Hadoop 集群（第 11 期）

——HBase 简介及安装

1、HBase简介

HBase 是一个高可靠性、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 上搭建起**大规模结构化**存储集群。

HBase 的**目标**是存储并处理大型的数据，更具体来说是仅需使用普通的硬件配置，就能够处理由**成千上万**的**行**和**列**所组成的**大型数据**。

HBase 是 Google **Bigtable** 的开源实现，但是也有很多不同之处。比如：Google Bigtable 利用 **GFS** 作为其文件存储系统，HBase 利用 Hadoop **HDFS** 作为其文件存储系统；Google 运行 **MapReduce** 来处理 Bigtable 中的海量数据，HBase 同样利用 Hadoop **MapReduce** 来处理 HBase 中的海量数据；Google Bigtable 利用 **Chubby** 作为协同服务，HBase 利用 **Zookeeper** 作为对应。

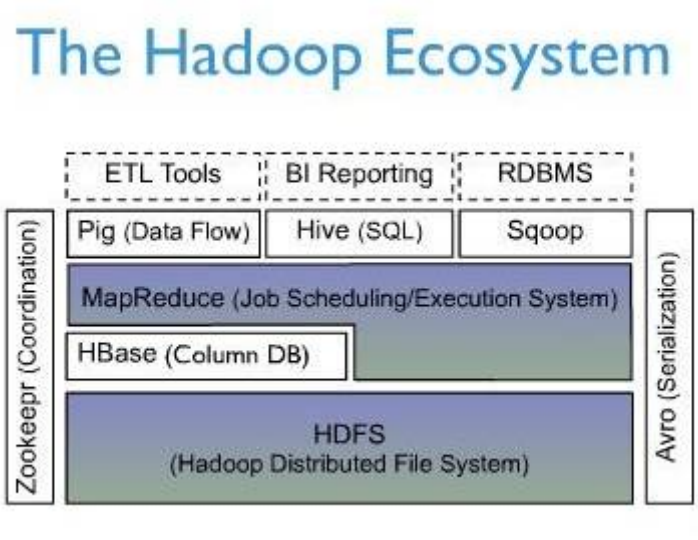


图 1-1 Hadoop 生态系统结构

上图描述了 **Hadoop EcoSystem** 中的**各层系统**，其中 **HBase** 位于**结构化存储层**，Hadoop HDFS 为 HBase 提供了高可靠性的底层存储支持，Hadoop MapReduce 为 HBase 提供了高性能的计算能力，Zookeeper 为 HBase 提供了稳定服务和 failover 机制。

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**处理变的非常简单。 **Sqoop** 则为 HBase 提供了方便的 **RDBMS** 数据**导入**功能，使得传统数据库数据向 HBase 中迁移变的非常方便。

另外，HBase 存储的是**松散型数据**。具体来说，HBase 存储的数据**介于映射**（key/value）和**关系型数据之间**。进一步讲，HBase 存储的数据可以理解为一种 key 和 value 的映射关系，但又不是简简单单的映射关系。除此之外它还有许多其他的特性。HBase 存储的数据从逻辑上来看就像一张很大的表，并且它的**数据列**可以根据需要**动态增加**。除此之外，每个 cell（由行和列所确定的位置）中的数据又可以具有多个版本（通过时间戳来区别）。

2、HBase体系结构

HBase 的**服务器体系结构**遵从简单的**主从服务器架构**，它由 **HRegin** 服务器（**HRegion Server**）群和 **HBase Master** 服务器（**HBase Master Server**）构成。HBase Master 服务器负责管理所有的 HRegion 服务器，而 HBase 中所有的服务器都是通过 **ZooKeeper** 来进行**协调**，并**处理** HBase 服务器**运行期间**可能**遇到**的**错误**。**HBase Master Server 本身**并**不存储** HBase 中的**任何数据**，HBase 逻辑上的表可能会被划分成多个 HRegion，然后存储到 HRegion Server 群中。HBase Master Server 中存储的是从数据到 HRegion Server 的映射。因此，HBase 体系结构如图 2-1 所示。

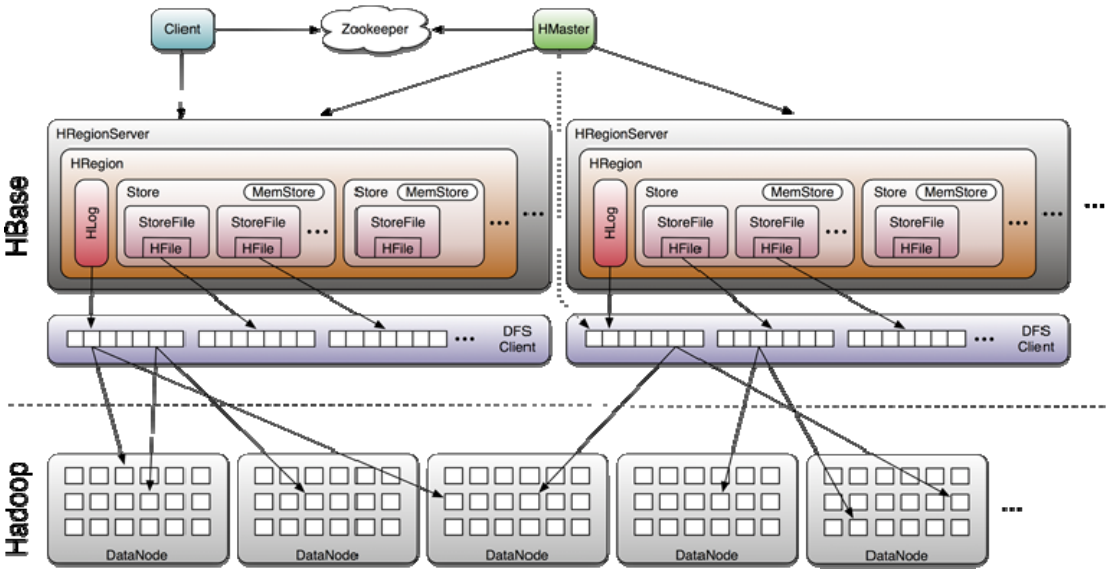


图 2-1 HBase 体系结构

2.1 Client

HBase Client 使用 HBase 的 **RPC 机制**与 **HMaster** 和 **HRegionServer** 进行**通信**，对于**管理类**操作，Client 与 HMaster 进行 RPC；对于**数据读写类**操作，Client 与 HRegionServer 进行 RPC。

2.2 Zookeeper

Zookeeper Quorum 中除了存储了**-ROOT-表的地址**和 **HMaster** 的地址，HRegionServer 也会把自己以 **Ephemeral** 方式注册到 Zookeeper 中，使得 HMaster 可以**随时感知**到各个 HRegionServer 的**健康状态**。此外，Zookeeper 也避免了 HMaster 的单点问题。

2.3 HMaster

每台 HRegion Server 都会 HMaster 通信，HMaster 的**主要任务**就是要告诉**每台 HRegion**

Server 它要维护那些 HRegion。

当一台新的 HRegion Server 登录到 HMaster 时，HMaster 会告诉它**等待分配数据**。而当一台 HRegion **死机**时，HMaster 会把它负责的 HRegion 标记为**未分配**，然后再把它们分配到其他 HRegion Server 中。

HMaster 没有**单点问题**（SPFO），HBase 中可以启动**多个 HMaster**，通过 **Zookeeper** 的 Master Election 机制保证**总有一个** Master 运行，HMaster 在功能上主要负责 Table 和 Region 的管理工作：

- 管理用户对 Table 的增、删、改、查操作；
- 管理 HRegion Server 的负载均衡，调整 Region 分布；
- 在 Region Split 后，负责新 Region 的分配；
- 在 HRegion Server 停机后，负责失效 HRegion Server 上的 Regions 迁移。

2.4 HRegion

当**表的大小超过设置值**的时候，**HBase** 会**自动**地将表**划分**为**不同的区域**，每个**区域**包含**所有行**的一个**子集**。**对用户来说**，每个表是一堆数据的集合，靠主键来区分。**从物理上来说**，一张表被拆分成了多块，每一块就是一个 HRegion。我们用**表名+开始/结束**主键，来区分每一个 HRegion，一个 HRegion 会保存一个表里面某段连续的数据，从开始主键到结束主键，一张**完整的**表格是**保存**在**多个** HRegion 上面。

