HBase编程实践

讲师：董西成
博客：dongxicheng.org
微信二维码见右。

Open Passion Value

目录

Hbase 访问方式

- **Native Java API**
  - ✓ 最常规和高效的访问方式；
- **HBase Shell**
  - ✓ HBase的命令行工具，最简单的接口，适合HBase管理使用；
- **Thrift Gateway**
  - ✓ 利用Thrift序列化技术，支持C++，PHP，Python等多种语言，适合其他异构系统在线访问HBase表数据；
- **REST Gateway**
  - ✓ 支持REST 风格的Http API访问HBase, 解除了语言限制；
- **MapReduce**
  - ✓ 直接使用MapReduce作业处理Hbase数据；
  - ✓ 使用Pig/hive处理Hbase数据。

目录

Hbase Java API概述

- Hbase是用Java语言编写的，支持Java编程是自然而然的事情；
- 支持CRUD操作；
  - ✓Create, Read, Update, Delete
- Java API包含Hbase shell支持的所有功能，甚至更多；
- Java API是访问Hbase最快的方式。

Java API程序设计步骤

步骤1：创建一个Configuration对象

- ✓ 包含各种配置信息

步骤2：构建一个HTable句柄

- ✓ 提供Configuration对象
- ✓ 提供待访问Table的名称

步骤3：执行相应的操作

- ✓ 执行put、get、delete、scan等操作

步骤4：关闭HTable句柄

- ✓ 将内存数据刷新到磁盘上
- ✓ 释放各种资源

Java API程序设计步骤

步骤1：创建一个Configuration对象

```
Configuration conf = HbaseConfiguration.create();
```

步骤2：构建一个HTable句柄

```
HTable table = new HTable(conf, tableName);
```

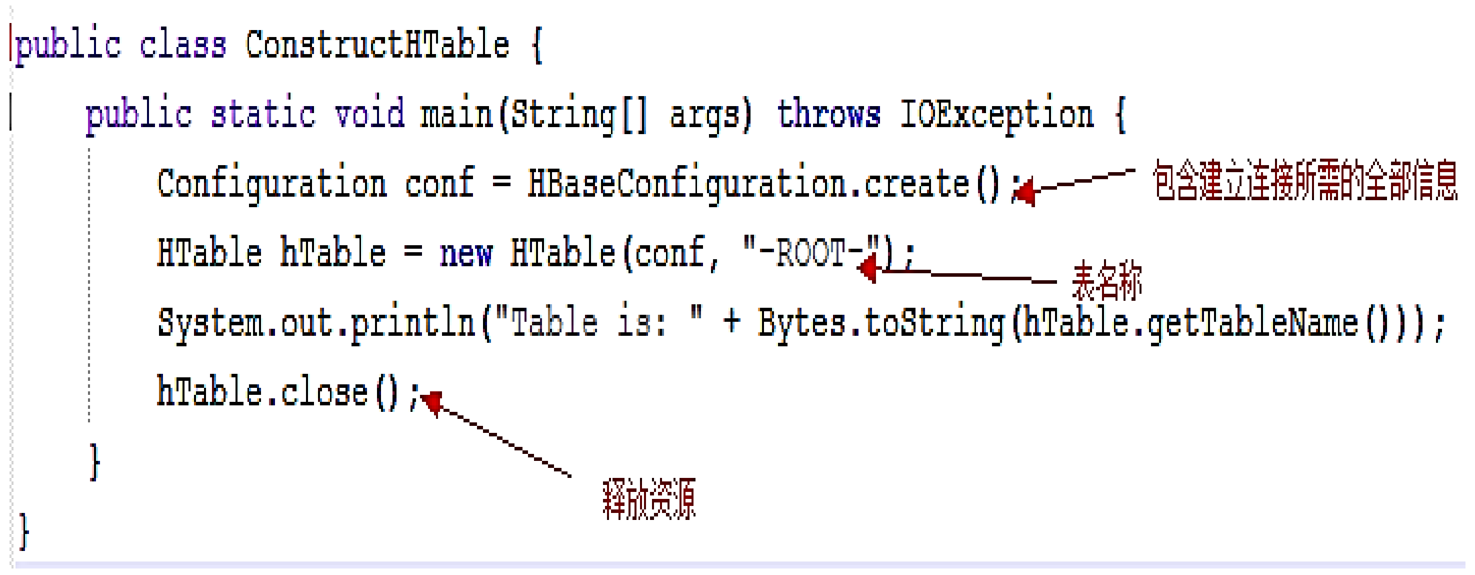
步骤3：执行相应的操作

```
table.getTableName();
```

步骤4：关闭HTable句柄

```
table.close();
```

示例程序



```
public class ConstructHTable {
    public static void main(String[] args) throws IOException {
        Configuration conf = HBaseConfiguration.create();
        HTable hTable = new HTable(conf, "-ROOT-");
        System.out.println("Table is: " + Bytes.toString(hTable.getTableName()));
        hTable.close();
    }
}
```

创建Configuration对象

- Configuration对象包装了客户端程序连接Hbase服务所需的全部信息；
  - ✓ Zookeeper位置
  - ✓ Zookeeper连接超时时间
- HbaseConfiguration.create()内部逻辑
  - ✓ 从CLASSPATH中加载hbase-default.xml和hbase-site.xml两个文件
    - hbase-default.xml已经被打包到Hbase jar包中
    - hbase-site.xml需添加到CLASSPATH中
    - hbase-site.xml将覆盖hbase-default.xml中的同名属性

创建Configuration对象

- **Hbase如何从CLASSPATH中获取hbase-site.xml信息;**
  - ✓ **修改hadoop脚本，将Hbase CLASSPATH加入**
  - ✓ **在<hadoop_install>/conf/hadoop-env.sh中设置**

    export HADOOP_CLASSPATH=$HBASE_HOME/*:$HBASE_HOME/conf:$HADOOP_CLASSPATH
- **检查Hadoop CLASSPATH**
  - ✓ **hadoop classpath**
  - ✓ **hadoop classpath | grep hbase**

