高级HBase

- 谭唐华

本章工作任务

- HBase表的设计
- HBase表属性
- HBase表管理
- HBase与Hive集成使用
- HBase实战案例

本课内容

- HBase表的设计
- HBase表的属性
- HBase创建表时的预分区
- HBase表设计

HBase表的创建

namespace命令

- 创建表的例子，如下：

```
Create a table with namespace=ns1 and table qualifier=t1
  hbase> create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}
```

- 创建一张表，伴随着一个namespace和表的唯一性标识符（也就是表的名称）。
- 在新版的HBase当中，表放在命名空间namespace中，类似于它属于哪个数据库，一个namespace下面有很多表。

1. **命名空间分类**
   - 默认情况下HBase有两个命名空间，一个是default，也就是说默认情况创建的表，都在此命名空间，一个是HBase，它是HBase的系统命令空间。
   - HBase这个命名空间有2张特别的表，一个是meta表，用来存储元数据，用户表region的相关信息；一个是namespace表，用来存储命名空间。

HBase表的创建

2. 命名空间查看帮助

- 如何创建一个命名空间,命名空间有一组单独的命令：

```
  Group name: namespace
  Commands: alter_namespace, create_namespace, describe_namespace, drop_namespace, list_namespac
e, list_namespace_tables
```

- 此语句的含义是：namespace组包含创建一个命名空间、查看命名空间的描述、删除命名空间并且列举一个命名空间有哪些表这些命令。如果不会用这些命令，记得使用help命令查看命令的详细信息，如下：

```
hbase(main):003:0> help "create_namespace"
Create namespace; pass namespace name,
and optionally a dictionary of namespace configuration.
Examples:

  hbase> create_namespace 'ns1'
  hbase> create_namespace 'ns1', {'PROERTY_NAME'=>'PROPERTY_VALUE'}
hbase(main):004:0>
```

- 上图是使用’ help “create_namespace”’ 查看后的详细信息，它提示我们创建时需要传递一个名称和一些可选项包括一个配置的目录语法等。

HBase表的创建

3. 命名空间的创建

- 语法格式如下：

  create_namespace ‘名称’或create_namespace ‘名称’,{可选项}

- 下图是表示创建一个名称为ns1的命名空间：

```
hbase(main):004:0> create_namespace 'ns1'
2015-10-12 10:56:45,260 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library
 for your platform... using builtin-java classes where applicable
0 row(s) in 3.9860 seconds
```

4. 查看命名空间

- 下图是查看有多少个命名空间：

```
hbase(main):005:0> list_namespace
NAMESPACE
default
hbase
ns1
3 row(s) in 0.1540 seconds
```

HBase表的创建

创建表命令

- 以下列举了一些常用的建表语法形式。第一种建表语法方式

  语法格式：create ‘命名空间:表名’,{列簇名，版本等可选项信息}，例：

```
create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}
```

  - 该语句用来创建一张表，指定其命名空间为’ ns1’ ，表名为’ t1’ ，列簇为 ‘f1’ ，版本信息为5

- 第二种建表语法方式

  语法格式：create ‘命名空间:表名’，’ 列簇名’ ，例：

```
hbase(main):006:0> create 'ns1:t1', 'cf'
```

  - 表示在ns1这个命名空间创建一张名称为t1的表，表有1个列簇，列簇名是cf。该形式等价于：“create ‘ns1:t1’ ,{NAME=>’ cf’ }”。如果t1这张表有更多的属性，我们就要用“create ‘ns1:t1’ ,{NAME=>’ cf’ }”这种方式来创建表。

HBase表的创建

- 第三种建表语法方式

语法格式：create ‘表名’, {NAME => ‘列簇名1’}, {NAME => ‘列簇名2’}.....

```
hbase> create 't1', {NAME => 'f1'}, {NAME => 'f2'}, {NAME => 'f3'}
```

- 可使用以下语法格式代替

语法格式：create ‘表名’ ‘列簇名1’, ‘列簇名2’.....

```
hbase> create 't1', 'f1', 'f2', 'f3'
```