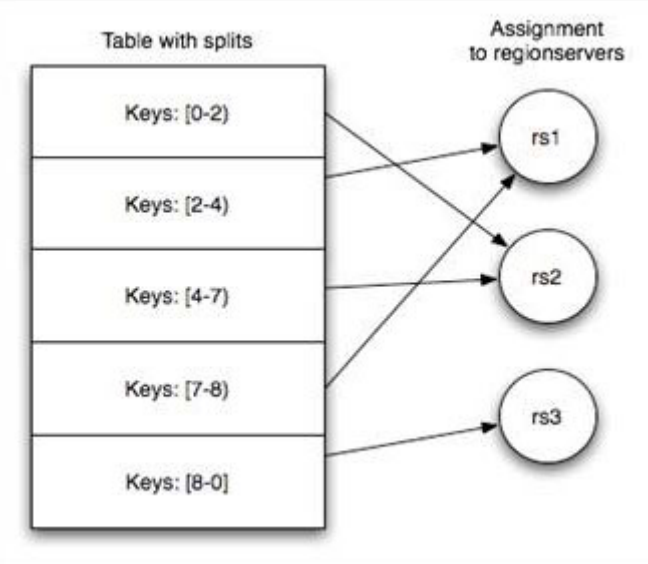


图 2.4-1 Table 分裂

上图表示当 Table 随着记录数不断增加而变大后，会逐渐分裂成多份 splits，成为 regions，一个 region 由[startkey,endkey]表示，不同的 region 会被 Master 分配给相应的 RegionServer 进行管理。

2.5 HRegion Server

所有的**数据库数据**一般是**保存**在 **Hadoop HDFS 分布式文件系统**上面，用户通过一系列 HRegion Server 获取这些数据，一台机器上面一般**只**运行一个 HRegion Server，且每一个**区**

段的 HRegion 也只会被一个 HRegion Server 维护。下面是 HRegion Server 数据存储关系图。

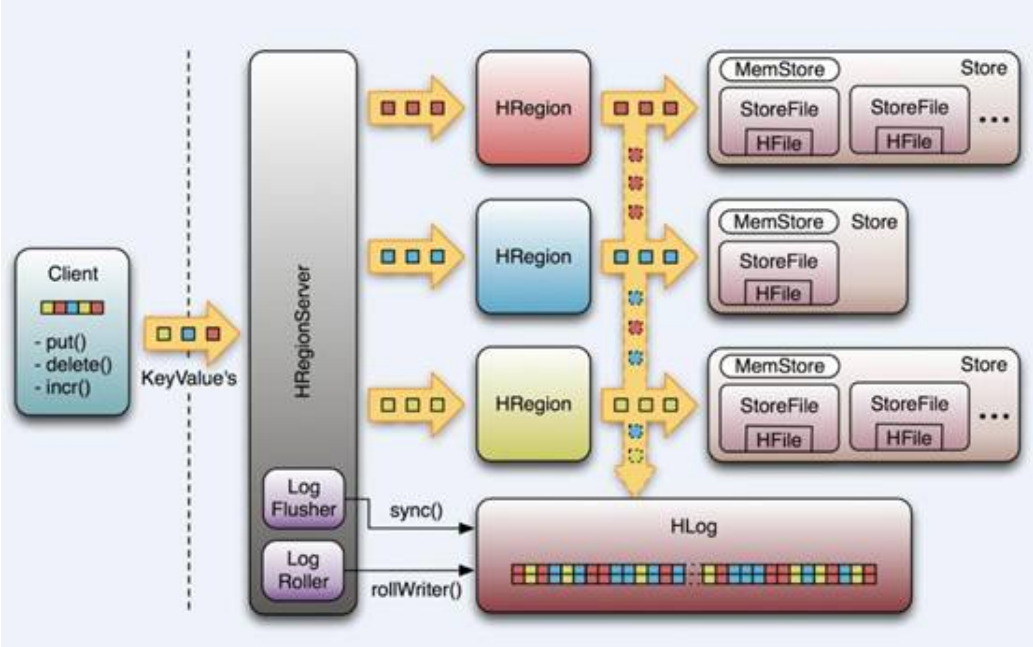


图 2.3-1 HRegion Server 数据存储关系

HRegion Server 主要负责响应用户 **I/O 请求**，向 HDFS 文件系统中**读写数据**，是 **HBase** 中**最核心**的模块。

HRegion Server **内部管理**了一系列 HRegion 对象，每个 **HRegion** 对应了 **Table** 中的一个 **Region**，**HRegion** 中由多个 **HStore** 组成。每个 **HStore** 对应了 Table 中的一个 **Column Family** 的存储，可以看出每个 **Column Family** 其实就是一个**集中**的**存储单元**，因此最好将具备共同 IO 特性的 column 放在一个 Column Family 中，这样最高效。

HStore 存储是 HBase 存储的**核心**了，其中由**两部分**组成，**一部分**是 **MemStore**，**一部分**是 **StoreFiles**。**MemStore** 是 Sorted Memory Buffer，用户写入的数据首先会放入 MemStore，当 MemStore 满了以后会 Flush 成一个 **StoreFile**（底层实现是 **HFile**），当 StoreFile 文件数量**增长**到一定**阈值**，会触发 **Compact** 合并操作，将多个 **StoreFiles** 合并成一个 **StoreFile**，合并过程中会进行**版本合并**和**数据删除**，因此可以看出 **HBase** 其实**只有增加数据**，所有的**更新**和**删除**操作**都是**在**后续**的 **compact** 过程中**进行**的，这使得用户的**写操作**只要进入内存中就可以立即返回，保证了 HBase I/O 的高性能。当 StoreFiles Compact 后，会逐步形成越来越大的 StoreFile，当**单个 StoreFile 大小**超过一定**阈值**后，会触发 **Split** 操作，同时把**当前 Region Split** 成 **2** 个 **Region**，父 Region 会下线，新 Split 出的 2 个孩子 Region 会被 HMaster 分配到相应的 HRegionServer 上，使得原先 1 个 Region 的压力得以分流到 2 个 Region 上。下图描述了 Compaction 和 Split 的过程。

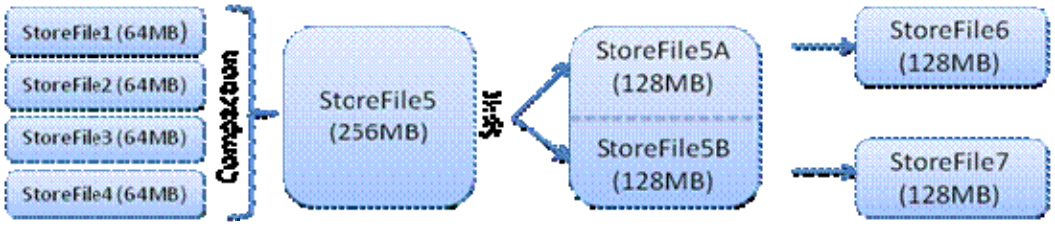


图 2.3-2 Compaction 和 Split 的过程

在理解了上述 HStore 的基本原理后，还必须了解一下 **HLog** 的功能，因为上述的 HStore 在系统正常工作的前提下是没有问题的，但是在**分布式系统环境**中，无法避免系统出错或者宕机，因此一旦 HRegion Server 意外退出，MemStore 中的内存数据将会丢失，这就需要引入 HLog 了。每个 HRegion Server 中都有一个 HLog 对象，HLog 是一个实现 **Write Ahead Log** 的类，在**每次**用户**操作写入 MemStore** 的同时，**也会写一份数据**到 **HLog 文件**中（HLog 文件格式见后续），**HLog 文件定期**会**滚动出新**的，并**删除**旧的文件（已持久化到 StoreFile 中的数据）。当 **HRegion Server 意外终止**后，**HMaster** 会通过 **Zookeeper 感知到**，HMaster **首先**会处理**遗留**的 **HLog 文件**，将其中**不同 Region** 的 Log 数据进行**拆分**，分别放到相应 region 的目录下，然后再将**失效**的 region **重新分配**，领取到这些 region 的 HRegion Server 在 Load Region 的过程中，会发现有历史 HLog 需要处理，因此会 **Replay HLog** 中的数据到 MemStore 中，然后 flush 到 StoreFiles，完成数据恢复。