创建Configuration对象

- ➢ 如果已经有一个Configuration文件，可进行如下操作；

```
Configuration newConf = Configuration.create(existingConf);
```

  - ✓ 用户自定义的配置文件将在已有配置文件之后加载
  - ✓ 将覆盖hbase-default.xml和hbase-site.xml中的配置

- ➢ 可单独覆盖某一个或多个参数值

```
Configuration conf = HbaseConfiguration.create();
conf.set("hbase.zookeeper.quorum", "node1,node2");
```

  - ✓ 通常不推荐这么做！

创建HTable句柄

org.apache.hadoop.hbase.client.HTable

- ➢ 一个table对应一个HTable句柄
- ➢ 提供了CRUD操作
- ➢ 设计简单、使用方便
- ➢ 提供行级事务
  - ✓ 不支持多行事务或者表级别的事务
  - ✓ 严格的行一致性
  - ✓ 并发读、顺序写

创建HTable句柄

- 创建HTable句柄代价很大
  - ✓ 扫描.META.表等；
  - ✓ 创建一次，以后尽可能复用；
  - ✓ 如果需要创建多个Htable句柄，使用HTablePool；
- HTable并非线程安全的
  - ✓ 一个线程创建一个即可
- Htable支持CRUD批处理
  - ✓ 非线程安全，仅是为了提高性能

向HBase写入数据

步骤1：创建一个Put对象；

- ✓ Put put = new Put(Bytes.toBytes("rowkey"));

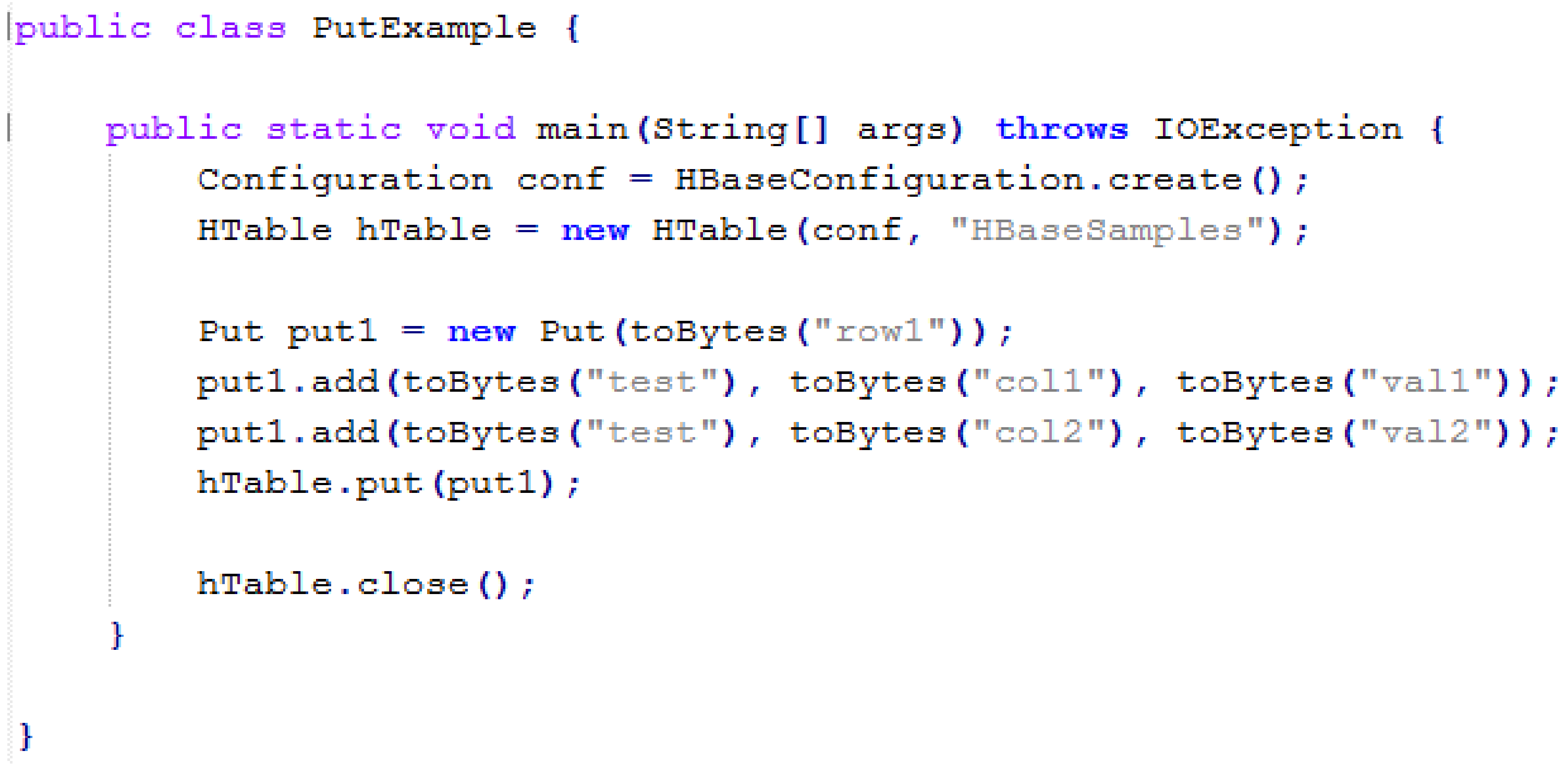
步骤2：设置cell值；

- ✓ Put.add(family, column, value)
- ✓ Put.add(family, column, timestamp, value)
- ✓ Put.add(KeyValue kv)

步骤3：调用HTable中的put方法，写入数据；

步骤4：关闭HTable句柄。

向HBase写入数据



```
public class PutExample {

    public static void main(String[] args) throws IOException {
        Configuration conf = HBaseConfiguration.create();
        HTable hTable = new HTable(conf, "HBaseSamples");

        Put put1 = new Put(toBytes("row1"));
        put1.add(toBytes("test"), toBytes("col1"), toBytes("val1"));
        put1.add(toBytes("test"), toBytes("col2"), toBytes("val2"));
        hTable.put(put1);

        hTable.close();
    }

}
```

