一页纸精华 | HBase

原创 2016-03-10 牛家浩 中兴大数据

>>>> 这是 **中兴大数据** 第**214**篇原创文章

要入门大数据，最好的办法就是理清Hadoop的生态系统。中兴大数据公众号将推出“一页纸精华”栏目，将用最精炼的语言，陆续为你介绍Hadoop生态系统的各个组件。本期为你介绍Hadoop分布式数据库HBase。

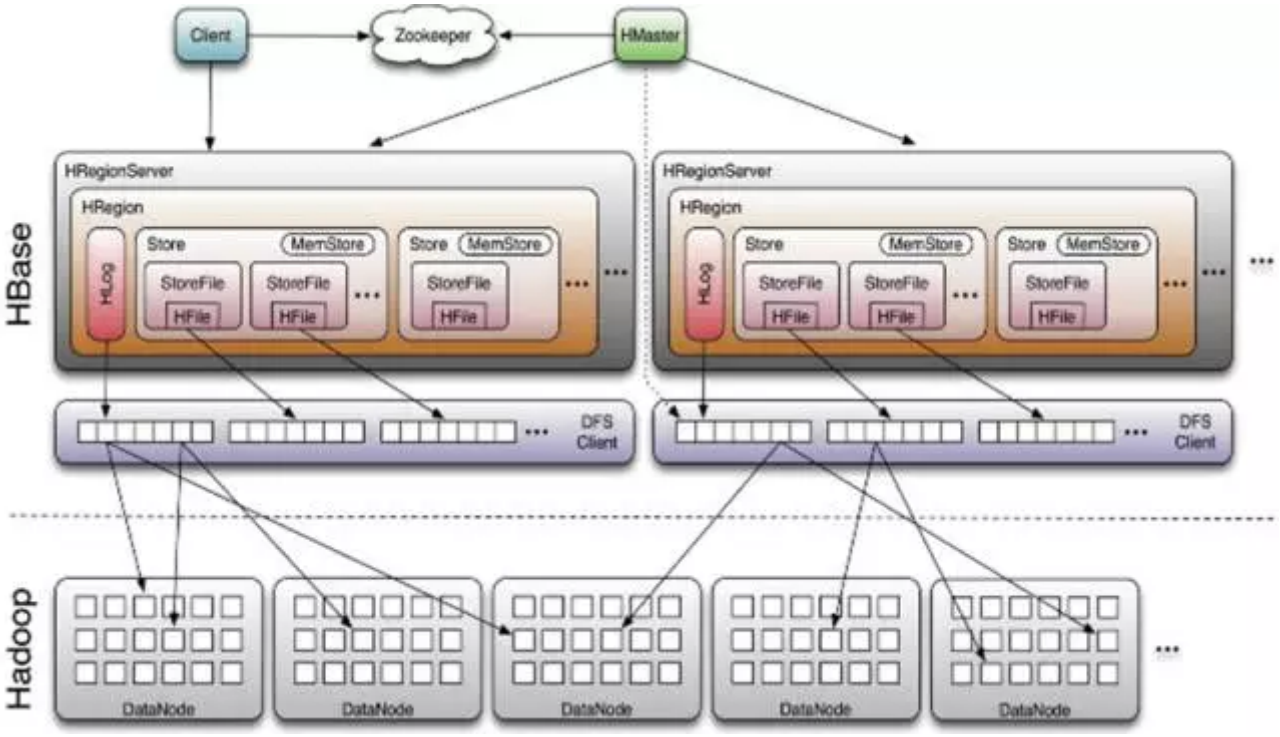
HBase建立在HDFS之上，提供高可靠性、高性能、列存储、可伸缩、实时读写的数据库系统。它介于NoSQL和RDBMS之间，仅能通过行键(row key)和行键序列来检索数据，仅支持单行事务(可通过Hive支持来实现多表联合等复杂操作)。主要用来存储非结构化和半结构化的松散数据。与Hadoop一样，HBase目标主要依靠横向扩展，通过不断增加廉价的商用服务器，来增加计算和存储能力。

HBase表一般有这样的特点：

- **大**：一个表可以有上亿行，上百万列
- **面向列**：面向列(族)的存储和权限控制，列(族)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。

HBase体系架构

HBase的服务器体系结构遵循简单的主从服务器架构。它由HRegion Server和HMaster组成，HMaster负责管理所有的HRegion Server，HBase中所有的服务器都通过ZooKeeper来协调。HBase的体系结构如下图所示。



HBase体系结构图

Client

HBase Client使用RPC与HMaster和HRegionServer通信。对于管理类操作，Client与HMaster通信，对于数据读写类操作，与HRegionServer通信。Client访问HBase上数据的过程并不需要HMaster参与（寻址访问Zookeeper和HRegion Server，数据读写访问HRegion Server），HMaster仅仅维护表和HRegion的元数据信息，负载很低。