2.6 HBase存储格式

HBase 中的所有数据文件都存储在 Hadoop HDFS 文件系统上，主要包括上述提出的两种文件类型：

- **HFile**， HBase 中 KeyValue 数据的存储格式，HFile 是 Hadoop 的二进制格式文件，实际上 StoreFile 就是对 HFile 做了轻量级包装，即 StoreFile 底层就是 HFile。

- **HLog File**，HBase 中 WAL（Write Ahead Log） 的存储格式，物理上是 Hadoop 的 Sequence File。

1）HFile 详细描述

下图是 HFile 的存储格式：

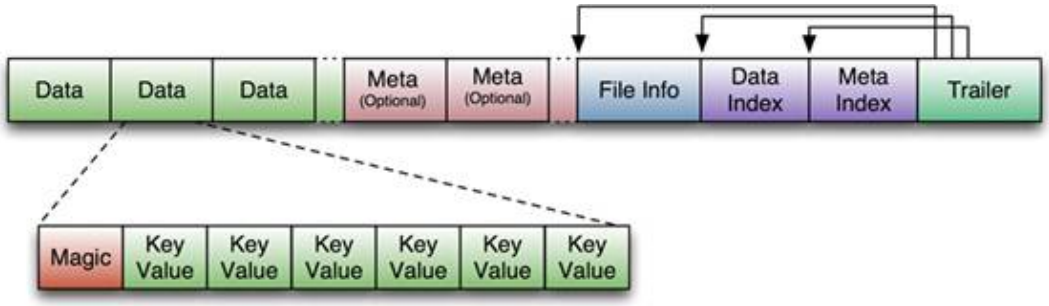


图 2.6-1 HFile 存储格式

首先 **HFile** 文件是**不定长**的，**长度固定**的**只有**其中的**两块**：**Trailer** 和 **File Info**。正如图中所示的，Trailer 中有指针指向其他数据块的起始点。File Info 中记录了文件的一些 Meta 信息，**例如**：AVG_KEY_LEN，AVG_VALUE_LEN，LAST_KEY，COMPARATOR，MAX_SEQ_ID_KEY 等。**Data Index** 和 **Meta Index** 块记录了每个 **Data** 块和 **Meta** 块的**起始点**。

Data Block 是 HBase I/O 的**基本单元**，为了提高效率，HRegion Server 中有基于 **LRU** 的 **Block Cache 机制**。每个 Data 块的大小可以在创建一个 Table 的时候通过参数指定，**大**号的 Block 有利于顺序 **Scan**，**小**号 Block 利于**随机查询**。 每个 Data 块除了开头的 Magic 以外就是一个个 KeyValue 对拼接而成，Magic 内容就是一些随机数字，目的是防止数据损坏。后面会详细介绍每个 KeyValue 对的内部构造。

HFile 里面的每个 **KeyValue** 对就是一个**简单**的 **byte 数组**。但是这个 byte 数组里面包含

了很多项，并且有固定的结构。我们来看看里面的具体结构：

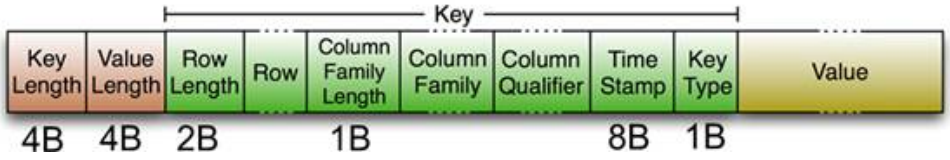


图 2.6-2 KeyValue 具体结构

开始是两个固定长度的数值，分别表示 **Key** 的长度和 **Value** 的长度。紧接着是 Key，开始是固定长度的数值，表示 RowKey 的长度，紧接着是 RowKey，然后是固定长度的数值，表示 Family 的长度，然后是 Family，接着是 Qualifier，然后是两个固定长度的数值，表示 Time Stamp 和 Key Type（Put/Delete）。Value 部分没有这么复杂的结构，就是纯粹的二进制数据了。

2）HLogFile 详细描述

其实 HLog 文件就是一个普通的 Hadoop Sequence File，Sequence File 的 Key 是 HLogKey 对象，HLogKey 中记录了写入数据的归属信息，除了 table 和 region 名字外，同时还包括 sequence number 和 timestamp，timestamp 是“写入时间”，sequence number 的起始值为 0，或者是最近一次存入文件系统中 sequence number。

HLog Sequece File 的 Value 是 HBase 的 KeyValue 对象，即对应 HFile 中的 KeyValue，可参见上文描述。下图中示意了 HLog 文件的结构：

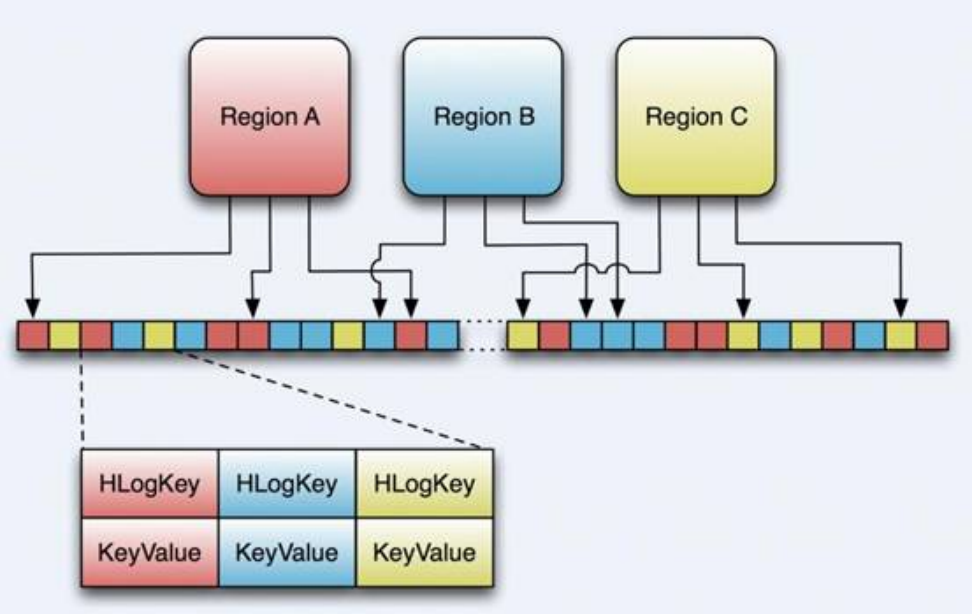


图 2.6-3 HLogFile 结构

2.7 ROOT表和META表

用户表的 Regions 元数据被存储在.META.表中，随着 Region 的增多，.META.表中的数据也会增大，并分裂成多个 Regions。为了定位.META.表中各个 Regions 的位置，把.META.表中所有 Regions 的元数据保存在-ROOT-表中，最后由 ZooKeeper 记录-ROOT-表的位置信息。所有客户端访问用户数据前，需要首先访问 ZooKeeper 获得-ROOT-的位置，然后访问-ROOT-表获得.META.表的位置，最后根据.META.表中的信息确定用户数据存放的位置，

如图 2.7-1 所示：

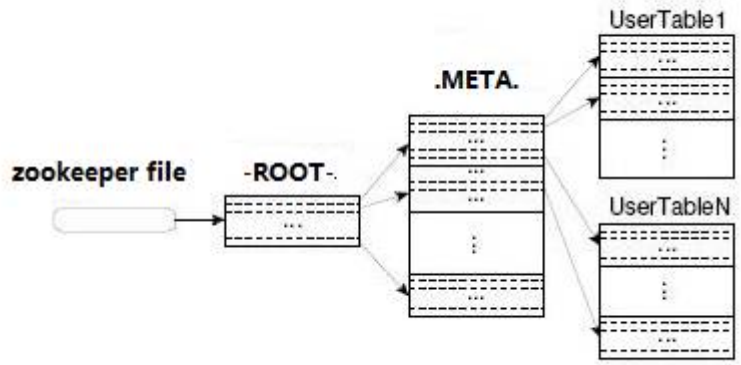


图 2.7-1 Region 定位示意图

-ROOT-表**永远不会**被**分割**，它**只有**一个 Region，这样可以保证最多需要**三次**跳转就可以**定位任意**一个 Region。为了加快访问速度，**.META.**表的 **Regions** 全部保存在**内存**中，如果**.META.**表中的每一行在内存中大约占 1KB，且每个 Region 限制为 128MB，那么上图所示的三层结构可以保存的 Regions 数目为：（128MB/1KB）*（128/1KB）=2^{34} 个。客户端会将查询过的位置信息缓存起来，且缓存不会主动失效。如果客户端根据缓存信息还访问不到数据，则询问只有相关**.META.**表的 Region 服务器，试图获取数据的位置，如果还是**失败**，则询问**-ROOT-**表相关的**.META.**表在哪里。最后，如果前面的信息**全部失效**，则通过 **ZooKeeper 重新定位** Region 的信息。所以如果**客户端**上的**缓存全部**是**失效**，则需要进行 **6** 次**网络来回**，才能定位到正确的 Region。

