云计算与大数据概论

HBase简介与原理

目标

- 掌握HBase的概念和作用
- 了解HBase使用场景和成功案例
- 了解HBase和传统关系数据库的对比分析
- 掌握HBase数据模型
- 掌握HBase的组成架构
- 掌握HBase的安装运行方法
- 了解HBase的访问接口

HBase的概念和作用

HBase-Hadoop DataBase是一个高性能、高可靠性、面向列、可伸缩的分布式存储系统，使用HBase技术，可以在廉价PC服务器上搭建起大规模结构化存储集群。

HBase是Google BigTable的开源实现，它模仿并提供了基于Google文件系统的BigTable数据库的所有功能：HBase使用Hadoop HDFS作为其文件存储系统；使用Hadoop MapReduce来处理HBase中的海量数据，使用ZooKeeper作为协同服务。

此外，Pig和Hive还为HBase提供了高层语言支持，使得在HBase上进行数据统计变得非常简单；Sqoop则为HBase提供了方便的传统关系数据库数据导入功能，使得传统数据库数据向HBase中迁移变得非常方便。

HBase的设计目的是处理非常庞大的表，甚至能使用普通的计算机处理超过10亿行的、由数百万列元素组成的数据表的数据。

HBase使用场景和成功案例——互联网搜索功能

搜索含有特定词语的文档，就需要查找索引，该索引提供了特定词语和包含该词语的所有文档的映射，因此，为了能够搜索，首先必须建立索引。Google和其他搜索引擎正是这么做的，它们的文档库是整个互联网，搜索的特定词语就是在搜索框里输入的任何东西。

BigTable及其开源仿制品HBase为这种文档库提供存储：BigTable提供行级访问，因此爬虫可以在BigTable中插入和更新单个文档，每个文档则保存为BigTable中的一行。

HBase的设计初衷之一就是用来存储互联网持续更新的网页副本，该功能在互联网服务的其它领域也同样适用。例如，由于HBase能完成从个人之间的通信信息存储到通信信息分析等多种任务，因而成为了Facebook、Twitter和StumbleUpon等多家社交网络公司的关键基础设施，在这一领域，HBase拥有众多的应用场景——抓取增量数据，内容服务，信息交换。

HBase使用场景和成功案例——抓取增量数据

HBase会抓取来自各种数据源的增量数据，这种数据源可能是网页爬虫，也可能是记录用户看了什么广告和看了多长时间广告的广告效果数据，还可能是记录各种参数的时间序列数据。

下面介绍HBase在该领域的几个成功案例：

1. 抓取监控指标：OpenTSDB

服务数百万用户的Web产品的后台基础设施一般都有数百或数千台服务器，这些服务器承担了各种功能，包括服务流量、抓取日志、存储数据和处理数据等。为了保证产品正常运行，监控这些服务器及其上面运行的软件的健康状态是至关重要的。而对整个环境进行大规模监控，需要能够采集和存储来自不同数据源的各种监控指标的监控系统，针对此问题，每个公司都有自己的解决方案，一些公司使用商业工具来收集和展示监控指标，而另外一些公司则使用开源框架。

StumbleUpon创建了一个开源框架OpenTSDB，它是Open Time Series DataBase（开放时间序列数据库——按照时间收集监控指标一般被称为时间序列数据，即按照时间顺序收集和记录的数据）的缩写。OpenTSDB收集服务器的各种监控指标，并使用HBase作为核心平台来存储和检索所收集的监控指标。StumbleUpon创建OpenTSDB是为了拥有一个可扩展的监控数据收集系统，一方面，该框架能够存储和检索监控指标数据并保存很长时间；另一方面，如果需要增加功能，该框架也可以添加各种新的监控指标。StumbleUpon使用OpenTSDB监控所有基础设施和软件，也包括HBase集群自身。

