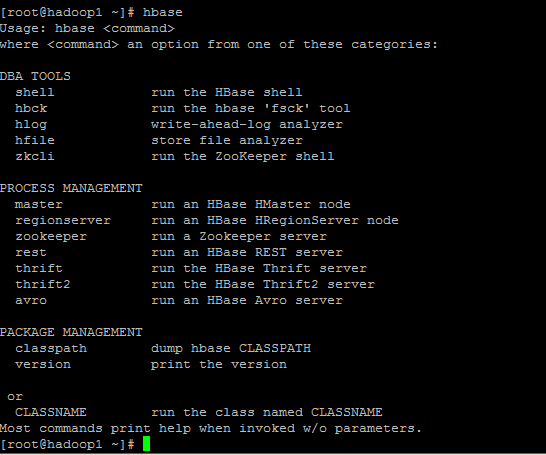
Hadoop学习之HBase基础知识、操作和原理

1. HBase 简介

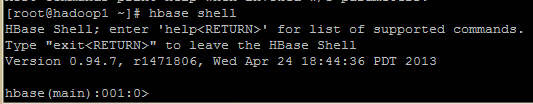
HBase(hadoop DataBase)是一个高可靠，高性能面向列，可伸缩的分布式存储系统，利用HBase技术可在廉价PC server 上搭建大规模结构化存储集群。HBase 利用HDFS作为其文件系统，利用Hadoop MapReduce来处理HBase中的海量数据

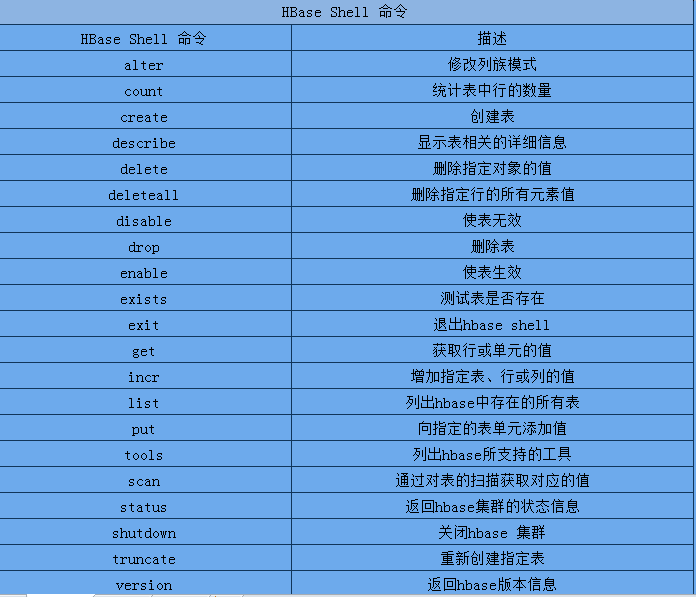
1. HBase Shell 操作

HBase 可执行参数

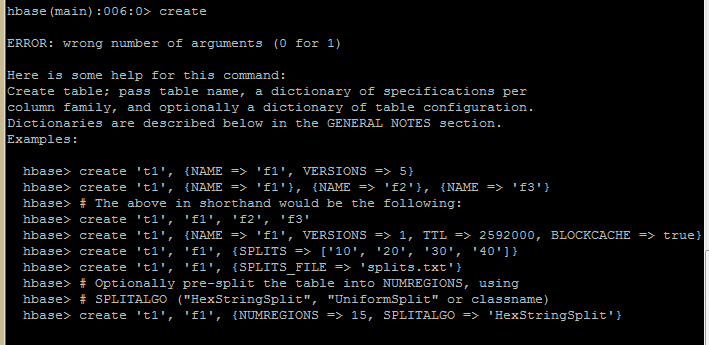


hbase shell

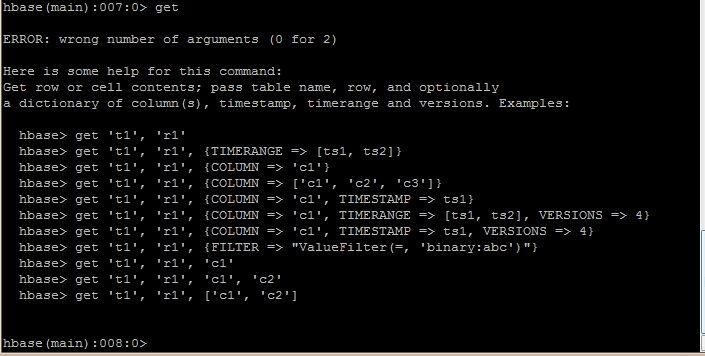




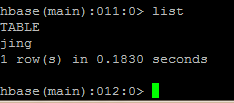
* 1. 创建表的帮助语法(create)