2.8 MapReduce On HBase

在 HBase 系统上运行批处理运算，最方便和实用的模型依然是 MapReduce，如下图：

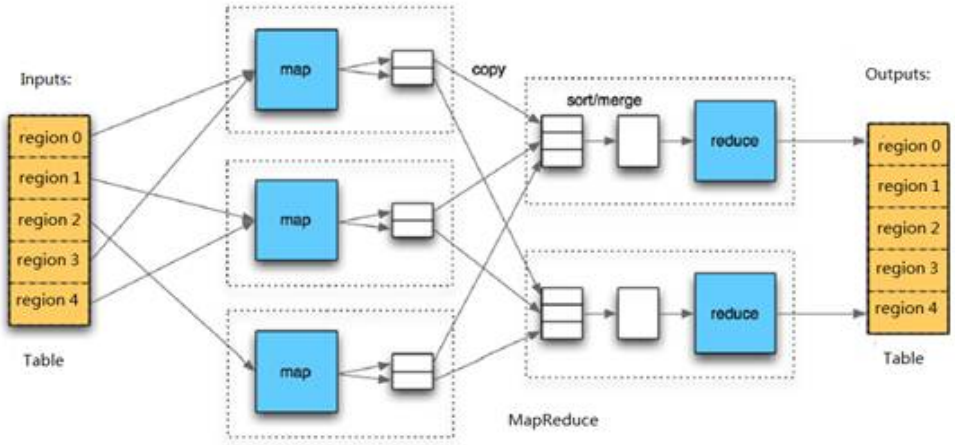


图 2.8-1 MapReduce On HBase

HBase Table 和 Region 的关系，比较类似 HDFS File 和 Block 的关系，HBase 提供了配套的 TableInputFormat 和 TableOutputFormat API，可以方便的将 HBase Table 作为 Hadoop MapReduce 的 Source 和 Sink，对于 MapReduce Job 应用开发人员来说，基本不需要关注 HBase

系统自身的细节。

3、HBase数据模型

HBase 是一个类似于 BigTable 的分布式数据库，它是一个稀疏的长期存储的（存在硬盘上）、多维度的、排序的映射表。这张表的索引是行关键字、列关键字和时间戳。HBase 的数据都是字符串，没有类型。

用户在表格中存储数据，每行都有一个可排序的主键和任意多的列。由于是稀松存储，所以同一张表里面的每行数据都可以由截然不同的列。

列名字的格式是“<family>:<qualifier>”（<列族>:<限定符>），都是又字符串组成的。每一张表有一个列族（family）集合，这个集合是固定不变的，只能通过改变表结构来改变。但是限定符（qualifier）的值相对于每一行来说都是可以改变的。

HBase 把同一个列族里面的数据存储在同一个目录底下，并且 HBase 的写操作时锁行的，每一行来说都是一个原子元素，都可以加锁。

HBase 所有数据库的更新都有一个时间戳标记，每个更新都是一个新的版本，HBase 会保留一定数量的版本，这个值是可以设定的。客户端可以选择获取距离某个时间点最近的版本单元的值，或者一次获取所有版本单元的值。

3.1 逻辑模型

我们可以将一个表想象成一个大的映射关系，通过行健、行健+时间戳或行健+列（列族：列修饰符），就可以定位特定数据。由于HBase是稀疏存储数据的，所以某些列可以空白的。表 3.1-1 给出了www.cnn.com网站的数据存放逻辑视图，表中仅有一行数据，行的唯一标识为“com.cnn.www”，对这行数据的每一次逻辑修改都有一个时间戳关联对应。表中共有四列：contents:html、anchor:cnnsi.com、anchor:my.look.ca、mime:type，每一行以前缀的方式给出其所属的列族。

表 3.1-1 数据存储逻辑视图

行 健	时间戳	列族：contents	列族：anchor	列族：mime
“com.cnn.www”	t9		achor:cnnsi.com=”CNN”	
	t8		achor:my.lock.ca=”CNN.com”	
	t6	contents:html=”<html>…”		mime:type=”text/html”
	t5	contents:html=”<html>…”		
	t3	contents:html=”<html>…”		

行健是数据行在表中的唯一标识，并作为检索记录的主键。在 HBase 中访问表中的行只有三种方式：通过当个行健访问；给定行健的范围访问；全表扫描。行健可以任意字符串（最大长度 64KB）并按照字典序进行存储。对于那些经常一起读取的行，需要对 key 值精心设计，以便它们能放在一起存储。

3.2 概念模型

HBase 是按照列存储的稀疏行/列矩阵，物理模型实际上就是把概念模型中的一行进行

切割，并按照列族存储，这点在进行数据设计和程序开发的时候必须牢记。

上面的逻辑视图在物理存储的时候应该表现成下面的样子，如表 3.2-1 所示。

表 3.2-1 物理上的存储方式

行 健	时 间 戳	列族：contents
“com.cnn.www”	t6	contents:html=”<html>…”
	t5	contents:html=”<html>…”
	t3	contents:html=”<html>…”
行 健	**时 间 戳**	**列族：anchor**
“com.cnn.www”	t9	achor:cnnsi.com=”CNN”
	t8	achor:my.lock.ca=”CNN.com”
行 健	**时 间 戳**	**列族：mime**
“com.cnn.www”	t6	mime:type=”text/html”

从表中可以看出表中的空值是不被存储的，所以查询时间戳为 t8 的“contents:html”将返回 null，同样查询时间戳为 t9，“anchor:my.lock.ca”的项也返回 null。如果没有指明时间戳，那么应该返回指定列的最新数据值，并且最新的值在表格里也是最先找到的，因为它们是按照时间排序的。所以，如果查询“contents:”而不指明时间戳，将返回 t6 时刻的数据；查询“anchor:”的“my.look.ca”而不指明时间戳，将返回 t8 时刻的数据。这种存储结构还有一个优势，可以随时向表中的任何一个列族添加新列，而不需要是事先说明。

4、HBase分布式安装

4.1 先决条件

HBase 有三种运行模式，其中单机模式的配置非常简单，几乎不用对安装文件做任何修改就可以使用，所以我们这里不再介绍 HBase 的单机模式的安装。从前面的讲解中，我们知道如果要运行分布式模式，Hadoop 是必不可少的。另外在对 HBase 的某些文件进行配置之前，还需要具备以下先决条件：

- **Java**：需要是 Java1.6.x 以上的版本。
- **Hadoop**：由于 HBase 架构基于其他文件存储系统之上，因此在分布式模式下安装 Hadoop 是必须的，但是，如果运行在单机模式下，此条件则可以省略。
- **SSH**：需要注意的是，SSH 是必须安装的，并且要保证用户可以 SSH 到系统的其他节点（包括本地节点）。因为，我们需要使用 Hadoop 来管理远程的 Hadoop 和 HBase 守护进程。

备注：在安装 Hadoop 的时候，要注意 HBase 的版本。也就是说，需要注意 Hadoop 和 HBase 之间的版本关系，如果不匹配，很可能会影响 HBase 系统的稳定性，在 HBase 的 lib 目录下可以看到对应的 Hadoop 的 jar 文件。默认情况下，HBase 的 lib 文件下对应的 Hadoop 版本相对稳定。如果用户想要使用其他的 Hadoop 版本，那么需要将 Hadoop 系统安装目录下的“hadoop-core-*.*.*.jar”文件和“hadoop-test-*.*.*.jar”文件拷贝到 HBase 的 lib 文件夹下，以替换其他版本的 Hadoop 文件。