HMaster

- 为HRegion Server分配HRegion
- 负责HRegion Server的负载均衡
- 发现失效的HRegion Server并重新分配其上的HRegion
- 处理元数据更新请求

HRegion Server

- HRegion Server维护HMaster分配给它的HRegion，处理对这些HRegion的I/O请求
- HRegion Server负责切分在运行过程中变得过大的HRegion

ZooKeeper

- 保证任何时候，集群中只有一个HMaster
- 存储所有HRegion的寻址入口
- 实时监控HRegion Server的状态，将HRegion Server的上线和下线信息实时通知给HMaster

HBase Shell

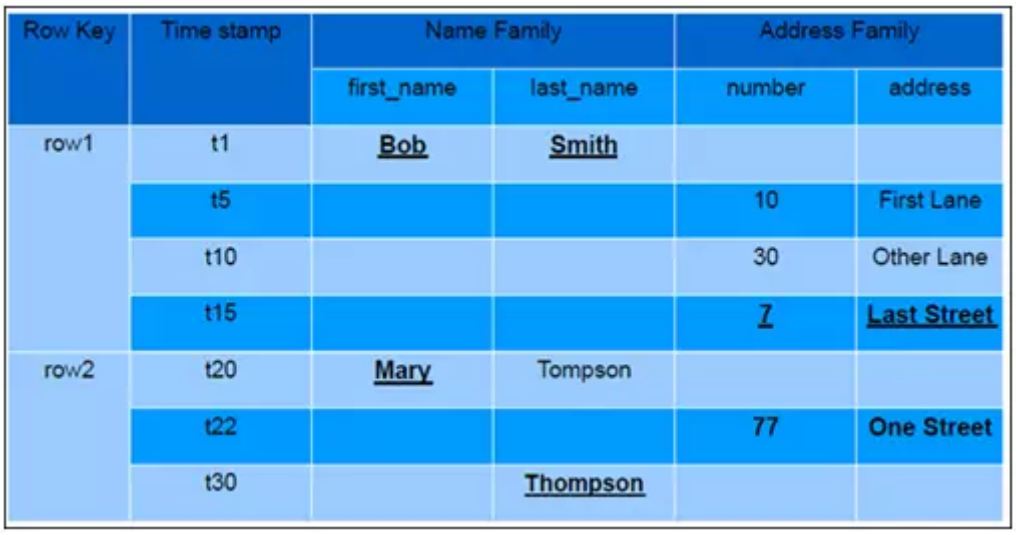
HBase Shell 提供了大量的 HBase 命令，通过HBase Shell 用户可以方便地创建、删除及修改表，还可以向表中添加数据、列出表中的相关信息等。

HBase架构之美

- **LSM**：解决磁盘随机写问题(顺序写才是王道)；采用LSM树（Log-Structured Merge Tree）作为存储引擎,支持增、删、读、改、顺序扫描。将对数据的修改增量保持在内存中，达到指定的大小限制后将这些修改操作批量写入磁盘，读取的时候稍微麻烦，需要合并磁盘中历史数据和内存中最近修改操作，写入性能大大提升，读取时可能需要先看是否命中内存，否则需要访问较多的磁盘文件。LSM树和B+树相比，LSM树牺牲了部分读性能，用来大幅提高写性能。
- **HFile**：解决数据索引问题(只有索引才能高效读)；
- **WAL**：解决数据持久化(面对故障的持久化解决方案)；
- **ZooKeeper**：解决核心数据的一致性和集群恢复；
- **HDFS**：使用HDFS存储，解决数据副本和可靠性问题。

HBase表逻辑视图

HBase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(column family)。HBase表逻辑结构如下图所示。



Row Key	Time stamp	Name Family		Address Family	
		first_name	last_name	number	address
row1	t1	Bob	Smith		
	t5			10	First Lane
	t10			30	Other Lane
	t15			7	Last Street
row2	t20	Mary	Tompson		
	t22			77	One Street
	t30		Thompson		

HBase表逻辑结构图

行键

与NoSQL数据库一样，行键（row key）是用来检索记录的主键。访问HBase表中的行，只有三种方式：

- 通过单个行键访问
- 通过行键序列访问
- 全表扫描

行键可以是任意字符串(最大长度是 64KB，实际应用中长度一般为 10-100Bytes)，在HBase内部，行键保存为字节数组。

存储时，数据按照行键的字典序排序存储。设计行键时，可以充分利用排序存储这个特性，将经常一起读取的行存储放到一起。

列族

HBase表中的每个列，都归属某个列族。列族是表元数据的一部分(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）。