从Hbase中读取数据

- 支持的API类型
  - ✓ 通过rowkey获取一行数据
  - ✓ 通过一个rowkey集合获取多条记录
  - ✓ 扫描整个表或者表的一部分
- 扫描表
  - ✓ 可指定扫描的范围[startkey endkey)
  - ✓ 表中数据是按照rowkey排序的
- API 特点
  - ✓ 数目有限、使用简单

从Hbase中读取数据

- 读取数据时注意事项
  - ✓ 只读取需要的数据
  - ✓ 尽可能增加数据约束条件
  - ✓ 可增加**family, column(s), time range** 和 **max versions**等约束条件
- 接口实例
  - ✓ **get.setTimeRange(minStamp, maxStamp)**
  - ✓ **get.setMaxVersions(maxVersions)**
  - ✓ **get.addFamily(family)**
  - ✓ **get.addColumn(family, column)**

从Hbase中读取数据

```
private static void print(Result result) {
    System.out.println("-----------------------------------");
    System.out.println("RowId: " + Bytes.toString(result.getRow()));
    byte [] val1 = result.getValue(toBytes("test"), toBytes("col1"));
    System.out.println("test1:col1="+Bytes.toString(val1));
    byte [] val2 = result.getValue(toBytes("test"), toBytes("col2"));
    System.out.println("test1:col2="+Bytes.toString(val2));
}
```

```
        print(result);

        get.addColumn(toBytes("test"), toBytes("col2"));
        result = hTable.get(get);
        print(result);

        hTable.close();
    }
```

从Hbase中删除数据

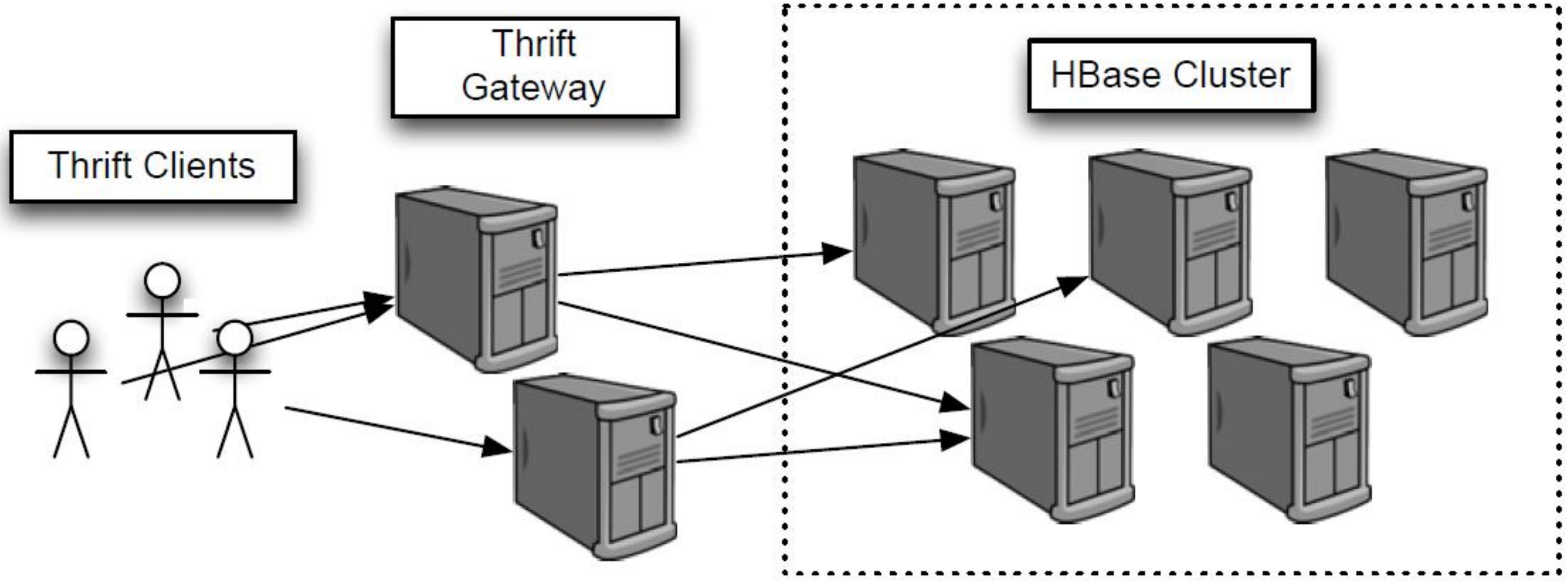
```
public class DeleteExample {
    public static void main(String[] args) throws IOException {
        Configuration conf = HBaseConfiguration.create();
        HTable hTable = new HTable(conf, "HBaseSamples");

        Delete delete = new Delete(toBytes("rowToDelete"));
        hTable.delete(delete);

        Delete delete1 = new Delete(toBytes("anotherRow"));
        delete1.deleteColumns(toBytes("metrics"), toBytes("loan"));
        hTable.delete(delete1);

        hTable.close();
    }
}
```

从Hbase中scan数据

```java
public class ScanExample {

    public static void main(String[] args) throws IOException {
        Configuration conf = HBaseConfiguration.create();
        HTable hTable = new HTable(conf, "HBaseSamples");

        scan(hTable, "row-03", "row-05");
        scan(hTable, "row-10", "row-15");
        hTable.close();
    }

    private static void scan(HTable hTable, String startRow,
            String stopRow) throws IOException {
        System.out.println("Scanning from " +
                "["+startRow+"] to ["+stopRow+"]");

        Scan scan = new Scan(toBytes(startRow), toBytes(stopRow));
        scan.addColumn(toBytes("metrics"), toBytes("counter"));
        ResultScanner scanner = hTable.getScanner(scan);
        for ( Result result : scanner){
            byte [] value = result.getValue(
                    toBytes("metrics"), toBytes("counter"));
            System.out.println("   " +
                    Bytes.toString(result.getRow()) + " => " +
                    Bytes.toString(value));
        }
        scanner.close();
    }
}
```