4.2 集群环境

下面为当前 Hadoop 集群的环境情况，务必确保在配置 Hadoop 时已配置好 hosts 文件的内容，否则在 HBase 的配置文件中使用主机名代替 IP 地址时会出现错误。

- **Java** 版本：jdk-6u31-linux-i586
- **Linux** 版本：CentOS-6.0-i386
- **HBase** 版本：hbase-0.92.0
- **Hadoop** 版本：hadoop-1.0.0
- **Hadoop** 集群：

表 4.2-1 Hadoop 集群信息

机器名称	IP 地址	守护进程
Master.Hadoop	192.168.1.2	NameNode、SecondaryNameNode、JobTracker
Salve1.Hadoop	192.168.1.3	DataNode、TaskTracker
Salve2.Hadoop	192.168.1.4	DataNode、TaskTracker
Salve3.Hadoop	192.168.1.5	DataNode、TaskTracker

下面所示为即将安装的 **HBase** 集群的运行情况。

表 4.2-2 HBase 集群信息

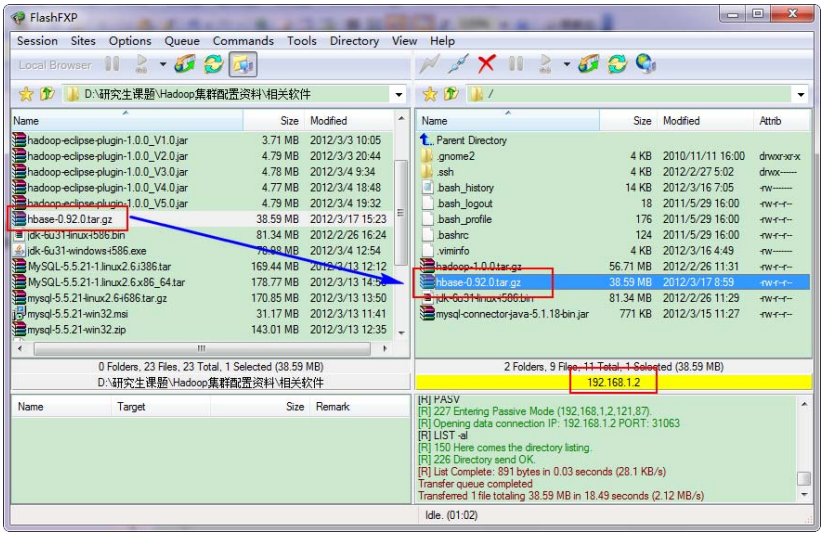
机器名称	IP 地址	守护进程
Master.Hadoop	192.168.1.2	HMaster
Salve1.Hadoop	192.168.1.3	HQuorumPeer、HRegionServer
Salve2.Hadoop	192.168.1.4	HQuorumPeer、HRegionServer
Salve3.Hadoop	192.168.1.5	HQuorumPeer、HRegionServer

备注：我们使用的 HBase 和 Hadoop 版本经过查看是相吻合的，不需要进行任何修改。

4.3 安装HBase

第一步：FTP 上传 HBase 安装文件

用“FlashFXP”把 HBase 安装文件上传到“Master.Hadoop”机器上。



用 SecureCRT 进行查看结果如下：

```
[hadoop@Master ~]$ pwd
/home/hadoop
[hadoop@Master ~]$ ll
总用量 181660
-rw-r--r--. 1 hadoop hadoop 59468784  2月 27 03:31 hadoop-1.0.0.tar.gz
-rw-r--r--. 1 hadoop hadoop 40461930  3月 18 00:59 hbase-0.92.0.tar.gz
-rw-r--r--. 1 hadoop hadoop 85292206  2月 27 03:29 jdk-6u31-linux-i586.bin
-rw-r--r--. 1 hadoop hadoop   789885  3月 16 03:27 mysql-connector-java-5.1.18-bin.jar
[hadoop@Master ~]$
```

当前路径

第二步：安装 HBase 数据库

首先切换到“**root**”用户下，我们这次之前安装 JDK 和 Hadoop 一样，我们都安装在“/usr”目录下面。

```
[hadoop@Master ~]$ su -
密码：
[root@Master ~]#
```

然后把“**hbase-0.92.0.tar.gz**”复制到“**/usr**”下面。

```
cp    /home/hadoop/hbase-0.92.0.tar.gz    /usr
```

```
[root@Master ~]# ll /home/hadoop
总用量 181660
-rw-r--r--. 1 hadoop hadoop 59468784  2月 27 03:31 hadoop-1.0.0.tar.gz
-rw-r--r--. 1 hadoop hadoop 40461930  3月 18 00:59 hbase-0.92.0.tar.gz
-rw-r--r--. 1 hadoop hadoop 85292206  2月 27 03:29 jdk-6u31-linux-i586.bin
-rw-r--r--. 1 hadoop hadoop   789885  3月 16 03:27 mysql-connector-java-5.1.18-bin.jar
[root@Master ~]# cp /home/hadoop/hbase-0.92.0.tar.gz /usr
[root@Master ~]# ll /usr
总用量 39620
dr-xr-xr-x.  2 root   root      24576  2月 24 05:34 bin
drwxr-xr-x.  2 root   root       4096 11月 11 2010 etc
drwxr-xr-x.  2 root   root       4096 11月 11 2010 games
drwxr-xr-x. 16 hadoop hadoop     4096  2月 29 20:50 hadoop
-rw-r--r--.  1 root   root   40461930  3月 18 01:16 hbase-0.92.0.tar.gz
drwxr-xr-x. 33 root   root       4096  2月 24 04:40 include
drwxr-xr-x.  3 root   root       4096  2月 28 04:06 java
dr-xr-xr-x. 53 root   root      32768  2月 24 05:34 lib
drwxr-xr-x. 14 root   root       4096  2月 24 05:34 libexec
drwxr-xr-x. 11 root   root       4096  2月 24 04:37 local
dr-xr-xr-x.  2 root   root       4096  2月 24 05:34 sbin
drwxr-xr-x. 95 root   root       4096  2月 24 04:41 share
drwxr-xr-x.  4 root   root       4096  2月 24 04:37 src
lrwxrwxrwx.  1 root   root         10  2月 24 04:37 tmp -> ../var/tmp
[root@Master ~]#
```

接着进入“**/usr**”目录下，用下面命令把“**hbase-0.92.0.tar.gz**”进行解压，并将其命名为“**hbase**”，把该文件夹的权限分配给普通用户 **hadoop**，然后删除“**hbase-0.92.0.tar.gz**”安装包。

```
cd    /usr                                  #进入“/usr”目录
tar   –zxvf    hbase-0.92.0.tar.gz          #解压“hbase-0.92.0.tar.gz”安装包
mv    hbase-0.92.0    hbase                 #将“hbase-0.92.0”文件夹重命名“hbase”
chown   –R    hadoop:hadoop    hbase        #将文件夹“hbase”权限分配给 hadoop 用户
```

rm　–rf　hbase-0.92.0.tar.gz　　#删除“**hbase-0.92.0.tar.gz**”安装包

```
[root@Master ~]# cd /usr
[root@Master usr]# ll
总用量 39620
dr-xr-xr-x.  2 root   root      24576  2月 24 05:34 bin
drwxr-xr-x.  2 root   root       4096 11月 11 2010 etc
drwxr-xr-x.  2 root   root       4096 11月 11 2010 games
drwxr-xr-x. 16 hadoop hadoop     4096  2月 29 20:50 hadoop
-rw-r--r--.  1 root   root   40461930  3月 18 01:16 hbase-0.92.0.tar.gz
drwxr-xr-x. 33 root   root       4096  2月 24 04:40 include
drwxr-xr-x.  3 root   root       4096  2月 28 04:06 java
dr-xr-xr-x. 53 root   root      32768  2月 24 05:34 lib
drwxr-xr-x. 14 root   root       4096  2月 24 05:34 libexec
drwxr-xr-x. 11 root   root       4096  2月 24 04:37 local
dr-xr-xr-x.  2 root   root       4096  2月 24 05:34 sbin
drwxr-xr-x. 95 root   root       4096  2月 24 04:41 share
drwxr-xr-x.  4 root   root       4096  2月 24 04:37 src
lrwxrwxrwx.  1 root   root         10  2月 24 04:37 tmp -> ../var/tmp
[root@Master usr]# tar -zxvf hbase-0.92.0.tar.gz
```

