Hbase空间数据存储组织

报告人　　袁帅

目录

一、Hbase简介

- NOSQL（Not Only SQL，不限于SQL）是一类范围非常广泛的持久化解决方案，它们不遵循关系数据库模型，也不使用SQL作为查询语言。
- 简单地讲，NOSQL数据库可以按照它们的数据模型分成4类：
  1. 键-值存储库（Key-Value-stores）
  2. BigTable实现（BigTable-implementations）HBase
  3. 文档库（Document-stores）
  4. 图形数据库（Graph Database）

一、Hbase简介

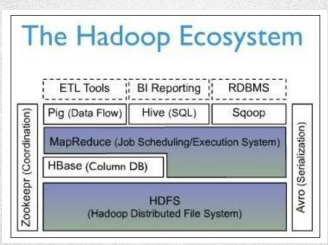
- Hbase是bigtable的开源山寨版本。是建立的hdfs之上，提供高可靠性、高性能、列存储、可伸缩、实时读写的数据库系统。它仅能通过主键(row key)和主键的range来检索数据，仅支持单行事务，主要用来存储非结构化和半结构化的松散数据。
- Hbase中的表一般有这样的特点：
  1 大：一个表可以有上亿行，上百万列
  2 面向列:面向列(族)的存储和权限控制，列(族)独立检索。
  3 稀疏:对于为空(null)的列，并不占用存储空间，因此，表可以设计的非常稀疏。

一、Hbase简介

- Hbase的优点
  1 列的可以动态增加，并且列为空就不存储数据,节省存储空间.
  2 Hbase自动切分数据，使得数据存储自动具有水平scalability.
  3 Hbase可以提供高并发读写操作的支持
- Hbase的缺点：不能支持条件查询，只支持按照Row key来查询.

一、Hbase简介

- 下面一幅图是Hbase在Hadoop Ecosystem中的位置



二、逻辑视图

- Hbase以表的形式存储数据。表有行和列组成。列划分为若干个列族(row family)

Row Key	column-family1		column-family2			column-family3
	column1	column1	column1	column2	column3	column1
key1	t1:abc t2:gdxdf		t4:dfads t3:hello t2:world			
key2	t3:abc t1:gdxdf		t4:dfads t3:hello		t2:dfdsfa t3:dfdf	
key3		t2:dfadfasd t1:dfdasddsf				t2:dfxxdfasd t1:taobao.com

二、逻辑视图

- 与nosql数据库们一样,row key是用来检索记录的主键。访问Hbase table中的行，只有三种方式：
  1 通过单个row key访问
  2 通过row key的range
  3 全表扫描
- Row key行键 (Row key)可以是任意字符串(最大长度是 64KB，实际应用中长度一般为 10-100bytes)，在Hbase内部，row key保存为字节数组。

二、逻辑视图

- 存储时，数据按照Row key的字典序(byte order)排序存储。设计key时，要充分考虑排序存储这个特性，将经常一起读取的行存储放到一起。(位置相关性)
- 注意：字典序对int排序的结果是
  1,10,100,11,12,13,14,15,16,17,18,19,2,20,21,…,9,91,92,93,94,95,96,97,98,99。要保持整形的自然序，行键必须用0作左填充。

二、逻辑视图

- 列族　　Hbase表中的每个列，都归属与某个列族。列族是表的chema的一部分(而列不是)，必须在使用表之前定义。列名都以列族作为前缀。例如courses:history，courses:math都属于courses 这个列族。
- 访问控制、磁盘和内存的使用统计都是在列族层面进行的。实际应用中，列族上的控制权限能帮助我们管理不同类型的应用：我们允许一些应用可以添加新的基本数据、一些应用可以读取基本数据并创建继承的列族、一些应用则只允许浏览数据(甚至可能因为隐私的原因不能浏览所有数据)。

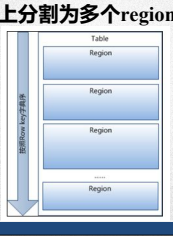
二、逻辑视图

时间戳

- Hbase中通过row和columns确定的为一个存贮单元称为cell。每个cell都保存着同一份数据的多个版本。版本通过时间戳来索引。时间戳的类型是 64位整型。时间戳可以由Hbase(在数据写入时自动)赋值，此时时间戳是精确到毫秒的当前系统时间。时间戳也可以由客户显式赋值。如果应用程序要避免数据版本冲突，就必须自己生成具有唯一性的时间戳。每个 cell中，不同版本的数据按照时间倒序排序，即最新的数据排在最前面。
- 为了避免数据存在过多版本造成的的管理 (包括存贮和索引)负担，Hbase提供了两种数据版本回收方式。一是保存数据的最后n个版本，二是保存最近一段时间内的版本(比如七天)。用户可以针对每个列族进行设置。