HBase表的创建

◆ 查看表的详细信息

◆ 查看一个表的详细信息，语句如下:

```
hbase(main):010:0> describe 'ns1:t2'
DESCRIPTION                                                                    ENABLED
 'ns1:t2', {NAME => 'f1', DATA_BLOCK_ENCODING => 'NONE', BLOOM true
 FILTER => 'ROW', REPLICATION_SCOPE => '0', VERSIONS => '1', C
 OMPRESSION => 'NONE', MIN_VERSIONS => '0', TTL => 'FOREVER',
 KEEP_DELETED_CELLS => 'false', BLOCKSIZE => '65536', IN_MEMOR
 Y => 'false', BLOCKCACHE => 'true'}, {NAME => 'f2', DATA_BLOC
 K_ENCODING => 'NONE', BLOOMFILTER => 'ROW', REPLICATION_SCOPE
  => '0', VERSIONS => '1', COMPRESSION => 'NONE', MIN_VERSIONS
  => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'false', BLO
 CKSIZE => '65536', IN_MEMORY => 'false', BLOCKCACHE => 'true'
 }, {NAME => 'f3', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER
 => 'ROW', REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESS
 ION => 'NONE', MIN_VERSIONS => '0', TTL => 'FOREVER', KEEP_DE
 LETED_CELLS => 'false', BLOCKSIZE => '65536', IN_MEMORY => 'f
 alse', BLOCKCACHE => 'true'}
1 row(s) in 0.2380 seconds
```

本课内容

- HBase表的设计
- HBase表的属性
- HBase创建表时的预分区
- HBase表设计

HBase表属性

- **VERSIONS**
  - 数值的版本
  - 默认值1

```
't1',
 {
        NAME => 'cf',
        DATA_BLOCK_ENCODING => 'NONE',
        BLOOMFILTER => 'ROW',
        REPLICATION_SCOPE => '0',
        VERSIONS => '1',
        COMPRESSION => 'NONE',
        MIN_VERSIONS => '0',
        TTL => 'FOREVER',
        KEEP_DELETED_CELLS => 'false',
        BLOCKSIZE => '65536',
        IN_MEMORY => 'false',
        BLOCKCACHE => 'true'
 }
```

Hadoop Native Libraries in HBase

- **snappy压缩**
  - 数据压缩方式
  - 默认值NONE

```
't1',
 {
        NAME => 'cf',
        DATA_BLOCK_ENCODING => 'NONE',
        BLOOMFILTER => 'ROW',
        REPLICATION_SCOPE => '0',
        VERSIONS => '1',
        COMPRESSION => 'NONE',
        MIN_VERSIONS => '0',
        TTL => 'FOREVER',
        KEEP_DELETED_CELLS => 'false',
        BLOCKSIZE => '65536',
        IN_MEMORY => 'false',
        BLOCKCACHE => 'true'
 }
```

HBase表属性

- **Memstore**
  - 写数据
- **BlockCache**
  - 读数据

```
't1',
 {
        NAME => 'cf',
        DATA_BLOCK_ENCODING => 'NONE',
        BLOOMFILTER => 'ROW',
        REPLICATION_SCOPE => '0',
        VERSIONS => '1',
        COMPRESSION => 'NONE',
        MIN_VERSIONS => '0',
        TTL => 'FOREVER',
        KEEP_DELETED_CELLS => 'false',
        BLOCKSIZE => '65536',
        IN_MEMORY => 'false',
        BLOCKCACHE => 'true'
 }
```

本课内容

- HBase表的设计
- HBase表的属性
- HBase创建表时的预分区
- HBase表设计

预分区是什么

```
Table configuration options can be put at the end.
Examples:

  hbase> create 'ns1:t1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS_FILE => 'splits.txt', OWNER => 'johndoe'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 5}, METADATA => { 'mykey'
  hbase> # Optionally pre-split the table into NUMREGIONS, using
  hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit
```

- 在上图中经常出现splits这个单词，它是什么意思呢？我们从表中数据开始分析。HBase的表的数据是存在Region里面的，Region有[startkey,endkey)，并且是包头不包尾的，每个Region都有一个范围。

【注】默认情况下，创建一个HBase表，会自动只为表分配1个Region。

预分区方式

- 预分区方式有两种常用方式，一种是利用HBase建表命令，一种是利用文件存放Rowkey及建表命令。
  - **方式一：**
    - 利用建表语句create ‘表名’,’ 列族’,splits => [‘Rowkey1’ ， ‘Rowkey2’ ……]

```
hbase(main):011:0> create 'bflogs', 'info', SPLITS => ['20151001000000000', '20151011000000000',
'20151021000000000']
```