目录

Hbase-Thrift 拓扑结构



使用步骤

1. **HDFS/HBase安装**
2. **启动Hbase thrift server**
   - ✓ **bin/hbase-daemon.sh start thrift**
3. **生成Hbase thrift client接口文件**
   - ✓ **thrift --gen php Hbase.thrift**
   - ✓ **thrift --gen cpp Hbase.thrift**
4. **编写客户端代码**

Hbase C++编程

1. 生成Hbase thrift client接口文件
   - ✓ thrift --gen cpp Hbase.thrift
   - ✓ hbase.thrift位置：

     ${HBASE_HOME}/src/main/resources/org/apache/hadoop/hbase/thrift/Hbase.thrift
2. 编写客户端代码
   - ✓${HBASE_HOME}/src/examples/thrift/DemoClient.cpp
3. 编译代码（make）
4. 运行程序
   - ✓ ./DemoClient

Hbase C++编程—头文件

```
#include <stdio.h>
#include <unistd.h>
#include <sys/time.h>
#include <poll.h>

#include <iostream>

#include <boost/lexical_cast.hpp>
#include <protocol/TBinaryProtocol.h>
#include <transport/TSocket.h>
#include <transport/TTransportUtils.h>

#include "Hbase.h"

using namespace apache::thrift;
using namespace apache::thrift::protocol;
using namespace apache::thrift::transport;

using namespace apache::hadoop::hbase::thrift;

namespace {

typedef std::vector<std::string> StrVec;
typedef std::map<std::string,std::string> StrMap;
typedef std::vector<ColumnDescriptor> ColVec;
typedef std::map<std::string,ColumnDescriptor> ColMap;
typedef std::vector<TCell> CellVec;
typedef std::map<std::string,TCell> CellMap;
```

Hbase C++编程—初始化

```
int
main(int argc, char** argv)
{
  if (argc < 3) {
    std::cerr << "Invalid arguments!\n" << "Usage: DemoClient host port" << std::endl;
    return -1;
  }

  boost::shared_ptr<TTransport> socket(new TSocket("localhost", boost::lexical_cast<int>(argv[2])));
  boost::shared_ptr<TTransport> transport(new TBufferedTransport(socket));
  boost::shared_ptr<TProtocol> protocol(new TBinaryProtocol(transport));
  HbaseClient client(protocol);

  try {
    transport->open();

    std::string t("demo_table");
```

Hbase C++编程—获取所有表

```
std::cout << "scanning tables..." << std::endl;
StrVec tables;
client.getTableNames(tables);
for (StrVec::const_iterator it = tables.begin(); it != tables.end(); ++it) {
  std::cout << "  found: " << *it << std::endl;
  if (t == *it) {
    if (client.isTableEnabled(*it)) {
      std::cout << "    disabling table: " << *it << std::endl;
      client.disableTable(*it);
    }
    std::cout << "    deleting table: " << *it << std::endl;
    client.deleteTable(*it);
  }
}
```

Hbase C++编程—创建表

```
ColVec columns;
columns.push_back(ColumnDescriptor());
columns.back().name = "entry:";
columns.back().maxVersions = 10;
columns.push_back(ColumnDescriptor());
columns.back().name = "unused:";

std::cout << "creating table: " << t << std::endl;
try {
  client.createTable(t, columns);
} catch (const AlreadyExists &ae) {
  std::cerr << "WARN: " << ae.message << std::endl;
}

ColMap columnMap;
client.getColumnDescriptors(columnMap, t);
std::cout << "column families in " << t << ": " << std::endl;
for (ColMap::const_iterator it = columnMap.begin(); it != columnMap.end(); ++it) {
  std::cout << "  column: " << it->second.name << ", maxVer: " << it->second.maxVersions << std::endl;
}
```

Hbase C++编程—插入数据

```
std::string invalid("foo-\xfc\xa1\xa1\xa1\xa1\xa1");
std::string valid("foo-\xE7\x94\x9F\xE3\x83\x93\xE3\x83\xBC\xE3\x83\xAB");

// non-utf8 is fine for data
std::vector<Mutation> mutations;
mutations.push_back(Mutation());
mutations.back().column = "entry:foo";
mutations.back().value = invalid;
client.mutateRow(t, "foo", mutations);

// try empty strings
mutations.clear();
mutations.push_back(Mutation());
mutations.back().column = "entry:";
mutations.back().value = "";
client.mutateRow(t, "", mutations);

// this row name is valid utf8
mutations.clear();
mutations.push_back(Mutation());
mutations.back().column = "entry:foo";
mutations.back().value = valid;
client.mutateRow(t, valid, mutations);
```

Hbase C++编程—全表扫描

```
StrVec columnNames;
columnNames.push_back("entry:");

std::cout << "Starting scanner..." << std::endl;
int scanner = client.scannerOpen(t, "", columnNames);
try {
  while (true) {
    std::vector<TRowResult> value;
    client.scannerGet(value, scanner);
    if (value.size() == 0)
      break;
    printRow(value);
  }
} catch (const IOError &ioe) {
  std::cerr << "FATAL: Scanner raised IOError" << std::endl;
}

client.scannerClose(scanner);
std::cout << "Scanner finished" << std::endl;
```

Hbase C++编程—全表扫描

```
columnNames.clear();
client.getColumnDescriptors(columnMap, t);
std::cout << "The number of columns: " << columnMap.size() << std::endl;
for (ColMap::const_iterator it = columnMap.begin(); it != columnMap.end(); ++it) {
  std::cout << " column with name: " + it->second.name << std::endl;
  columnNames.push_back(it->second.name);
}
std::cout << std::endl;

std::cout << "Starting scanner..." << std::endl;
scanner = client.scannerOpenWithStop(t, "00020", "00040", columnNames);
try {
  while (true) {
    std::vector<TRowResult> value;
    client.scannerGet(value, scanner);
    if (value.size() == 0)
      break;
    printRow(value);
  }
} catch (const IOError &ioe) {
  std::cerr << "FATAL: Scanner raised IOError" << std::endl;
}

client.scannerClose(scanner);
std::cout << "Scanner finished" << std::endl;

transport->close();
```