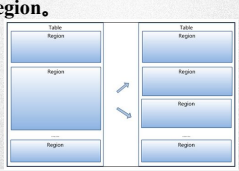
三、物理存储

- 1 Table中的所有行都按照row key的字典序排列。
- 2 Table 在行的方向上分割为多个region。



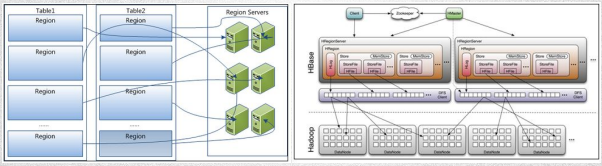
三、物理存储

- 3 region按大小分割的，每个表一开始只有一个region，随着数据不断插入表，region不断增大，当增大到一个阀值的时候，region就会等分会两个新的region。当table中的行不断增多，就会有越来越多的region。



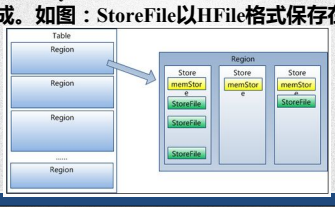
三、物理存储

- 4 Region是Hbase中分布式存储和负载均衡的最小单元。最小单元就表示不同的region可以分布在不同的Region server上。但一个region是不会拆分到多个server上的。



三、物理存储

- 5 Region虽然是分布式存储的最小单元，但并不是存储的最小单元。事实上，Region由一个或者多个Store组成，每个store保存一个columns family。每个Strore又由一个memStore和0至多个StoreFile组成。如图：StoreFile以HFile格式保存在HDFS上。



四、插入删除

创建表：

```
hbase(main):010:0> create 'blog_user','userInfo'
0 row(s) in 15.3320 seconds

=> Hbase::Table - blog_user
```

四、插入删除

插入数据

```
hbase(main):012:0> put'blog_user','www.aboutyun.com','userInfo:user_Name' 'abou
yun'
0 row(s) in 1.7530 seconds
```

- 上面我们看到了1所示是什么，这是Hbse所特有的，是一个rowkey，是系统自带的，也是Hbse中一条记录的唯一标识。
- 2所示是我们插入的列user_Name，这也是最难以理解的地方，列竟然可以插入。并且其‘value’为3即'aboutyun'我们插入了列，下面我们来查看一下效果：

四、插入删除

```
hbase(main):014:0> scan 'blog_user'
ROW
 www.aboutyun.com
1 row(s) in 0.2010 seconds
```

- 下面来解释一下上面的含义：

  1为rowkey，插入数据’www.aboutyun.com‘，

  2为列族下面列的名字user_Name，

  3我们并没有在设计的添加这个列族，所以这个是系统自带的，这个是记录的操作时间，以时间戳的形式放到hbase里面。

  4是我们插入的user_Name的值

四、插入删除

下面我们在插入password：

```
hbase(main):015:0> put'blog_user','www.aboutyun.com','userInfo:user_Password','aboutyun'
0 row(s) in 0.1130 seconds
```

再次查询结果：

```
hbase(main):016:0> scan 'blog_user'
ROW                           COLUMN+CELL
 www.aboutyun.com             column=userInfo:user_Name, timestamp=1400663775901, value=aboutyun
 www.aboutyun.com             column=userInfo:user_Password, timestamp=1400665203430, value=aboutyun
1 row(s) in 0.0390 seconds
```

- **到这里，我们看到两行记录，传统数据块认为这是两行数据，对于Hbse，这是一条记录。**

四、插入删除

删除列数据

删除数据分为删除列和删除记录

1.删除列

```
hbase(main):017:0> delete 'blog_user','www.aboutyun.com','userInfo:user_Password'
0 row(s) in 0.4190 seconds

hbase(main):018:0> scan 'blog_user'
ROW                           COLUMN+CELL
 www.aboutyun.com             column=userInfo:user_Name, timestamp=1400663775901, value=aboutyun
```

从上面我们看出列被删除了

四、插入删除

2.删除记录：

这是删除之前显示结果：

```
hbase(main):013:0> scan 'blog_user'
ROW                           COLUMN+CELL
 www.aboutyun.com             column=userInfo:user_Name, timestamp=1400667424333, value=aboutyun
1 row(s) in 0.0360 seconds
```

删除后结果

```
hbase(main):006:0> deleteall 'blog_user','www.aboutyun.com'
0 row(s) in 0.2270 seconds

hbase(main):007:0> scan 'blog_user'
ROW                           COLUMN+CELL
0 row(s) in 0.0300 seconds
```

执行命令

删除后结果查询为空

THANKS