Hbase

演讲人：

时间：2018.3

提纲

Hbase概述

HBase是一个开源的非关系型分布式数据库（NoSQL），它参考了谷歌的BigTable建模，实现的编程语言为 Java。它是Apache软件基金会的Hadoop项目的一部分，运行于HDFS文件系统之上，为 Hadoop 提供类似于BigTable规模的服务，可以存储海量稀疏的数据，并具备一定的容错性、高可靠性及伸缩性。主要应用场景是实时随机读写超大规模的数据。

HBase在列上实现了BigTable论文提到的压缩算法、内存操作和布隆过滤器。HBase的表能够作为MapReduce任务的输入和输出，可以通过Java API来存取数据，也可以通过REST、Avro或者Thrift的API来访问。

HBase不能取代RDBMS，因为二者的应用场景不同。HBase为了解决海量数据的扩展性，支持简单的增加节点来实现线性扩展，从而在集群上管理海量的非结构化或半结构化的稀疏数据。HBase仅能通过主键（raw key）或主键的range检索数据，支持单行事务。

Hbase概述

Hbase的特点：

- 大：一个表可以有上亿行，上百万列。
- 面向列：面向列表（簇）的存储和权限控制，列（簇）独立检索。
- 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。
- 无模式：每一行都有一个可以排序的主键和任意多的列，列可以根据需要动态增加，同一张表中不同的行可以有截然不同的列。
- 数据多版本：每个单元中的数据可以有多个版本，默认情况下，版本号自动分配，版本号就是单元格插入时的时间戳。
- 数据类型单一：HBase中的数据都是字符串，没有类型。

Hbase概述

HBase的优势主要在以下几方面：

- 海量数据存储。
- 快速随机访问。
- 大量写操作的应用。

常见的应用场景：

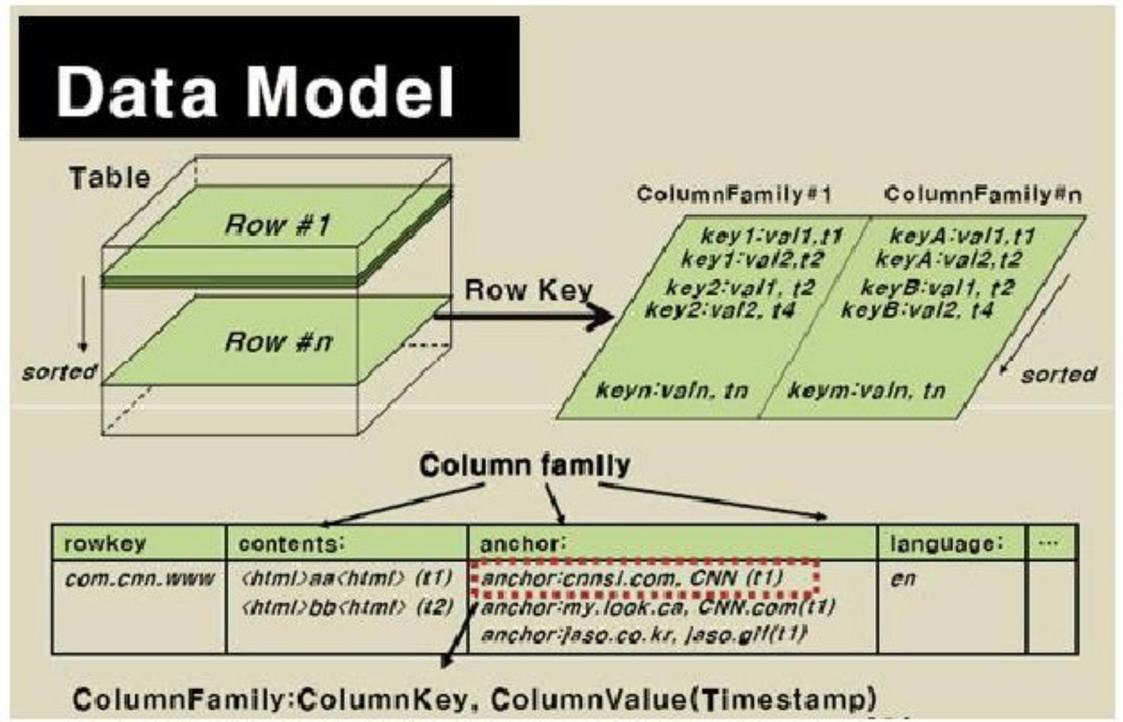
- 互联网搜索引擎数据存储（BigTable要解决的问题）
- 审计日志系统
- 实时系统
- 消息中心
- 内容服务系统