2. 抓取用户交互数据：Facebook和StumbleUpon

抓取监控指标是一种常见的HBase应用场景，另一种常见的应用场景则是抓取用户交互数据——例如，谁看了什么，或是某个按钮被点击了多少次。抓取和分析这些数据可以跟踪数百万用户在网站上的活动，以了解哪一个网站功能最受欢迎，或者怎样让某一次网页浏览直接影响到下一次，等等。Facebook和Stumble里的“Like（喜欢）”按钮和StumbleUpon里的“+1”按钮本质上是一个计数问题——用户每次“Like”一个特定主题，计数器就增加一次。

StumbleUpon最初使用的是MySQL技术，但随着网站服务越来越庞大，用户在线负载需求急剧增长，远远超过了MySQL集群的处理能力。最终，StumbleUpon选择使用HBase来替换MySQL集群，但是，当时的HBase尚不能直接提供其必需的功能，于是StumbleUpon对其作了一些小的开发改动。

FaceBook使用HBase的计数器来计量用户对特定网页的“Like”次数。使网页内容创作者和网页主人可以得到接近实时的用户偏好数据信息，从而更精准地判断应该为用户提供什么内容。为此，Facebook还创建了一个名为Facebook Insights的系统，由于该系统需要一个可扩展的存储系统，Facebook考虑了很多可能的选择，包括关系型数据库管理系统、内存数据库和Cassandra数据库，最后决定使用HBase。基于HBase，Facebook可以很方便地横向扩展服务规模，给数百万用户提供服务，并可继续沿用自身运行大规模HBase集群的已有经验。如今，Facebook Insights每天处理数百亿条事件，记录数百个监控指标。

3. 遥测技术：Mozilia和Trend Micro

软件崩溃报告是非常有用的软件运行数据，经常用于探究软件质量和规划软件开发路线图，而使用HBase，可以成功地捕获和存储用户计算机上生成的软件崩溃报告。

Mozilla基金会旗下的主要软件产品有二——FireFox网络浏览器和Thunderbird电子邮件客户端，这些软件产品安装在全世界数百万台计算机上，支持各种操作系统，当这些软件崩溃时，会以Bug报告的形式返回一个软件崩溃报告给Mozilla。那么Mozilla是如何收集这些数据的？收集后又是如何使用的呢？答案是：Mozilla的Socorro的系统收集了这些报告，用于指导研发部门研制更稳定的产品，而Socorro系统的数据存储和分析功能则建构在HBase上。

Trend Micro为企业客户提供互联网安全服务。安全服务的重要环节是感知，而日志收集和分析对于提供这种感知能力至关重要。为此，Trend Micro使用HBase来管理网络信誉数据库，该数据库有点像Socorro系统，需要对行级更新和MapReduce批处理的支持，而HBase被用来收集和分析日志活动，每天可以收集数十亿条记录。HBase灵活的数据模式允许数据结构出现变化，因此当分析流程重新调整时，允许Trend Micro增加新属性。

4. 广告效果和点击流

过去的十年，在线广告成为互联网产品的一个主要收入来源，主流模式为先提供免费服务给用户，在用户使用服务的同时，再投放广告给目标用户。这种精准投放需要对用户交互数据进行充分的捕获和详细的分析，以便理解用户的特征，再基于这种特征，选择并投放广告。精细的用户交互数据有利于塑造更好的用户特征模型，进而产生更好的广告投放效果，并获得更多的收入。用户交互数据有两个特点：一是往往以连续流的形式出现，二是很容易按用户划分。理想情况下，用户交互数据一旦产生就应该能够马上使用，使用户特征模型可以没有延迟地持续优化。

HBase使用场景和成功案例——内容服务

传统数据库最主要的使用场景之一是为用户提供内容服务。用户希望使用和交互的内容种类越来越多，互联网以及终端设备的迅猛增长，对这些应用的接入方式也提出了更高的要求，因为各种各样的终端设备需要以不同的格式使用同样的内容。

1. URL短链接

URL短链接指将长的URL转换为短的URL，这种URL短链接在最近非常流行，许多公司都推出了类似的产品，而StumbleUpon使用名字为su.pr.的短链接产品，该产品以HBase为基础，可以用来缩短URL，存储大量的短链接与和原始长链接的映射关系，而HBase则帮助这个产品实现扩展能力。