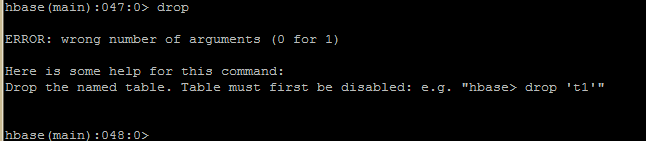
* 1. 获取行或单元的值(get)



* 1. 列出hbase 上的所有表(list)

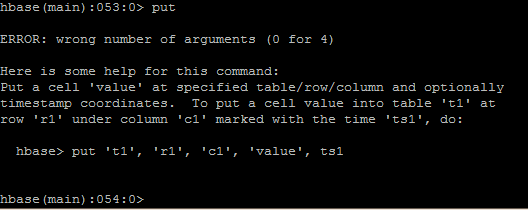


* 1. 删除表(drop)

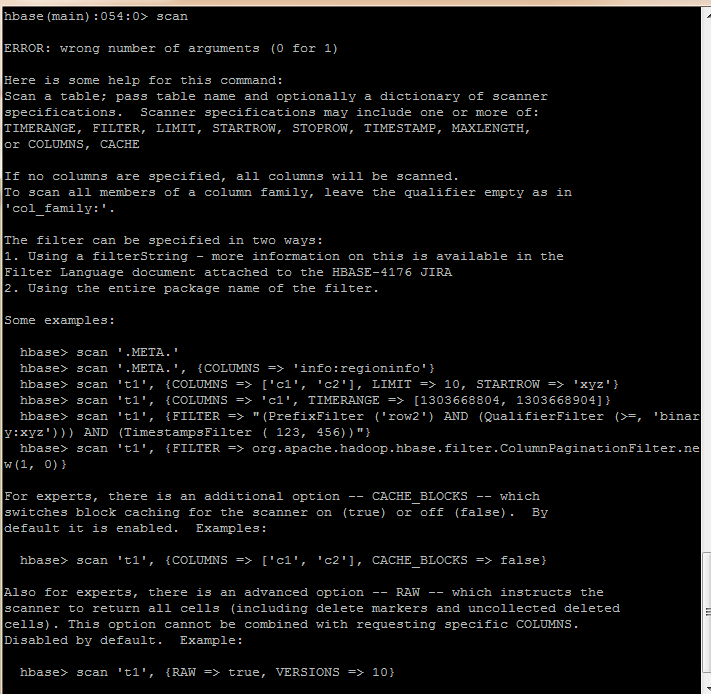


删除表之前必须先让表失效:disable ‘t1’

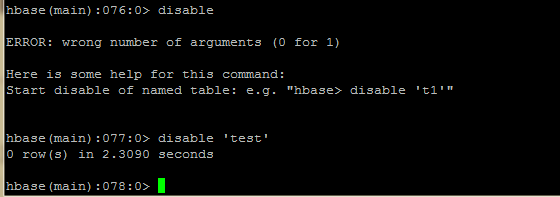
* 1. 向hbase指定的hbase表单元添加值(put)



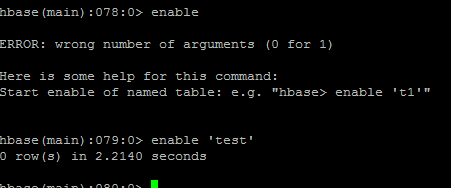
* 1. 获取指定表的相关信息(scan)



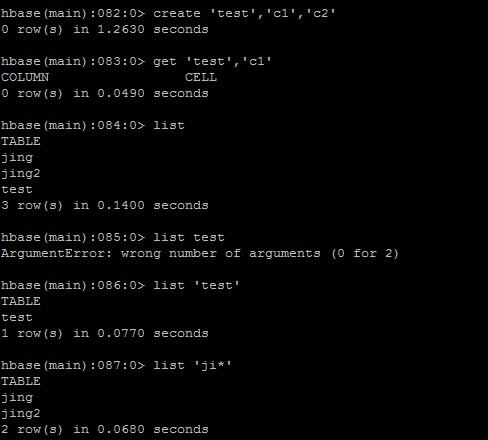
* 1. 使表失效(disable)