解压之后，然后重命名。

```
[root@Master usr]# ll
总用量 39624
dr-xr-xr-x.  2 root   root      24576  2月 24 05:34 bin
drwxr-xr-x.  2 root   root       4096 11月 11 2010 etc
drwxr-xr-x.  2 root   root       4096 11月 11 2010 games
drwxr-xr-x. 16 hadoop hadoop     4096  2月 29 20:50 hadoop
drwxr-xr-x.  8   1002   1002     4096  1月 16 23:03 hbase-0.92.0
-rw-r--r--.  1 root   root   40461930  3月 18 01:16 hbase-0.92.0.tar.gz
drwxr-xr-x. 33 root   root       4096  2月 24 04:40 include
drwxr-xr-x.  3 root   root       4096  2月 28 04:06 java
dr-xr-xr-x. 53 root   root      32768  2月 24 05:34 lib
drwxr-xr-x. 14 root   root       4096  2月 24 05:34 libexec
drwxr-xr-x. 11 root   root       4096  2月 24 04:37 local
dr-xr-xr-x.  2 root   root       4096  2月 24 05:34 sbin
drwxr-xr-x. 95 root   root       4096  2月 24 04:41 share
drwxr-xr-x.  4 root   root       4096  2月 24 04:37 src
lrwxrwxrwx.  1 root   root         10  2月 24 04:37 tmp -> ../var/tmp
[root@Master usr]# mv hbase-0.92.0 hbase
[root@Master usr]# ll
总用量 39624
dr-xr-xr-x.  2 root   root      24576  2月 24 05:34 bin
drwxr-xr-x.  2 root   root       4096 11月 11 2010 etc
drwxr-xr-x.  2 root   root       4096 11月 11 2010 games
drwxr-xr-x. 16 hadoop hadoop     4096  2月 29 20:50 hadoop
drwxr-xr-x.  8   1002   1002     4096  1月 16 23:03 hbase
-rw-r--r--.  1 root   root   40461930  3月 18 01:16 hbase-0.92.0.tar.gz
drwxr-xr-x. 33 root   root       4096  2月 24 04:40 include
drwxr-xr-x.  3 root   root       4096  2月 28 04:06 java
dr-xr-xr-x. 53 root   root      32768  2月 24 05:34 lib
drwxr-xr-x. 14 root   root       4096  2月 24 05:34 libexec
drwxr-xr-x. 11 root   root       4096  2月 24 04:37 local
dr-xr-xr-x.  2 root   root       4096  2月 24 05:34 sbin
drwxr-xr-x. 95 root   root       4096  2月 24 04:41 share
```

把“**/usr/hbase**”的权限分配给 **hadoop** 用户。（**非常重要**）

```
[root@Master usr]# chown -R hadoop:hadoop hbase
[root@Master usr]# ll
总用量 39624
dr-xr-xr-x.  2 root   root      24576  2月 24 05:34 bin
drwxr-xr-x.  2 root   root       4096 11月 11 2010 etc
drwxr-xr-x.  2 root   root       4096 11月 11 2010 games
drwxr-xr-x. 16 hadoop hadoop     4096  2月 29 20:50 hadoop
drwxr-xr-x.  8 hadoop hadoop     4096  1月 16 23:03 hbase
-rw-r--r--.  1 root   root   40461930  3月 18 01:16 hbase-0.92.0.tar.gz
drwxr-xr-x. 33 root   root       4096  2月 24 04:40 include
drwxr-xr-x.  3 root   root       4096  2月 28 04:06 java
dr-xr-xr-x. 53 root   root      32768  2月 24 05:34 lib
drwxr-xr-x. 14 root   root       4096  2月 24 05:34 libexec
drwxr-xr-x. 11 root   root       4096  2月 24 04:37 local
dr-xr-xr-x.  2 root   root       4096  2月 24 05:34 sbin
drwxr-xr-x. 95 root   root       4096  2月 24 04:41 share
drwxr-xr-x.  4 root   root       4096  2月 24 04:37 src
lrwxrwxrwx.  1 root   root         10  2月 24 04:37 tmp -> ../var/tmp
[root@Master usr]#
```

删除“**hbase-0.92.0.tar.gz**”安装包。

```
[root@Master usr]# rm -rf hbase-0.92.0.tar.gz
[root@Master usr]#
```

第三步：编辑 HBase 配置文件

1）配置 hbase-env.sh

该文件“**hbase-env.sh**”位于“**/usr/hbase/conf**”目录下。

```
[root@Master conf]# pwd
/usr/hbase/conf
[root@Master conf]# ll
总用量 24
-rw-r--r--. 1 hadoop hadoop 2335  1月 16 23:03 hadoop-metrics.properties
-rw-r--r--. 1 hadoop hadoop 3502  1月 16 23:03 hbase-env.sh
-rw-r--r--. 1 hadoop hadoop 2250  1月 16 23:03 hbase-policy.xml
-rw-r--r--. 1 hadoop hadoop  983  1月 16 23:03 hbase-site.xml
-rw-r--r--. 1 hadoop hadoop 2070  1月 16 23:03 log4j.properties
-rw-r--r--. 1 hadoop hadoop   10  1月 16 23:03 regionservers
[root@Master conf]# vim hbase-env.sh
```

在文件的**尾部**添加下面的内容

```
# set hbase environment
export JAVA_HOME=/usr/java/jdk1.6.0_31          #Java 安装位置
export HBASE_MANAGES_ZK=true                    #由 HBase 负责启动和关闭 ZooKeeper
export HBASE_CLASSPATH=/usr/hadoop/conf         #HBase 类路径
```

```
# set hbase environment
export JAVA_HOME=/usr/java/jdk1.6.0_31
export HBASE_MANAGES_ZK=true
export HBASE_CLASSPATH=/usr/hadoop/conf

~
~
"hbase-env.sh" 87L, 3636C 已写入
[hadoop@Master conf]$
```

其实上面的变量在“**hbase-env.sh**”就已经存在了，只是注释掉了，但是我们不在原来的基础改，主要是把添加的变量集中管理。

2）配置 hbase-site.xml

该文件“**hbase-site.xml**”位于“**/usr/hbase/conf**”目录下。

```
<configuration>
    <property>
        <name>hbase.master</name>
        <value>Master.Hadoop:60000</value>
    </property>
    <property>
        <name>hbase.master.maxclockskew</name>
        <value>180000</value>
    </property>
    <property>
        <name>hbase.rootdir</name>
        <value>hdfs://Master.Hadoop:9000/hbase</value>
    </property>
    <property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
    </property>
    <property>
          <name>hbase.zookeeper.quorum</name>
          <value>Slave1.Hadoop,Slave2.Hadoop,Slave3.Hadoop</value>
    </property>
    <property>
          <name>hbase.zookeeper.property.dataDir</name>
          <value>/home/ ${user.name} /tmp/zookeeper</value>
    </property>
</configuration>
```

上面的这张配置单中用**蓝色颜色**标识的“hbase.rootdir”和“hbase.cluster.distributed”两个参数对于 HBase 来说是必需的。通过“hbase.rootdir”来指定 HBase 的存储目录，它的值

必须与“core-site.xml”配置文件中“fs.default.name”保持一致，如果你 Hadoop 的 hdfs 使用了其它端口，请在这里也修改。通过“hbase.cluster.distributed”来说明其运行模式：true 为全分布式模式；false 为单机模式或伪分布模式。将“hbase.zookeeper.quorum”设置为所有 ZooKeeper 节点的主机名，默认为 localhost，它的值必须是奇数。属性“hbase.zookeeper.property.dataDir”表示 ZooKeeper 的目录，默认为“/tmp”，系统重启后会被清空。参数“hbase.master.maxclockskew”是用来防止“hbase 结点之间时间不一致造成 regionserver 启动失败”，默认的值为“30000”，现改为“180000”。

```
<configuration>
        <property>
                <name>hbase.master</name>
                <value>Master.Hadoop:6000</value>
        </property>
        <proterty>
                <name>hbase.master.maxclockskew</name>
                <value>180000</value>
        </proterty>
        <property>
                <name>hbase.rootdir</name>
                <value>hdfs://Master.Hadoop:9000/hbase</value>
        </property>
        <property>
                <name>hbase.cluster.distributed</name>
                <value>true</value>
        </property>
        <property>
                <name>hbase.zookeeper.quorum</name>
                <value>Slave1.Hadoop,Slave2.Hadoop,Slave3.Hadoop</value>
        </property>
        <property>
                <name>hbase.zookeeper.property.dataDir</name>
                <value>/home/${user.name}/tmp/zookeeper</value>
        </property>
</configuration>
"hbase-site.xml" 49L, 1598C 已写入
[hadoop@Slave3 conf]$
```