2. 用户模型服务

用户模型是对网站目标群体真是特征的勾勒，是真实用户的虚拟代表。建立用户模型可以减少主观猜测，走近用户，理解他们真正的需要，从而能更好的为不同类型的用户服务。

经HBase处理过的内容往往不直接提交给用户使用，而是用来决定应该提交给用户什么内容，这种中间数据常用于丰富用户的交互。前面提到的广告服务场景里的用户特征（或者说模型）就来自HBase，这类模型多种多样，可用于多种不同场景，例如决定针对特定用户投放什么广告；决定用户在电商网站购物时的实时报价；决定用户用搜索引擎检索时增加的背景信息和关联内容等。

3. 信息交换

随着各种社交网站不断涌现，世界正变得越来越小。社交网站的一个重要作用就是帮助人们进行互动，这种互动有时在群组内发生（小规模和大规模），有时在两个人之间发生，实际上，每时每刻都有数亿人正在通过社交网络进行对话。但是，仅和远处的人对话并不足以让用户满意，他们还希望能查看和其他人对话的历史记录，而让社交网站公司感到幸运的是，大数据领域的创新让保存和使用这些历史记录变得十分廉价。

在这方面，Facebook的短信系统是个经常被讨论的案例——当你使用Facebook时，你可以随时收到短信，或者发送短信给你的朋友。Facebook的这一特性完全依赖于HBase，因为用户读写的所有短信都存储在HBase里。Facebook的短信系统要求具备高的写吞吐量、极大的表以及数据中心内的强一致性，而HBase则是一个理想的解决方案，因为它具备上述所有特性，且拥有一个活跃的用户社区。

在HBaseCon 2012大会上，Facebook的工程师分享了一些惊人的数据：在Facebook平台上每天会交换数十亿条短信，带来大约750亿次操作。在高峰时刻，Facebook的HBase集群平均每秒发生150万次操作。从数据规模角度看，Facebook的HBase集群每月增加250TB的新数据，这可能是已知最大的HBase部署，无论是在服务器的数量方面，还是服务器所承载的用户量方面。某种意义上说，Facebook的短信系统也极大地推动了HBase的发展。

上述案例解释了HBase如何解决一些有趣的老问题和新问题，它们揭示的一个共同点是HBase可以对相同数据进行在线服务和离线处理，而这正是HBase的独特之处。

HBase和传统关系数据库的对比分析-1

HBase与传统关系数据库存在很大区别，它按照BigTable模型开发，是一个稀疏的、分布式的、持续多维度的排序映射数组。HBase是一个基于列模式的映射数据库，它只能表示很简单的“键-数据”映射关系，因而大大简化了传统的关系数据库。

传统关系数据库基本都具备以下特点：

- 面向磁盘存储和索引结构。
- 多线程访问。
- 基于锁的同步访问机制。
- 基于日志记录的恢复机制。

HBase和传统关系数据库的对比分析-2

而HBase和传统关系数据库的具体区别如下：

- 数据类型：HBase只有简单的字符串类型，所有其他类型都由用户自己定义，它只保存字符串，而关系数据库有丰富的数据类型和存储方式。
- 数据操作：HBase只提供很简单的插入、查询、删除、清空等操作，且HBase的表和表之间是分离的，没有复杂的表表间关系，也没必要实现表和表之间的关联等操作，而传统的关系数据库通常有各种各样的函数和连接操作。
- 存储模式：HBase是基于列存储的，几个文件保存在一个列族中，不同列族的文件是分离的，而传统的关系数据库是基于表格结构和行模式保存的。
- 数据维护：HBase的更新其实不是更新，只是一个主键或者列对应的新版本，其旧有的版本仍然会保留，所以实际上只是插入了新的数据，而不是传统关系数据库里的替换修改。
- 可伸缩性：HBase能够轻易地增加或者减少（在硬件错误的时候）硬件数量，且对错误的兼容性较高，而传统的关系数据库通常需要增加中间层才能实现类似的功能。