Hbase Python编程

1. **生成Hbase thrift client接口文件**
   - ✓ **thrift --gen py hbase.thrift**
   - ✓ **hbase.thrift位置：**

     ${HBASE_HOME}/src/main/resources/org/apache/hadoop/hbase/thrift/Hbase.thrift
2. **编写客户端代码**
   - ✓**${HBASE_HOME}/src/examples/thrift/DemoClient.py**
3. **运行程序**
   - ✓ **python DemoClient.py**

Hbase Python编程—头文件与初始化

```
import sys
import time

from thrift import Thrift
from thrift.transport import TSocket, TTransport
from thrift.protocol import TBinaryProtocol
from hbase import ttypes
from hbase.Hbase import Client, ColumnDescriptor, Mutation

# Make socket
transport = TSocket.TSocket('localhost', 9090)

# Buffering is critical. Raw sockets are very slow
transport = TTransport.TBufferedTransport(transport)

# Wrap in a protocol
protocol = TBinaryProtocol.TBinaryProtocol(transport)

# Create a client to use the protocol encoder
client = Client(protocol)

# Connect!
transport.open()

t = "demo_table"
```

Hbase Python编程—获取所有表

```
print "scanning tables..."
for table in client.getTableNames():
  print "  found: %s" %(table)
  if table == t:
    if client.isTableEnabled(table):
      print "    disabling table: %s"  %(t)
      client.disableTable(table)
    print "    deleting table: %s"  %(t)
    client.deleteTable(table)
```

Hbase Python编程—创建表

```
columns = []
col = ColumnDescriptor()
col.name = 'entry:'
col.maxVersions = 10
columns.append(col)
col = ColumnDescriptor()
col.name = 'unused:'
columns.append(col)

try:
  print "creating table: %s" %(t)
  client.createTable(t, columns)
except AlreadyExists, ae:
  print "WARN: " + ae.message
```

Hbase Python编程—插入数据

```
invalid = "foo-\xfc\xa1\xa1\xa1\xa1\xa1"
valid = "foo-\xE7\x94\x9F\xE3\x83\x93\xE3\x83\xBC\xE3\x83\xAB";

# non-utf8 is fine for data
mutations = [Mutation(column="entry:foo",value=invalid)]
print str(mutations)
client.mutateRow(t, "foo", mutations)

# try empty strings
mutations = [Mutation(column="entry:", value="")]
client.mutateRow(t, "", mutations)

# this row name is valid utf8
mutations = [Mutation(column="entry:foo", value=valid)]
client.mutateRow(t, valid, mutations)
```

Hbase Python编程—全表扫描

```
# Run a scanner on the rows we just created
print "Starting scanner..."
scanner = client.scannerOpen(t, "", ["entry:"])

r = client.scannerGet(scanner)
while r:
  printRow(r[0])
  r = client.scannerGet(scanner)
print "Scanner finished"
```

Hbase Python编程—全表扫描

```
columnNames = []
for (col, desc) in client.getColumnDescriptors(t).items():
  print "column with name: "+desc.name
  print desc
  columnNames.append(desc.name+":")

print "Starting scanner..."
scanner = client.scannerOpenWithStop(t, "00020", "00040", columnNames)

r = client.scannerGet(scanner)
while r:
  printRow(r[0])
  r = client.scannerGet(scanner)

client.scannerClose(scanner)
print "Scanner finished"

transport.close()
```

Hbase更多语言编程

1. **PHP**

   ✓**${HBASE_HOME}/src/examples/thrift/DemoClient.php**

2. **Ruby**

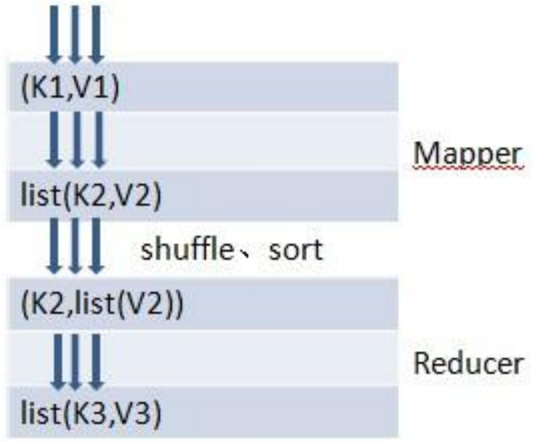
   ✓ **${HBASE_HOME}/src/examples/thrift/DemoClient.rb**

3. **其他语言**

   ✓ **Thrift支持的语言全部支持，包括：C++、C#、Cocoa、D、Delphi、Erlang、Haskell、Java、Perl、PHP、Python、Ruby、Smalltalk等**