备注：HBase 对“hbase-site.xml”中某些选项并不识别机器 IP，为了保险起见都设成了机器主机名，前提是在 Hadoop 集群中的机器“/etc/hosts”添加上[IP 和 HostName]，这个我们在安装 Hadoop 时已经设置了。如果忘了的记得要设置，不然无法解析主机名。

3）配置 regionservers

该文件“regionservers”位于“/usr/hbase/conf”目录下。

“regionservers”文件列出了所有运行 HBase 的机器（即 HRegionServer）。此文件的配置和 Hadoop 的 Slaves 文件十分类似，每一行指定一台机器。当 HBase 启动的时候，会将此文件中列出的所有机器启动；同样，当 HBase 关闭的时候，也会同时关闭它们。

在该“regionservers”内容设置为：

```
Slave1.Hadoop
Slave2.Hadoop
Slave3.Hadoop
```

这就意味着，HBase RegionServer 运行在 Slave1.Hadoop、Slave2.Hadoop、Slave3.Hadoop 三台机器上。

```
[root@Master conf]# vim regionservers

Slave1.Hadoop
Slave2.Hadoop
Slave3.Hadoop
```

第四步：添加 HBase 环境变量

在“**/etc/profile**”文件的尾部添加以下内容，并使其有效（source /etc/profile）：

```
# set hbase environment
export HBASE_HOME=/usr/hadoop
export PATH=$PATH :$ HBASE_HOME /bin
```

```
# set java environment
export JAVA_HOME=/usr/java/jdk1.6.0_31
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib
export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin

# set hadoop environment
export HADOOP_HOME=/usr/hadoop
export HADOOP_HOME_WARN_SUPPRESS=1
export PATH=$PATH:$HADOOP_HOME/bin

# set hbase environment
export HBASE_HOME=/usr/hbase
export PATH=$PATH:$HBASE_HOME/bin

"/etc/profile" 83L, 1854C 已写入
[root@Master ~]#
```

从上图中得知，我们还可以看到之前设置的 Java 和 Hadoop 的环境变量。

第五步：复制 HBase 到其他节点

将 Master 上配置好的 hadoop 所在文件夹“**/usr/hbase**”复制到所有的 Slave 的“/usr”目录下，用下面命令格式进行。

```
scp　-r　/usr/hbase　　root@服务器 IP:/usr/
```

例如：从“Master.Hadoop”到“Slave1.Hadoop”复制配置 HBase 的文件。

```
[root@Master ~]# scp -r /usr/hbase  root@192.168.1.3:/usr/
root@192.168.1.3's password:
```

上图中以 **root** 用户进行复制，用 **root** 进行“**scp**”时，扔提示让你输入“Slave1.Hadoop”服务器用户 root 的密码。

查看“**Slave1.Hadoop**”服务器的“**/usr**”目录下是否已经存在“**hbase**”文件夹，确认已经复制成功。查看结果如下：

```
[root@Slave1 usr]# pwd
/usr
[root@Slave1 usr]# ll
总用量 108
dr-xr-xr-x.  2 root   root   24576  2月 23 06:32 bin
drwxr-xr-x.  2 root   root    4096 11月 11 2010 etc
drwxr-xr-x.  2 root   root    4096 11月 11 2010 games
drwxr-xr-x. 16 hadoop hadoop  4096  2月 29 20:52 hadoop
drwxr-xr-x.  8 root   root    4096  3月 18 05:42 hbase
drwxr-xr-x. 33 root   root    4096  2月 23 04:51 include
drwxr-xr-x.  3 root   root    4096  2月 28 05:19 java
dr-xr-xr-x. 53 root   root   32768  2月 23 06:32 lib
drwxr-xr-x. 14 root   root    4096  2月 23 06:32 libexec
drwxr-xr-x. 11 root   root    4096  2月 23 04:45 local
dr-xr-xr-x.  2 root   root    4096  2月 24 03:19 sbin
drwxr-xr-x. 95 root   root    4096  2月 23 04:53 share
drwxr-xr-x.  4 root   root    4096  2月 23 04:45 src
lrwxrwxrwx.  1 root   root      10  2月 23 04:45 tmp -> ../var/tmp
[root@Slave1 usr]#
```

从上图中知道，hbase 文件夹确实已经复制了，但是我们发现 hbase **权限**是 root，所以我们现在要给“Slave1.Hadoop”服务器上的用户 hadoop 添加对“/usr/hbase”权限。

以 **root** 用户登录“Slave1.Hadoop”，执行下面命令。

```
chown  -R  hadoop:hadoop（用户名：用户组）  hbase（文件夹）
```

```
[root@Slave1 usr]# ll | grep hbase
drwxr-xr-x.  8 root   root    4096  3月 18 05:42 hbase
[root@Slave1 usr]# chown -R hadoop:hadoop hbase
[root@Slave1 usr]# ll | grep hbase
drwxr-xr-x.  8 hadoop hadoop  4096  3月 18 05:42 hbase
[root@Slave1 usr]#
```

接着在“Slave1 .Hadoop”上按照“**第四步**”修改“/etc/profile”文件，添加完内容之后，并使其有效（source /etc/profile）。到此为此在一台 Slave 机器上的 HBase 配置就结束了。剩下的事儿就是照葫芦画瓢把剩余的几台 Slave 机器按照《**从“Master.Hadoop”到“Slave1.Hadoop”复制 HBase 的安装包。**》这个例子进行部署 Hadoop。

4.4 启动HBase

在“Master.Hadoop”机器上，使用用户“**hadoop**”执行下面命令启动 HBase 数据库。

start-hbase.sh

加入你忘记了设置 HBase 的环境变量，用上面命令是不能启动的，系统会提示你该命令没有找到。这是你可以用下面方式启动。

/usr/hbase/bin/start-hbase.sh

HBase 启动如下图所示：

```
[hadoop@Master ~]$ start-hbase.sh
Slave3.Hadoop: starting zookeeper, logging to /usr/hbase/logs/hbase-hadoop-zookeeper-Slave3.Hadoop.out
Slave2.Hadoop: starting zookeeper, logging to /usr/hbase/logs/hbase-hadoop-zookeeper-Slave2.Hadoop.out
Slave1.Hadoop: starting zookeeper, logging to /usr/hbase/logs/hbase-hadoop-zookeeper-Slave1.Hadoop.out
starting master, logging to /usr/hbase/logs/hbase-hadoop-master-Master.Hadoop.out
Slave1.Hadoop: starting regionserver, logging to /usr/hbase/logs/hbase-hadoop-regionserver-Slave1.Hadoop.out
Slave2.Hadoop: starting regionserver, logging to /usr/hbase/logs/hbase-hadoop-regionserver-Slave2.Hadoop.out
Slave3.Hadoop: starting regionserver, logging to /usr/hbase/logs/hbase-hadoop-regionserver-Slave3.Hadoop.out
```

用 jps 工具测验一下 HBase 集群进程。

- “**Master.Hadoop**”机器

```
[hadoop@Master ~]$ jps
2611 JobTracker
3400 HMaster
2518 SecondaryNameNode
3500 Jps
2351 NameNode
[hadoop@Master ~]$
```

- “**Slave*.Hadoop**”机器

```
[hadoop@Slave1 conf]$ jps
5497 Jps
2602 TaskTracker
2515 DataNode
5263 HQuorumPeer
5323 HRegionServer
[hadoop@Slave1 conf]$
```