提纲

1. Hbase概述
2. Hbase数据模型
3. Hbase架构
4. Hbase安装
5. Hbase基本操作

Hbase数据模型

Hbase一般以表的形式存储数据。表由行和列组成。列划分为若干个列族（row family），如下图：



Hbase数据模型

Hbase不存储空白数据，所以Hbase逻辑模型中的空白在物理磁盘中是实际不存在。逻辑数据和实际存储数据区别如下图：

HBase的逻辑数据模型：

Row Key	Time Stamp	Column "contents:"	Column "anchor:"		Column "mime:"
"com.cnn.www"	t9		"anchor:cnnsi.com"	"CNN"	
	t8		"anchor:my.look.ca"	"CNN.com"	
	t6	"<html>..."			"text/html"
	t5	"<html>..."			
	t3	"<html>..."			

HBase的物理数据模型（实际存储的数据模型）：

Row Key	Time Stamp	Column "contents:"
"com.cnn.www"	t6	"<html>..."
	t5	"<html>..."
	t3	"<html>..."

Row Key	Time Stamp	Column "anchor:"	
"com.cnn.www"	t9	"anchor:cnnsi.com"	"CNN"
	t8	"anchor:my.look.ca"	"CNN.com"

Row Key	Time Stamp	Column "mime:"
"com.cnn.www"	t6	"text/html"

Hbase数据模型

Hbase数据模型主要有5部分组成：

- 行健（Row Key）：表的主键，表中的记录默认按照行健升序排序。
- 时间戳（Timestamp）：每次数据操作对应的时间戳，可以看作是数据的版本号。
- 列族（Column Family）：表在水平方向有一个或者多个列族组成，一个列族中可以由任意多个列组成，列族支持动态扩展，无需预先定义列的数量以及类型，所有列均以二进制格式存储，用户需要自行进行类型转换。所有的列族成员的前缀是相同的，例如“abc:a1”和“abc:a2”两个列都属于abc这个列族。
- 表和区域（Table&Region）：当表随着记录数不断增加而变大后，会逐渐分裂成多份，成为区域，一个区域是对表的水平划分，不同的区域会被Master分配给相应的RegionServer进行管理。
- 单元格（Cell）：表存储数据的单元。由{行健，列（列族:标签），时间戳}唯一确定，其中的数据是没有类型的，以二进制的形式存储。

Hbase数据模型

- **Row Key**

NoSQL 数据库一样，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 全表扫描。

3)Row Key 可以使任意字符串（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。

在存储时，数据按照*Row Key 的字典序（byte order）排序存储*。设计 Key 时，要充分排序存储这个特性，将经常一起读取的行存储到一起（位置相关性）。

注意 字典序对 int 排序的结果是 1，10,100,11,12,13,14,15,16,17,18,19,20,21，…， 9，91,92,93,94,95,96,97,98,99。要保存整形的自然序，Row Key 必须用 0 进行左填充。

行的一次读写是原子操作（不论一次读写多少列）。这个设计决策能够使用户很容易理解程序在对同一个行进行并发更新操作时的行为。

Hbase数据模型

- **时间戳**

HBase 中通过 Row 和 Columns 确定的一个存储单元称为 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 可以由客户显示赋值。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管理（包括存储和索引）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（比如最近七天）。用户可以针对每个列族进行设置。

Hbase数据模型

- **列族**

HBase 表中的每个列都归属于某个列族。列族是表的 Schema 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。在实际应用中，列族上的控制权限能帮助我们管理不同类型的应用， 例如，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）。