  - **方式二:**
    - 指定一个文件，我们可以把我们要分区的Rowkey放在一个文件当中，然后通过建表命令的SPLITES_FILE=>’文件名’来指定
      1. **创建bflogs-split.txt的文件作为rowkey的文件**
      2. **编辑bflogs-split.txt文件，写上Rowkey，写的时候是不用加引号的：**

```
[beifeng@hadoop-senior datas]$ vi bflogs-split.txt
```

预分区方式

```
create 'bflogs2', 'info', SPLITS_FILE => '/opt/datas/bflogs-split.txt'
```

③ 再创建一张表为bflogs2,SPLITES_FILE=>”绝对路径”,例：

④ 验证，查看表bflogs2详细信息。

- 其他方式：
  - 除了以上两个常用的方式官方还提供了一些其它方法的案例，例：

```
hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
```

  - 它其实是指定了一个类，为表进行划分区域，测试该第三种方法：

```
create 't11', 'f11', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit'}
```

  - NUMERGIONS=>15是设置15个region,用HexStringSplit这个来生成 Region，HDFS根据16进制来生成的,HexStringSplit表示16进制的一个字符串。该方法用处不是很大。

本课内容

- HBase表的设计
- HBase表的属性
- HBase创建表时的预分区
- HBase表设计

分析【话单数据分析】项目需求

2015-04-12 13:54:53	江苏南京	主叫	18[redacted]	39秒	国内通话	0.00
2015-04-12 10:28:01	江苏南京	被叫	186[redacted]190	1分6秒	国内通话	0.00
2015-04-12 10:20:40	江苏南京	主叫	13485[redacted]	52秒	国内通话	0.00
2015-04-12 10:16:43	江苏南京	被叫	18[redacted]4599	1分10秒	国内通话	0.00
2015-04-12 10:10:50	江苏南京	被叫	15[redacted]144	4分58秒	国内通话	0.00
2015-04-12 09:53:31	江苏南京	被叫	[redacted]62775	58秒	国内通话	0.00
2015-04-12 09:21:53	江苏南京	被叫	18502[redacted]	11秒	国内通话	0.00
2015-04-11 18:58:33	江苏南京	被叫	156[redacted]982	36秒	国内通话	0.00
2015-04-11 18:30:35	江苏南京	主叫	156553[redacted]	57秒	国内通话	0.00
2015-04-11 11:21:43	江苏南京	被叫	[redacted]562775	2分29秒	国内通话	0.00