通过 SecureCRT 查看 HBase 在 Hadoop 集群的 HDFS 中是否自动生成了“/hbase”目录，用于存放数据。查看结果如下所示。

```
[hadoop@Master ~]$ hadoop fs -ls /
Found 5 items
-rw-r--r--   3 hadoop supergroup     283236 2012-03-11 00:42 /EKB.owl
drwxr-xr-x   - hadoop supergroup          0 2012-03-18 22:40 /hbase
drwxr-xr-x   - hadoop supergroup          0 2012-03-16 00:34 /lib
drwxr-xr-x   - hadoop supergroup          0 2012-03-16 22:19 /user
drwxr-xr-x   - hadoop supergroup          0 2012-03-18 00:53 /usr
[hadoop@Master ~]$
```

4.5 关闭HBase

使用下面命令即可关闭 HBase 数据库。

```
stop-hbase.sh
```

HBase 关闭如下图所示：

```
[hadoop@Master ~]$ stop-hbase.sh
stopping hbase..........
Slave3.Hadoop: stopping zookeeper.
Slave1.Hadoop: stopping zookeeper.
Slave2.Hadoop: stopping zookeeper.
[hadoop@Master ~]$
```

5、HBase用户界面

5.1 Master页面

通过地址“http:192.168.1.2:60010/master.jsp”可以查看 HBase 的相关信息，主要包含的信息如下：

（1）Master 属性信息包含了当前集群的详细信息，从上往下依次为 **HBase** 的版本及编译信息、**Hadoop** 的版本及编译信息、**HBase** 根目录的路径、**Region** 服务器的平均负载以及 **Zookeeper** Quorums 的地址。

Attributes

Attribute Name	Value	Description
HBase Version	0.92.0, r1231986	HBase version and revision
HBase Compiled	Mon Jan 16 13:16:35 UTC 2012, jenkins	When HBase version was compiled and by whom
Hadoop Version	1.0.0, r1214675	Hadoop version and revision
Hadoop Compiled	Fri Dec 16 20:01:27 UTC 2011, hortonfo	When Hadoop version was compiled and by whom
HBase Root Directory	hdfs://Master.Hadoop:9000/hbase	Location of HBase home directory
HBase Cluster ID	969cc53c-8d27-400c-8439-8805dce7e3e3	Unique identifier generated for each HBase cluster
Load average	0.67	Average number of regions per regionserver. Naive computation.
Zookeeper Quorum	Slave2.Hadoop:2181,Slave1.Hadoop:2181,Slave3.Hadoop:2181	Addresses of all registered ZK servers. For more, see zk dump.
Coprocessors	[]	Coprocessors currently loaded loaded by the master
HMaster Start Time	Mon Mar 19 00:32:54 CST 2012	Date stamp of when this HMaster was started
HMaster Active Time	Mon Mar 19 00:32:54 CST 2012	Date stamp of when this HMaster became active

（2）目录表信息包含两个目录：**-ROOT-**和**.META.**。

Tables

Catalog Table	Description
-ROOT-	The -ROOT- table holds references to all .META. regions.
.META.	The .META. table holds references to all User Table regions

（3）用户表信息给出了 **HBase** 中的表信息及相关属性。

1 table(s) in set. [Details]

User Table	Description
tab_student	{NAME => 'tab_student', FAMILIES => [{NAME => 'age', MIN_VERSIONS => '0'}, {NAME => 'id', MIN_VERSIONS => '0'}, {NAME => 'name', MIN_VERSIONS => '0'}]}

备注：当系统没有创建用户表时，不显示任何信息。

（4）**Region 服务器**信息给出了所有 Region 服务器的地址。

Region Servers

	ServerName	Start time	Load
	Slave1.Hadoop,60020,1332088436608	Mon Mar 19 00:33:56 CST 2012	requestsPerSecond=0, numberOfOnlineRegions=1, usedHeapMB=25, maxHeapMB=998
	Slave2.Hadoop,60020,1332088311179	Mon Mar 19 00:31:51 CST 2012	requestsPerSecond=0, numberOfOnlineRegions=0, usedHeapMB=29, maxHeapMB=998
	Slave3.Hadoop,60020,1332089705277	Mon Mar 19 00:55:05 CST 2012	requestsPerSecond=0, numberOfOnlineRegions=1, usedHeapMB=29, maxHeapMB=998
Total:	servers: 3		requestsPerSecond=0, numberOfOnlineRegions=2

5.2 ZooKeeper页面

通过 Master 页面中 Master 属性提供的链接，可以进入 ZooKeeper 页面，该页面显示了 HBase 的根目录、当前的主 **Master** 地址、保存-**ROOT**-表的 Region 服务器地址、其他 Region 服务器的地址及 ZooKeeper 的一些内部信息，如下图所示。

```
HBase is rooted at /hbase
Master address: Master.Hadoop,60000,1332088374111
Region server holding ROOT: Slave3.Hadoop,60020,1332089705277
Region servers:
 Slave2.Hadoop,60020,1332088311179
 Slave3.Hadoop,60020,1332089705277
 Slave1.Hadoop,60020,1332088436608
Quorum Server Statistics:
 Slave2.Hadoop:2181
  Zookeeper version: 3.4.2-1221870, built on 12/21/2011 20:46 GMT
  Clients:
   /192.168.1.4:55694[1](queued=0,recved=22,sent=24)
   /192.168.1.2:52372[0](queued=0,recved=1,sent=0)

  Latency min/avg/max: 0/4/66
  Received: 24
  Sent: 25
  Outstanding: 0
  Zxid: 0x40000001b
  Mode: follower
  Node count: 17
 Slave1.Hadoop:2181
  Zookeeper version: 3.4.2-1221870, built on 12/21/2011 20:46 GMT
  Clients:
   /192.168.1.2:57479[0](queued=0,recved=1,sent=0)
   /192.168.1.4:56519[1](queued=0,recved=34,sent=41)
   /192.168.1.2:46581[1](queued=0,recved=90,sent=105)

  Latency min/avg/max: 0/3/95
  Received: 126
  Sent: 147
  Outstanding: 0
  Zxid: 0x40000001b
  Mode: follower
  Node count: 17
 Slave3.Hadoop:2181
  Zookeeper version: 3.4.2-1221870, built on 12/21/2011 20:46 GMT
  Clients:
   /192.168.1.5:36044[1](queued=0,recved=45,sent=52)
   /192.168.1.3:42582[1](queued=0,recved=43,sent=50)
   /192.168.1.5:36045[1](queued=0,recved=22,sent=24)
   /192.168.1.2:59133[1](queued=0,recved=134,sent=136)
   /192.168.1.2:49571[0](queued=0,recved=1,sent=0)
   /192.168.1.3:42583[1](queued=0,recved=28,sent=30)

  Latency min/avg/max: 0/0/12
  Received: 274
  Sent: 293
  Outstanding: 0
  Zxid: 0x40000001b
  Mode: leader
  Node count: 17
```

5.3 User Tables页面

通过 Master 页面中用户表信息提供的链接，可以进入用户表页面，如下图所示。该页面给出了表当前是否可以用以及表在 Region 服务器上的信息。同时还提供了根据行健合并及拆分表的操作。

5.4 Region服务器页面

通过 Master 页面中 Region 服务器信息提供的链接，可以进入 Region 服务器页面，该页面显示了 Region 服务器的基本属性和其上所有 Regions 的信息。如下图所示。

参考文献

感谢以下文章的编写作者，没有你们的铺路，我或许会走得很艰难，参考不分先后，贡献同等珍贵。

【1】Hadoop 实战——陆嘉恒——机械工业出版社

【2】实战 Hadoop——刘鹏——电子工业出版社

【3】Hadoop0.20.203.0+Hbase0.90.4 完全分布式配置

地址：http://www.cnblogs.com/flyoung2008/archive/2011/12/02/2272761.html

【4】Hbase 安装配置（含分布式 ZooKeeper）

地址：http://linuxjcq.blog.51cto.com/3042600/760634

【5】HBase 分布式模式的安装、配置和使用

地址：http://www.mcncc.com/thread-80401-1-1.html

【6】HBase 技术介绍

地址：http://www.searchtb.com/2011/01/understanding-hbase.html

【7】HBase 入门笔记（四）--完全分布式 HBase 集群安装配置

地址：http://www.cnblogs.com/ventlam/archive/2011/01/22/HBaseCluster.html

【8】hbase 无法启动问题 时间设置

地址：http://taoo.iteye.com/blog/1266576

【9】hbase 结点之间时间不一致造成 regionserver 启动失败

地址：http://www.gather-data.info/blog/?p=2454