- 相比之下，BigTable和HBase这类基于列模式的分布式数据库显然更适应海量存储和互联网应用的需求：首先，灵活的分布式架构使其可以利用廉价的硬件设备组建庞大的数据仓库；其次，互联网应用是以字符为基础的，而BigTable和HBase正是针对这些应用而开发出来的数据库。

HBase数据模型

HBase中，表的索引是行关键字、列关键字和时间戳，每个值是一个不加解释的字符数组，数据则都是字符串，没有其他类型。HBase数据实例的模型如表:

RowKey	Timestamp	Column Family	
		URI	Parser
r1	t3	url=http://www.taobao.com	title=天天特价
	t2	host=taobao.com	
	t1		
r2	t5	url=http://www.alibaba.com	content=每天...
	t4	host=alibaba.com	

HBase数据模型——相关概念

在HBase数据模型中，存在以下三个重要概念：

- 行键（RowKey）：HBase表的主键，表中的记录按照行键排序。
- 时间戳（Time Stamp）：每次数据操作对应的时间戳，可以看作数据的版本号。
- 列族（Column Family）：表在水平方向由一个或者多个列族组成，一个列族则可以由任意多个列组成，即列族支持动态扩展，无需预先定义列的数量及类型，所有列均以二进制格式存储，用户需要手动进行类型转换。

HBase数据模型——概念视图

可以将一个HBase表看作一个大的映射关系，通过主键，或者主键+时间戳，就可以定位一行数据，由于是稀疏数据，所以某些列可以是空白的。HBase表所存储数据的概念视图如表:

RowKey	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>c..."			"text/html"
	t5	"<html>b..."			
	t3	"<html>a..."			

该表是一个Web网页数据的存储片断，其中，行键名是一个反向URL（即com.cnn.www）；contents列族用来存放网页内容；anchor列族存放引用该网页的锚链接文本；CNN的主页被Sports Illustrater（即SI，CNN的王牌体育节目）和MY-look的主页引用，因此该行包含了名为“anchor:cnnsi.com”和“anchhor:my.look.ca”的两个列；每个网页的锚链接只有一个版本（由时间戳标识，如t9、t8），而contents列则有三个版本，分别由时间戳t3，t5和t6标识。

HBase数据模型——物理视图

虽然从概念视图来看，HBase中的每个表是由很多行组成的，但是，在物理存储时，它是按照列来保存的，例如，上表的概念视图在物理存储时应该呈现出类似于下面表的形态

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

RowKey	Time Stamp	Column "contents:"
"com.cnn.www"	t6	"<html>c..."
	t5	"<html>b..."
	t3	"<html>a..."