目录

Hbase MapReduce编程—基础



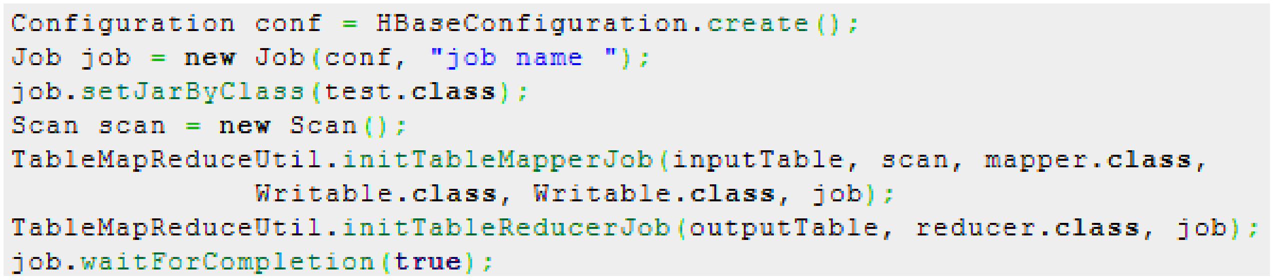
Hbase MapReduce	Hadoop MapReduce
org.apache.hadoop.hbase.mapreduce.TableMapper	org.apache.hadoop.mapreduce.Mapper
org.apache.hadoop.hbase.mapreduce.TableReducer	org.apache.hadoop.mapreduce.Reducer
org.apache.hadoop.hbase.mapreduce.TableInputFormat	org.apache.hadoop.mapreduce.InputFormat
org.apache.hadoop.hbase.mapreduce.TableOutputFormat	org.apache.hadoop.mapreduce.OutputFormat

```
public abstract class TableMapper<KEYOUT, VALUEOUT>
   extends Mapper<ImmutableBytesWritable, Result, KEYOUT, VALUEOUT>
{}
```

```
public abstract class TableReducer<KEYIN, VALUEIN, KEYOUT>
   extends Reducer<KEYIN, VALUEIN, KEYOUT, Writable> { }
```

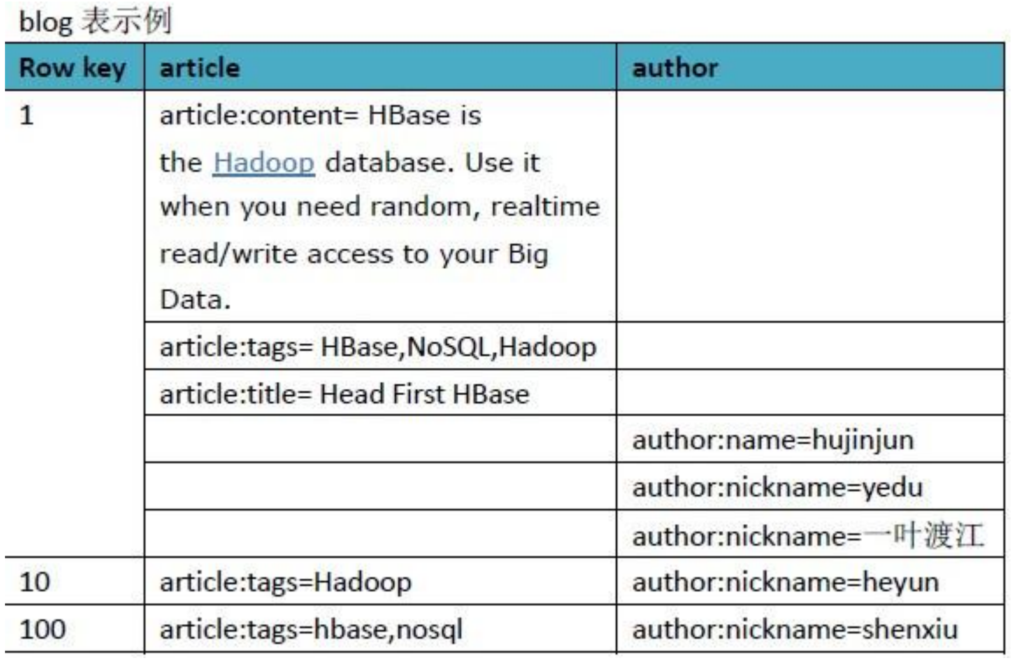
Hbase MapReduce编程—基础

- **MapReduce程序基本框架**
  - ✓ 创建Job对象，设置基本属性;
  - ✓ 设置scan对象，指定扫描区间和数据列;
  - ✓ 调用TableMapReduceUtil的initTableMapperJob和initTableReducerJob设置Mapper和Reducer等信息;
  - ✓ 提交作业



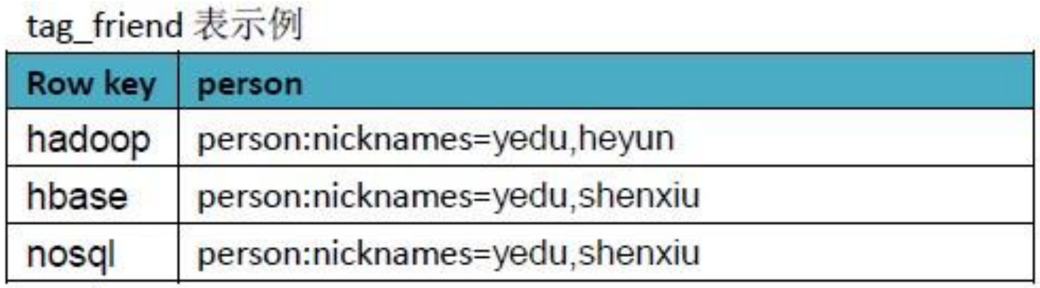
```
Configuration conf = HBaseConfiguration.create();
Job job = new Job(conf, "job name ");
job.setJarByClass(test.class);
Scan scan = new Scan();
TableMapReduceUtil.initTableMapperJob(inputTable, scan, mapper.class,
              Writable.class, Writable.class, job);
TableMapReduceUtil.initTableReducerJob(outputTable, reducer.class, job);
job.waitForCompletion(true);
```