时间戳

HBase中通过行键和列确定的一个存贮单元称为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。

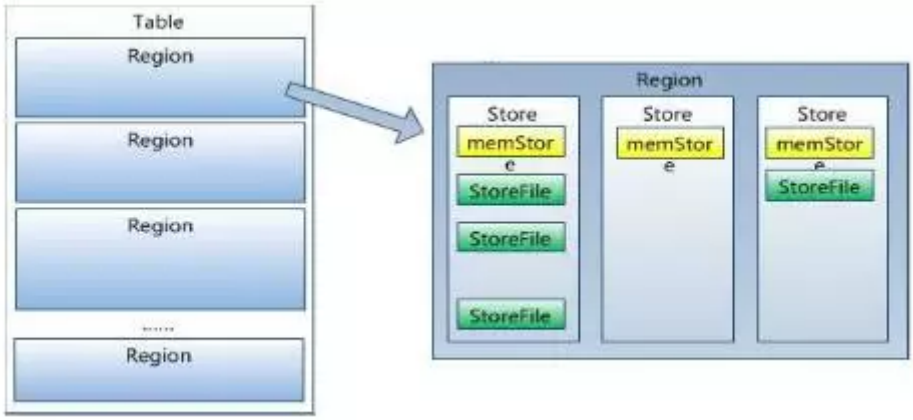
为了避免数据存在过多版本造成的的管理 (包括存贮和索引)负担，HBase提供了两种数据版本回收方式。一是保存数据的最后n个版本，二是保存最近一段时间内的版本（比如最近七天）。用户可以针对每个列族进行设置。

单元

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

HBase表物理存储

表中的所有行都按照行键的字典序排列，表在行的方向上分割为多个HRegion。如下图所示：

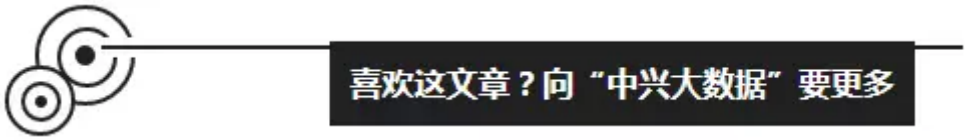


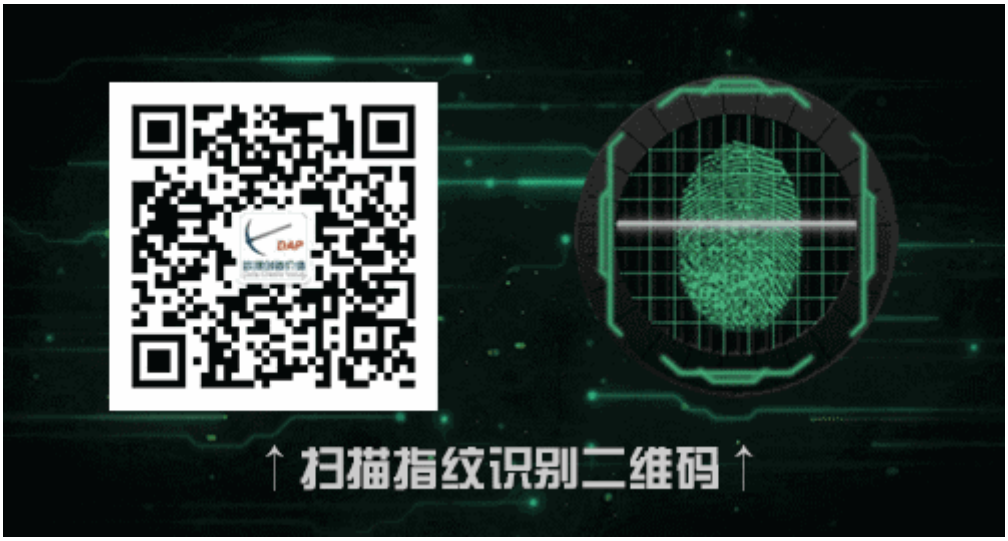
HBase物理存储示意图

HRegion按大小可以分割，每一个表一开始只有一个HRegion，随着数据不断插入表，HRegion不断增大，当增大到一定阈值的时候，HRegion就会等分为两个新的HRegion。当表中的行不断增多，就会有越来越多的HRegion。

HRegion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HRegion可以分布在不同的HRegion Server上。但是一个HRegion是不会拆分到多个HRegion Server上。

HRegion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，HRegion由一个或者多个Store组成，每一个Store都保存一个列族。每一个Store又由一个memStore和多个StoreFile组成。StoreFile以HFile格式保存在HDFS上。




↑扫描指纹识别二维码↑