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

HBase数据模型——物理存储

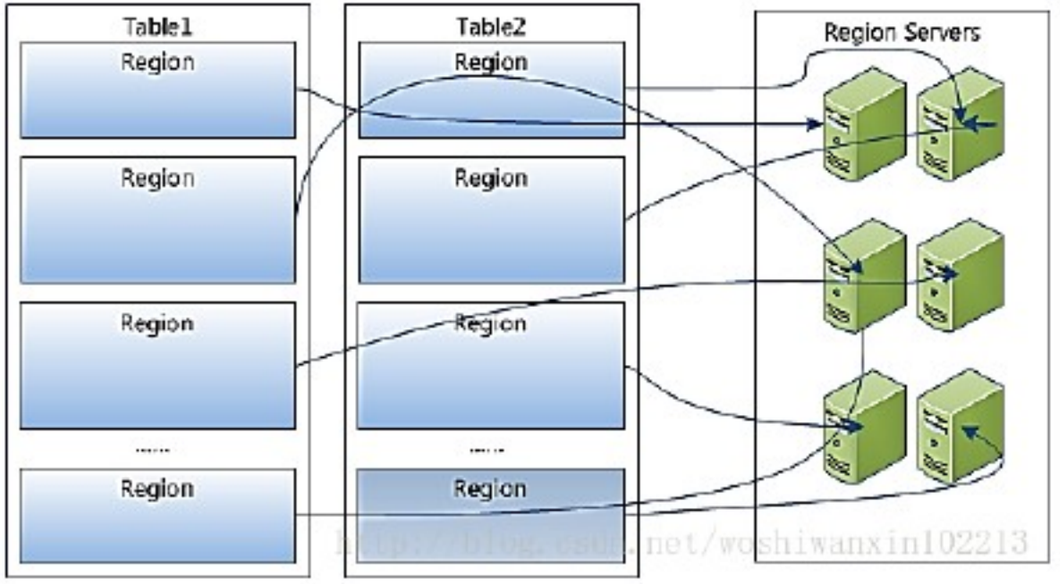
物理存储特点如下：

a) Region是HBase数据存储和管理的基本单位，一个表可以包含一个或多个Region，Table在行的方向上分割为多个Region。

b) Table中的所有行都按照RowKey的字典序排列。

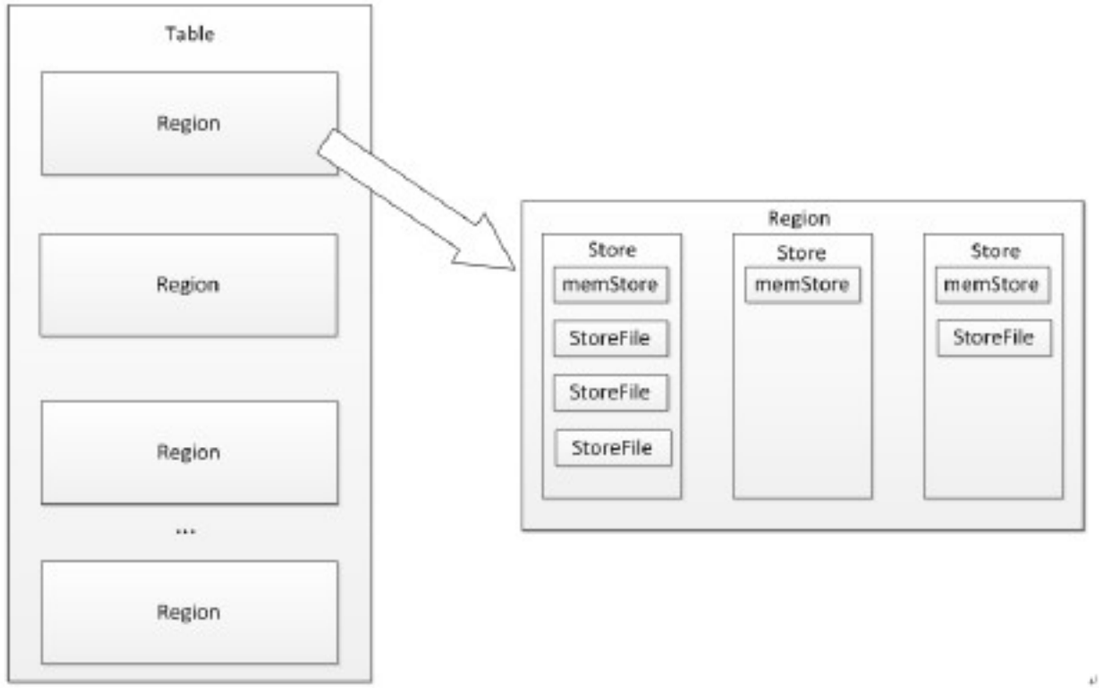
c) Region是按大小分割的，每个表开始只有一个Region，随着数据增多，Region会不断增大，当增大到一个阀值的时候，Region就会等分成两个新的Region，以此类推，会产生越来越多的Region。