Hbase MapReduce编程—实例



blog 表示例

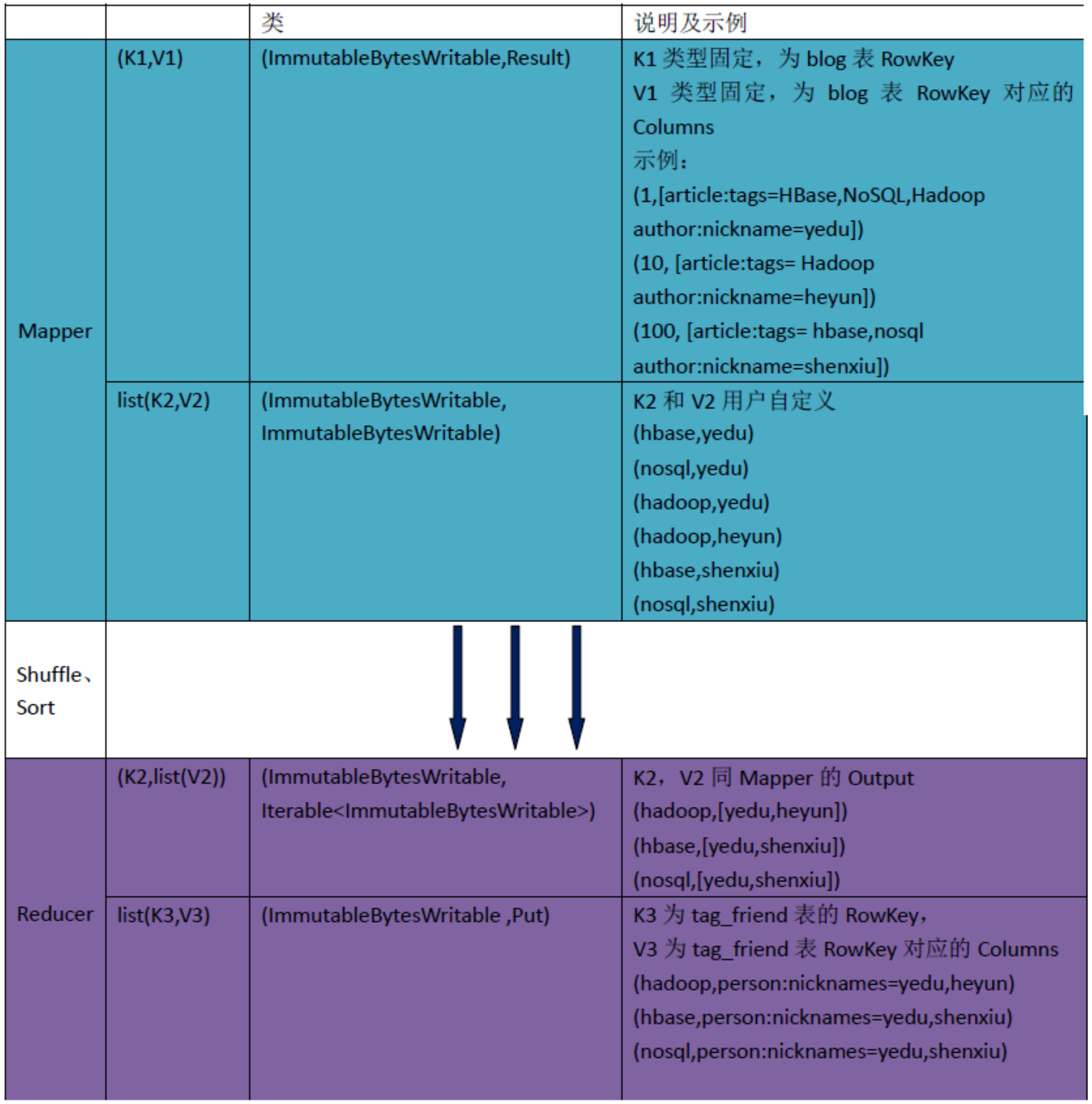
Row key	article	author
1	article:content= HBase is the Hadoop database. Use it when you need random, realtime read/write access to your Big Data.	
	article:tags= HBase,NoSQL,Hadoop	
	article:title= Head First HBase	
		author:name=hujinjun
		author:nickname=yedu
		author:nickname=一叶渡江
10	article:tags=Hadoop	author:nickname=heyun
100	article:tags=hbase,nosql	author:nickname=shenxiu



tag_friend 表示例

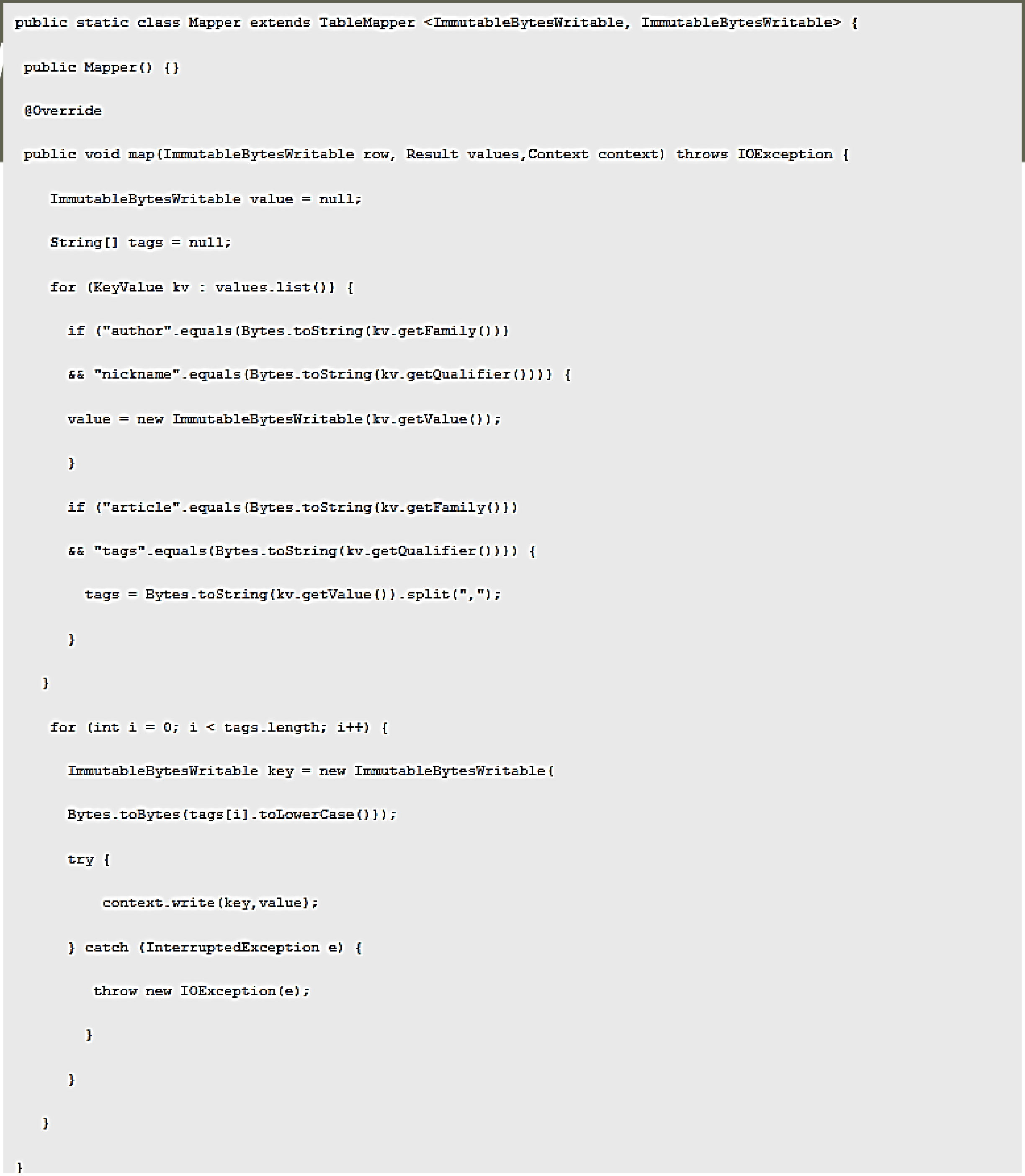
Row key	person
hadoop	person:nicknames=yedu,heyun
hbase	person:nicknames=yedu,shenxiu
nosql	person:nicknames=yedu,shenxiu