- **Cell**

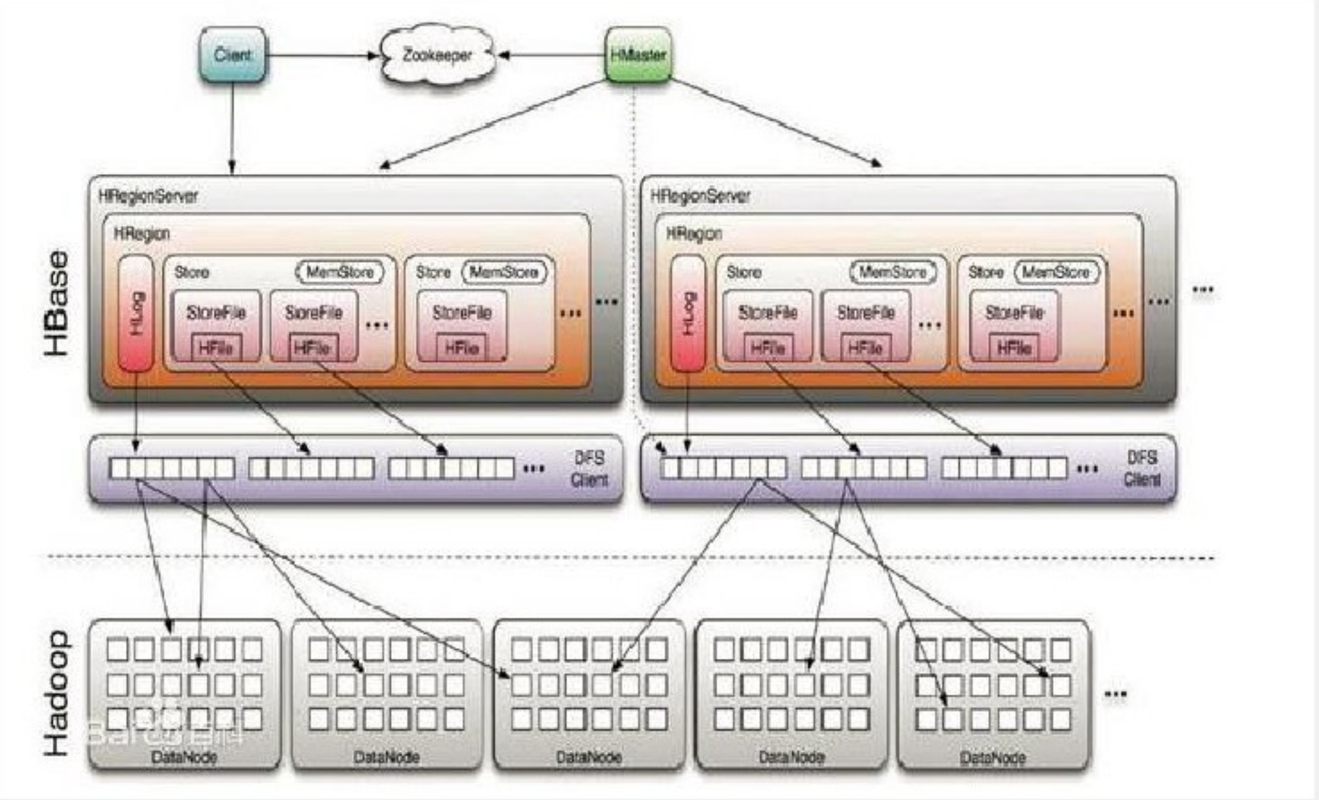
Cell 是由 {row key， column(=< family> + < label>)， version} 唯一确定的单元。Cell 中的数据是没有类型的，全部是字节码形式存储。

提纲

1. Hbase概述
2. Hbase数据模型
3. **Hbase架构**
4. Hbase安装
5. Hbase基本操作

Hbase架构

HBase架构中只有一个Master节点，称HMaster，还有多台RegionServer成为HRegionServer，每个RegionServer包含多个Region。



Hbase架构

Hbase主要组件：

- HBase访问接口：Java，REST，Thrift。
- Master：集群的管理服务器，为RegionServer分配Region，负责RegionServer的负载均衡，处理schema更新请求。
- RegionServer：管理HBase的数据存储，维护Region，处理IO请求。
- Zookeeper：保证集群的高可用性、存储Region的寻址入口，并实时监控RegionServer的状态，存储HBase的Schema。

client访问hbase上数据的过程并不需要Master参与（寻址访问Zookeeper和RegionServer，数据读写访问RegionServer），Master仅仅维护Table和Region的元数据信息，负载很低。

提纲

1. Hbase概述
2. Hbase数据模型
3. Hbase架构
4. **Hbase安装**
5. Hbase基本操作

Hbase安装：环境准备

Hbase环境准备：

1：Jdk1.6+

2：hadoop2.x

3：SSH服务开启

4：Hbase

下载地址：http://mirrors.cnnic.cn/apache/hbase/

Hbase安装：单机模式

1:解压Hbase安装包

```
tar zxvf hbase-0.98.11-hadoop2-bin.tar.gz
mv hbase-0.98.11-hadoop2 /opt/hbase
chmod 777 /opt/hbase
```

2:配置 hbase-site.xml

在运行之前，我们需要对HBase进行相关配置。建议大家修改 ${HBase-Dir}/conf/hbase-site.xml 文件，因为即使你修改了hbase-default.xml文件，也会被hbase-site.xml中的配置所覆盖。也就是说，最终是以 hbase-site.xml 中的配置为准的

```
<configuration>
    <property>
        <name>hbase.rootdir</name>
        <value>file:///tmp/hbase-${user.name}/hbase</value>
    </property>
</configuration>
```

注意：修改 ${user.name}为你自己的 hadoop 用户名

Hbase安装：伪分布模式

1:按单机模式安装Hbase

2:配置 hbase-site.xml

```
<configuration>
    <property>
        <name>hbase.rootdir</name>
        <value>hdfs://localhost:9000/hbase</value>
    </property>
    <property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
    </property>
</configuration>
```

hbase.rootdir：该参数制定了HReion服务器的位置，即数据存放的位置。主要端口号要和Hadoop相应配置一致。

hbase.cluster.distributed：HBase的运行模式。false是单机模式，true是分布式模式。若为false，HBase和Zookeeper会运行在同一个JVM里面。默认为false.

