");

HbaseExe.putRecord(tablename, "Mona", "course", "math", "97");

HbaseExe.putRecord(tablename, "Mona", "course", "art", "87");

/\* add record Joe \*/

HbaseExe.putRecord(tablename, "Joe", "grade", "Average", "4");

HbaseExe.putRecord(tablename, "Joe", "course", "math", "89");

HbaseExe.putRecord(tablename, "Joe", "course", "sci", "99");

System.out.println("===========get one record========");

HbaseExe.getRecord(tablename, "Mona");

System.out.println("===========show all record========");

HbaseExe.ScanTable(tablename);

} catch (Exception e) {

e.printStackTrace();

}

}

}

Step 2: Create class path

MYCLASSPATH=$(hadoop classpath):$(hbase classpath)

Step 3: Goto the directory where HBaseExe.java exists & compile the code

javac -cp $MYCLASSPATH HBaseExe.java

Step 4: Run the java code

java -cp .:$MYCLASSPATH HBaseExe

**Note: prog needs to be replaced by the class name of the code.**