Hbase MapReduce编程—实例概述



		类	说明及示例
Mapper	(K1,V1)	(ImmutableBytesWritable,Result)	K1 类型固定，为 blog 表 RowKey V1 类型固定，为 blog 表 RowKey 对应的 Columns 示例： (1,[article:tags=HBase,NoSQL,Hadoop author:nickname=yedu]) (10, [article:tags= Hadoop author:nickname=heyun]) (100, [article:tags= hbase,nosql author:nickname=shenxiu])
	list(K2,V2)	(ImmutableBytesWritable, ImmutableBytesWritable)	K2 和 V2 用户自定义 (hbase,yedu) (nosql,yedu) (hadoop,yedu) (hadoop,heyun) (hbase,shenxiu) (nosql,shenxiu)
Shuffle、Sort			
Reducer	(K2,list(V2))	(ImmutableBytesWritable, Iterable<ImmutableBytesWritable>)	K2，V2 同 Mapper 的 Output (hadoop,[yedu,heyun]) (hbase,[yedu,shenxiu]) (nosql,[yedu,shenxiu])
	list(K3,V3)	(ImmutableBytesWritable ,Put)	K3 为 tag_friend 表的 RowKey， V3 为 tag_friend 表 RowKey 对应的 Columns (hadoop,person:nicknames=yedu,heyun) (hbase,person:nicknames=yedu,shenxiu) (nosql,person:nicknames=yedu,shenxiu)

Hbase M



```
public static class Mapper extends TableMapper <ImmutableBytesWritable, ImmutableBytesWritable> {

 public Mapper() {}

 @Override

 public void map(ImmutableBytesWritable row, Result values,Context context) throws IOException {

    ImmutableBytesWritable value = null;

    String[] tags = null;

    for (KeyValue kv : values.list()) {

      if ("author".equals(Bytes.toString(kv.getFamily()))

      && "nickname".equals(Bytes.toString(kv.getQualifier()))) {

      value = new ImmutableBytesWritable(kv.getValue());

      }

      if ("article".equals(Bytes.toString(kv.getFamily()))

      && "tags".equals(Bytes.toString(kv.getQualifier()))) {

        tags = Bytes.toString(kv.getValue()).split(",");

      }

   }

    for (int i = 0; i < tags.length; i++) {

      ImmutableBytesWritable key = new ImmutableBytesWritable(

      Bytes.toBytes(tags[i].toLowerCase()));

      try {

          context.write(key,value);

      } catch (InterruptedException e) {

         throw new IOException(e);

        }

      }

   }

}
```

Hbase MapReduce编程—Reducer实现

```
public static class Reducer extends TableReducer <ImmutableBytesWritable, ImmutableBytesWritable, ImmutableBytesWritable> {

 @Override

 public void reduce(ImmutableBytesWritable key,Iterable values,

   Context context) throws IOException, InterruptedException {

  String friends="";

  for (ImmutableBytesWritable val : values) {

   friends += (friends.length()>0?",":"")+Bytes.toString(val.get());

  }

  Put put = new Put(key.get());

  put.add(Bytes.toBytes("person"), Bytes.toBytes("nicknames"),

  Bytes.toBytes(friends));

  context.write(key, put);

 }

}
```

Hbase MapReduce编程—main函数实现

```
public static void main(String[] args) throws Exception {

 Configuration conf = new Configuration();

 conf = HBaseConfiguration.create(conf);

 Job job = new Job(conf, "HBase_FindFriend");

 job.setJarByClass(FindFriend.class);

 Scan scan = new Scan();

 scan.addColumn(Bytes.toBytes("author"),Bytes.toBytes("nickname"));

 scan.addColumn(Bytes.toBytes("article"),Bytes.toBytes("tags"));

 TableMapReduceUtil.initTableMapperJob("blog", scan,FindFriend.Mapper.class,

  ImmutableBytesWritable.class, ImmutableBytesWritable.class, job);

 TableMapReduceUtil.initTableReducerJob("tag_friend",FindFriend.Reducer.class, job);

 System.exit(job.waitForCompletion(true) ? 0 : 1);

}
```

目录

总结

- **Hbase API概述**
- **Hbase Java编程**
- **Hbase多语言编程**
- **Hbase-MapReduce编程**
- **总结**