d) 不同Region分布到不同的RegionServer上，如图



HBase数据模型——物理存储

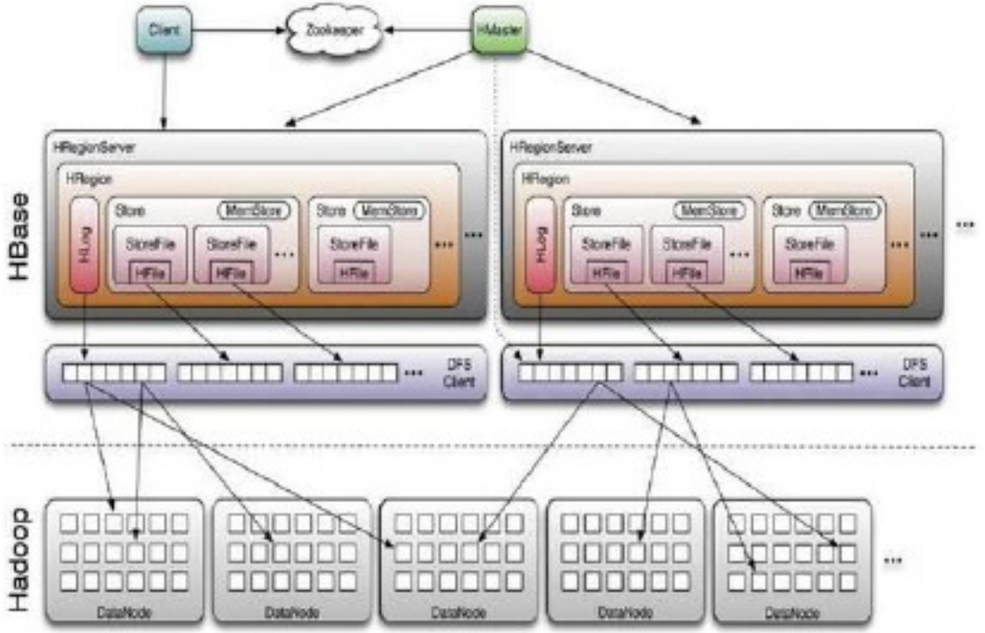
e) Region虽然是分布式存储的最小单元，但并不是存储的最小单元。Region由一个或者多个Store组成，每个Store保存一个Columns Family，每个Store又由一个MemStore和0至多个StoreFile组成，StoreFile则包含HFile。MemStore存储在内存中，StoreFile存储在HDFS上，如图



HBase组成架构

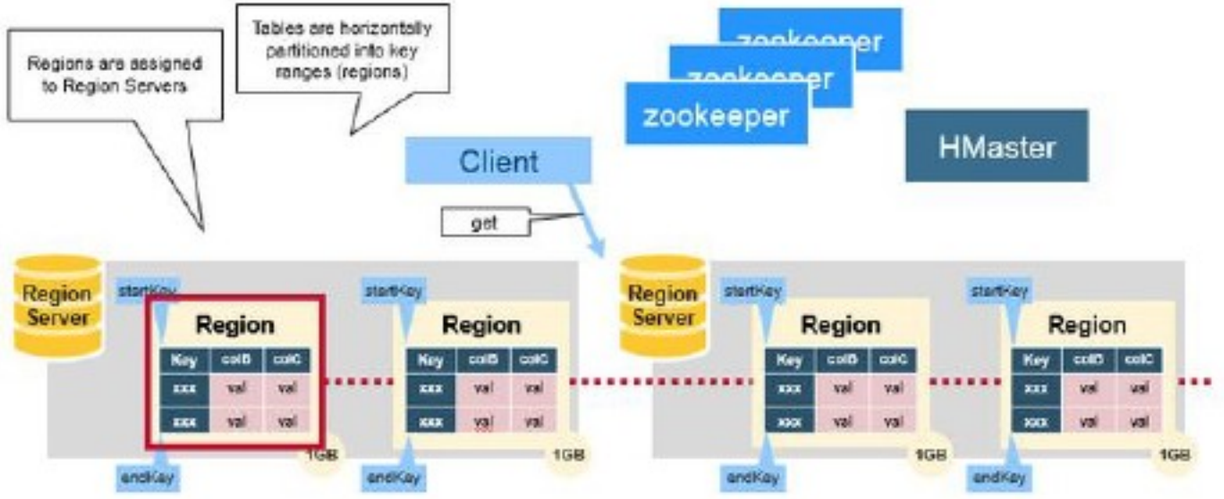
HBase采用Master/Slave架构搭建集群，它隶属于Hadoop生态系统，由三种类型的节点组成——HMaster节点、HRegionServer节点、ZooKeeper集群。而在底层，HBase将数据存储于HDFS中，因而也涉及到HDFS的NameNode节点、DataNode节点等。