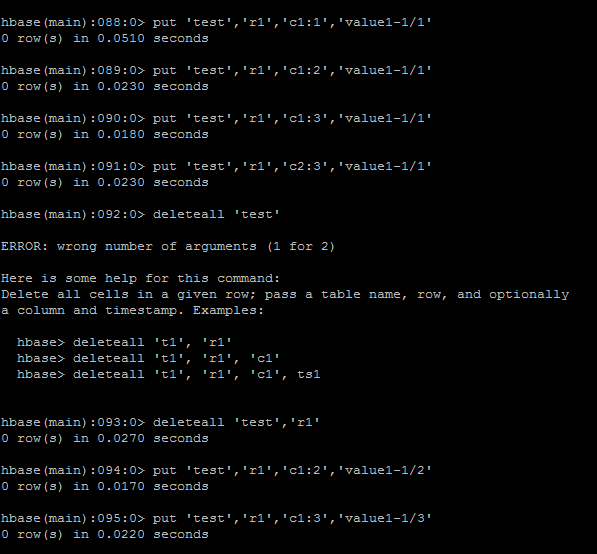


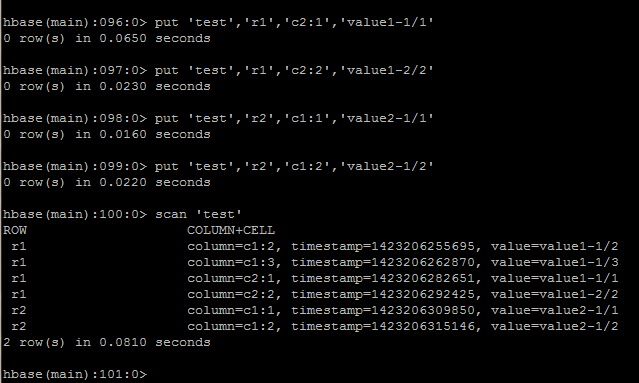
* 1. 使表生效(enable)

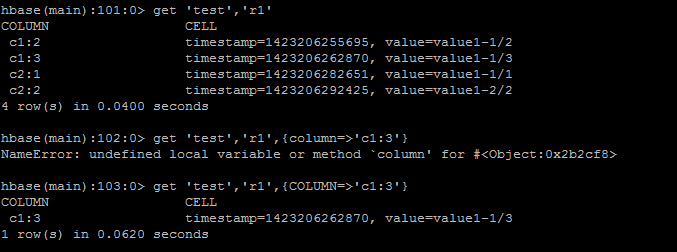


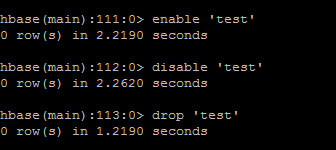
实例：











1. HBase 配置

<configuration>

<property>

<name>hbase.rootdir</name>

<value>hdfs://hadoop1:9000/hbase</value>

</property>

<property>

<name>hbase.cluster.distributed</name>

<value>true</value>

</property>

<property>

<name>hbase.zookeeper.quorum</name>

<value>hadoop1,hadoop2,hadoop3</value>

</property>

<property>

<name>hbase.master</name>

<value>hdfs://hadoop1:60000</value>

</property>

<property>

<name>hbase.client.write.buffer</name>

<value>2097152</value>

<description>设置缓冲区大小为2M</description>

</property>

<property>

<name>hbase.master.meta.thread.rescanfrequency</name>

<value>60000</value>

<description>hbase扫描ROOT和META表的时间间隔</description>

</property>

<property>

<name>hbase.regionserver.handler.count</name>

<value>10</value>

<description>RegionServer上等待处理请求的实例数目,默认10</description>

</property>

<property>

<name>hbase.hregion.max.filesize</name>

<value>268435456</value>

<description>HRegion上stove文件最大值,默认为256M，以字节为单位</description>

</property>

<property>

<name>hfile.block.cache.size</name>

<value>0.2</value>

<description>HFile/StoreFile缓存所在java虚拟机堆大小的百分比，默认0.2，占比20%,0为禁用</description>

</property>

<property>

<name>hbase.regionserver.global.memstore.upperLimit</name>

<value>0.4</value>

<description>RegionServer上所有的memstore所在java虚拟机的比例上限，默认为0.4,40%,当memstore所占空间超过此值，更新操作被阻塞，所有内容强制写出</description>

</property>

<property>

<name>hbase.hregion.memstore.flush.size</name>

<value>67108864</value>

<description>memstore缓存写入到磁盘上的值内容大小限度值</description>

</property>

</configuration>

1. HBase 的体系结构

主从服务器：HRegion服务器群和HMaster服务器构成。通过zookeeper协调HMaster不存储任何数据，hbase逻辑上的表可以分为多个HRegion，存储到HRegion服务器群中，HMaster中存储的是从数据到HRegion的映射。

HRegion服务器:HLOG部分和HRegion部分。HLOG是用来存储数据日志，先写日志方式;HRegion部分由很多HRegion组成，存储的是实际数据。

每个HRegion由很多Store组成，每个store存储的是一个列族下的数据。在每个hstore中包含一块memstore和多个storefile(HFile).