- 用户需要进行实时的查询，那么这些数据是放在HBase当中的，每个客户每天接打电话至少20个左右，而通信公司拥有很多用户，每天产生的数据都是上亿条。

分析【话单数据分析】项目需求

- **分析上图得到以下结果：**

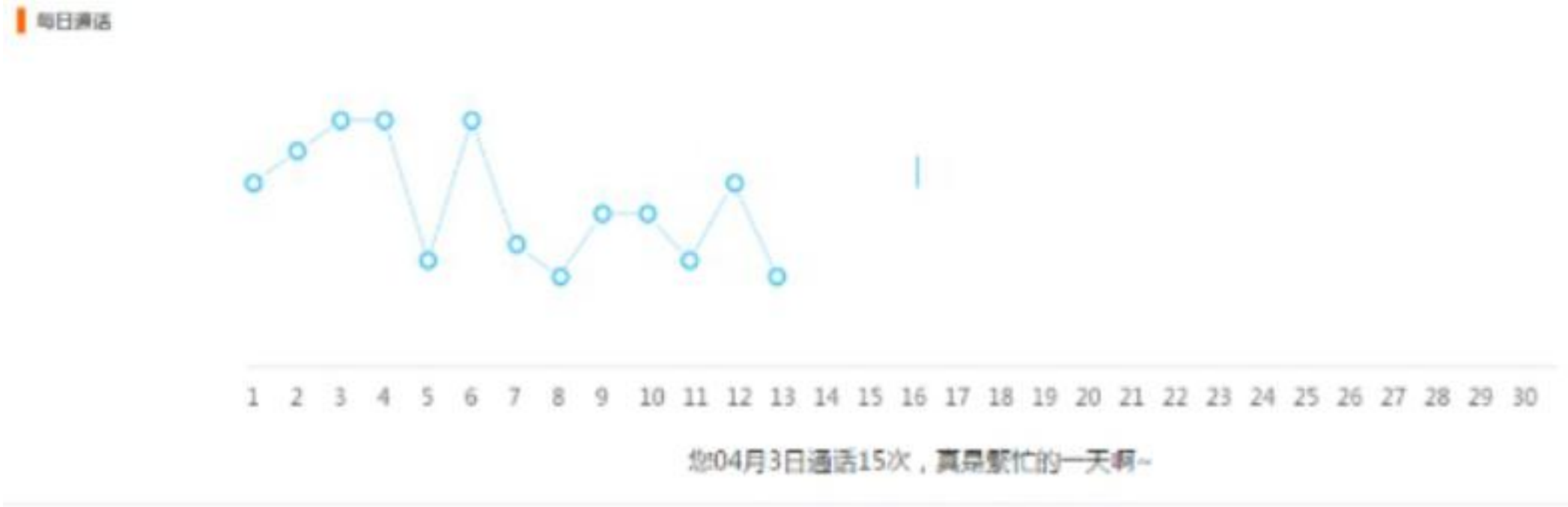
1. 上图中总通话127次，是查询在时间范围内的所有通话记录，可用count统计总数;日均9次：总数/天数;

2. 累计约4时16分：累加在时间范围内的通话时间；日均9分钟:总通话时间/天数

分析【话单数据分析】项目需求

- 常联系的小伙伴功能中包含前10位常被联系的人，和前10位主动联系的人。并且各人都统计了次数。

分析【话单数据分析】项目需求



- 统计出通话次数最多的一天是4月3日

分析【话单数据分析】项目需求

- 分析上面的功能需求，提取出需要的信息，主要包括以下几点：

1. 自己的号码:telphone
2. 拨打或接听时间：teltime
3. 区域:area
4. 主叫或被叫:active
5. 对方的号码:phone
6. 通话时长：talktime
7. 通话模式(国内或国外)：mode
8. 费用：price

- 而大部分功能的查询条件分析如下：

  - telphone +(starttime - endtime)，条件是：号码+开始时间——结束时间

分析【话单数据分析】项目表

1. 设计Rowkey

- 条件在上一章提过：号码+开始时间——结束时间，那么设计Rowkey就是telphone(电话号码)+teltime(通话时间)
- **在表的Rowkey设计中:**
  - **核心思想：**
    - **依据Rowkey查询最快**
    - **在实际的应用当中，就是对Rowkey进行范围查询range，Rowkey通常都是多个字段组成的。**
    - **Rowkey是前缀匹配的**

在设计表的时候掌握这几点就能把rowkey设计好。

分析【话单数据分析】项目表

2. 索引表/辅助表(主表)

- 在客户购买新房没装修前，装潢公司会每天打电话寻求客户的意见，是否要装修。现在，装潢公司想查询三个月以来向这个客户拨打了多少次电话。

  设计出rowKey：phone(客户号码) + time(时间)

- **索引表/辅助表功能**：用于辅助主表。
- **索引表创建方式**：

  列名：Rowkey+列值：主表对应该Rowkey的值

- phone_time，比如（要查询10月1日到10月24日这23天打了多少次电话）根据依据设计出来rowkey如下：

  182600937646_2015100100000

  182600937646_2015102400000

分析【话单数据分析】项目表

- **索引表/辅助表同步的问题：**
  - **第一个：**如果是手动往HBase里面实时抽数据的话，需要自己写代码程序同步。
  - **第二个：**在HBase当中使用phoenix，这个框架提供了像JDBC的方式。**注意：**往表里面插入数据时，只能通过JDBC的方式，才能同步主表和索引表的数据，其它的方式都不能同步。
  - **创建索引表的另一种方式：**把一张表重要的几个字段，给它创建索引，存在solr里面，进行查询的时候，首先查询solr里面的数据，然后把对应的rowkey拿出来，之后从HBase主表直接Get。