HBase的总体架构如图：



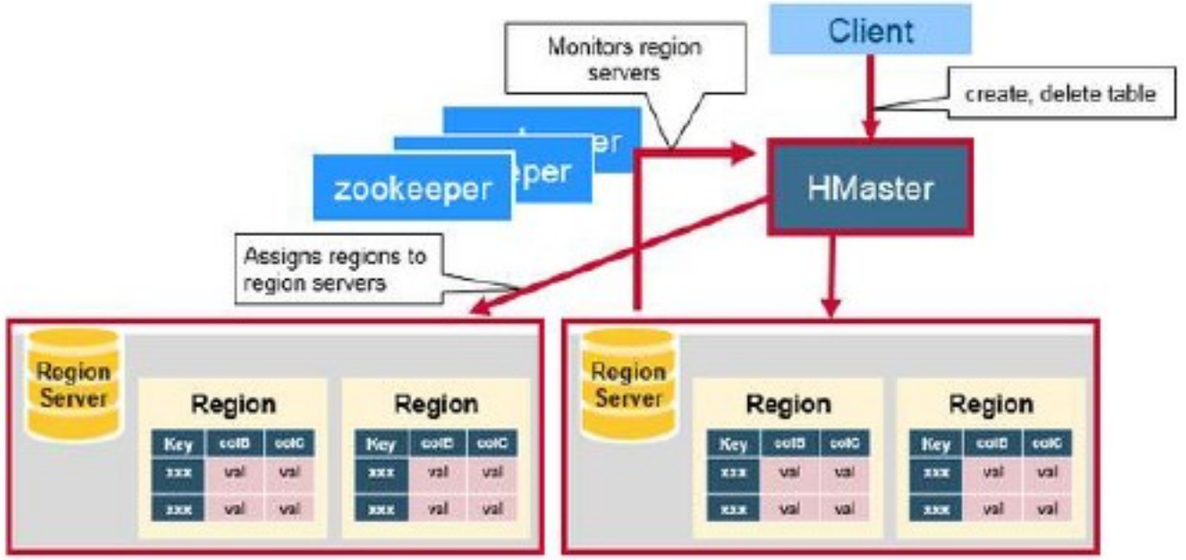
HBase组成架构——HRegion

HBase使用RowKey将表水平切割为多个HRegion，从HMaster的角度，每一个HRegion都纪录了RowKey的StartKey和EndKey。由于RowKey是可以排序的，因此Client可以通过HMaster节点快速定位每一个RowKey都在哪个HRegion中。HRegion由HMaster节点分配到相应的HRegionServer节点中，然后由HRegionServer节点负责HRegion的启动和管理以及和Client的通信，并实现数据的读操作（使用HDFS），如图：



HBase组成架构——HMaster

HMaster避免了单点故障问题，用户可以启动多个HMaster节点，并通过ZooKeeper的Master Election机制保证同时只有一个HMaster节点处于active状态，其他的HMaster节点则处于热备份状态。但是，一般情况下只会启动两个HMaster节点，因为非active状态的HMaster节点会定期和active状态下的HMaster节点通信，获取其最新状态来保证自身的实时更新，如启动的HMaster节点过多，反而会增加active状态下的HMaster节点的负担。
HMaster职责主要包括两大部分，如图：



HBase组成架构——ZooKeeper

ZooKeeper为HBase集群提供协调服务，它管理着HMaster节点和HRegionServer节点的状态（available/alive等），并且会在它们宕机时通知HMaster节点，从而实现HMaster节点之间的故障切换，或对宕机的HRegionServer节点中的HRegion进行修复（将它们分配给其他的HRegionServer节点），如图：