Hbase安装：伪分布模式

3:设置环境变量

修改HBase下的conf目录中的hbase-env.sh文件

```
export JAVA_HOME=/usr/local/jdk1.7.0_67
export HBASE_MANAGES_ZK=true
```

export HBASE_MANAGES_ZK=true 表示设置由hbase自己管理zookeeper，不需要单独的zookeeper，本文搭建的 Hbase 用的是自带的 zookeeper，故设置为true.

4:修改profile

```
vim /etc/profile
修改 /etc/profile 文件：
# set hbase path
export HBASE_HOME=/opt/hbase
export PATH=$PATH:/opt/hbase/bin
```

Hbase安装：完全分布模式

1:完全配置好的hadoop集群
2:按单机模式安装Hbase
3:配置 hbase-site.xml
4:修改regionservers，在regionservers文件中添加如下内容

hadoop1
hadoop2

hbase-site.xml 配置信息如下：

Hbase安装：完全分布模式

```
<configuration>
    <property>
        <name>hbase.rootdir</name>
        <value>hdfs://localhost:9000/hbase</value>
    </property>
    <property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
    </property>
    <property>
       <name>hbase.zookeeper.quorum</name>
       <value>hadoop1,hadoop2</value>
     </property>
    <property>
       <name>hbase.zookeeper.property.dataDir</name>
       <value>/home/hadoop1/hadoopdata/zookeeper</value>
     </property>
</configuration>
```

Hbase安装：完全分布模式

在上面的配置文件中，第一个属性指定本机的hbase的存储目录；第二个属性指定hbase的运行模式，true代表全分布模式；第三和第四个属性是关于Zookeeper集群的配置。我的Zookeeper安装在hadoop1和hadoop2上。

提纲

1. Hbase概述
2. Hbase数据模型
3. Hbase架构
4. Hbase安装
5. Hbase基本操作

Hbase基本操作

- Hbase Shell:

1. 连接HBase

   使用hbase shell命令来连接正在运行的Hbase实例。

   ```
   ./bin/hbase shell
   ```

2. 显示HBase Shell 帮助文档

   输入help并按Enter键，可以显示HBase Shell的基本使用信息

3. 退出HBase Shell

   使用quit命令，退出HBase Shell 并且断开和集群的连接，但此时HBase仍然在后台运行

4. 查看HBase状态

   ```
   status
   ```

5. 关闭HBase

Hbase基本操作

```
./bin/stop-hbase.sh
```

- 数据定义（DDL）操作

1：创建新表

使用create命令来创建一个新的表。在创建的时候，必须指定表名和列族名。

```
create 'test', 'cf '
```

2：列举表信息

使用list命令

```
list 'test '
```

3：获取表描述

使用describe命令

```
describe 'test'
```

Hbase基本操作

4：删除表

删除表之前，先disable表,再使用drop命令实现删除表的功能

```
drop 'test'
```

5：检查表是否存在

使用exists命令

```
exists 'member'
```

- 数据管理（DML）操作

1：向表中插入数据

使用put命令，将数据插入表中：

```
put 'test', 'row1', 'cf:a', 'value1 ‘
```

2：一次性扫描全表数据

一种获取HBase数据的方法是扫描，使用scan命令来扫描表的数据。（可以限制扫描的范围）

Hbase基本操作

3：获取一个行数据

使用get命令来获得某一行的数据：

```
get 'test', 'row1 '
```

4：禁用一个表

如果你想要删除一个表或是修改它的设置，或者是其它的情况，都需要首先禁用该表。使用disable命令禁用表，enable命令重新启用表。

```
disable 'test '
enable 'test '
```

5：删除数据

删除行中的某个列值

语法： delete <table>, <rowkey>, <family:column> , <timestamp>

```
delete 't1','rowkey001','cf:a'
```

Hbase基本操作

删除行

语法：deleteall <table>, <rowkey>, <family:column> , <timestamp>

```
delete 'test','row1 '
```

删除表中的所有数据

语法： truncate <table>

```
truncate 'test'
```

THANKS