HFile：负责实际的数据存储，HBase的最小存储单元。

1. HBase数据模型(NoSql)
   1. 表(table)，是存储管理数据的
   2. 行键(row key)，类似于Mysql中的主键

行键是HBase表天然自带的

* 1. 列族(column family),列的集合

HBase中列族是需要在定义表时指定的，列是在插入数据时自动增加的

HBase表中的数据，每个列族淡出一个文件

* 1. 时间戳(timestamp),列(也称作标签，修饰符)的一个属性

行键和列确定的单元格，可以存储多个数据，每个数据含有时间戳属性，数据具有版本特性。

如果不指定时间戳或者版本，默认取最新的

* 1. 存储的数据都是字节数组。
  2. 表中的数据是按照行键的顺序物理存储的。

字符串、证书、二进制串甚至串行化的 都可以作为行键

表按照行键的“逐字节排序”顺序对行进行有序化处理

表内数据非常”稀疏”，不同的行的列数完全可以大不相同

可以只对一行上”锁”

对行的写操作始终是”原子的”

1. Hbase 的物理模型
   1. HBase是适合海量数据(20PB)的秒级简单查询的数据库
   2. Table在按照行键,分割为多个HRegion，一个region由(startKey,endKey)表示，每个HRegion分散在不同的RegionServer中
   3. HBase对表的操作转化为对多台Regionserver的并行查询
2. HBase的体系结构
   1. HBase是主从结构，HMaster、HregionServer允许有多个HMaster存在，但同一时间只有一个HMaster有效
   2. HBase集群是需要Zookeeper的：
      1. 保证任何时候，集群中只有一个running master
      2. 存贮所有Region的寻址路口
      3. 实时监控RegionServer的状态，将RegionServer的上线和下线信息，实时通知给Master
      4. 存储HBase的schema，包括有哪些table，每个table 有哪些列族

Master可以启动多个HMaster，通过zookeeper的Master Election机制保证总有一个Master运行。

* + 1. 为Region server分配region
    2. 负责region server的负载均衡
    3. 发现失效的region server 并小心分配其上的region
  1. Region Server
     1. 维护Master分配给它的region，处理对这些region的IO请求
     2. 负责切分在运行过程中变得过大的region
  2. HBase有两张特殊的表，-ROOT-和.META.
     1. .META.：记录了用户表的Region的信息，.META.可以有多个region
     2. -ROOT-：记录了.META.表的Region，-ROOT-只有一个region

1. HBase 与RDBMS
   * 1. 数据类型：HBase只字符串，所有类型交用户处理。RDBMS丰富的类型。
     2. 数据操作：HBase只有简单的插入、查询、删除、清空操作，表与表之间分离 ，没有复杂关系，不能实现表间关联。RDBMS含有各种函数，连接操作。
     3. 存储模式：HBase基于列存储，每列几个文件保存，不同列族文件分离。RDBMS基于表结构和行模式
     4. 数据维护：HBase的更新操作，实际是数据的新版本的插入操作。
     5. 可伸缩性：HBase轻松增减硬件数量，容错性高；RDBMS需要加上中间层实现。
2. 模式设计的原则
   * 1. 列族的数量及列族的势
        1. 建议列族的数量越少越好，hbase对两个及以上的列族处理的并不好，hbase的flushing和压缩是基于HRegion的。

同一个表中不同列族所存储的记录数量的差别，即列族的势。当两个列族数量差别过大时，会将包含记录较少的列族的数量分散到多个region上，而region可能存储在不同的regionserver上，这样进行查询或scan操作时，会对性能有影响。

* + 1. 行键的设计

避免使用时序或单调(递增、递减)行键，hbase根据行键确定存储位置，即region的位置，如果采用时序或单调，连续到来的数据会被存储到一个region中，而其他region是空闲的，这是分布式系统不希望看到的。

* + 1. 最小化行键和列族的大小

值🡨 行键、列、时间戳。

Hbase的索引是为了加快随机访问的速度：行键+列族:列+时间戳+值

* + 1. 版本的数量

默认情况下，每个数据存储3个版本，可通过HColumnDescriptor设置