第一章 Hbase 介绍

行存储：**表中列是固定的，不能动态增加（建立索引和视图花费大）（以一行记录为单位）(读取是从行的第一列到最后一列，读写一致）**

列存储：**以列数据集合为单位（读取是列数据集中的一段或 all，写入时，一行记录拆分成多列，每一列数据追加到对应列末尾）**

（都是从上到下，从左到右排列）

1.1 对比:

表 1-3 行/列存储方式优缺点对比

项目	行存储	列存储
优点	写入效率高，提供数据完整性保证	读取过程有冗余，适合数据定长的大数据计算
缺点	数据读取有冗余现象，影响计算速度	缺乏数据完整性保证，写入效率低
改进	优化的存储格式，保证能够在内存中快速删除冗余数据	多磁盘多线程并行读/写（需要增加运行成本和修改软件）
应用环境	商业领域、互联网	互联网

1.2 HDFS 分布式存储的特点

优点：

1) 高容错性：

上传数据自动保存多个副本（副本多，容错性高）

若某副本丢失，会复制其他机器上副本（不用关系怎么实现）

2) 适合大数据处理

GB、TB、甚至 PB，处理百万规模数据

3) 流式文件写入

一次写入，多次读取

一旦写入，不能更改，只能增加，保持数据一致性

4) 可构建在廉价机器上

通过多副本提高可靠性，提供容错和恢复机制

缺点：

1) 不适合低延迟数据访问

高吞吐量，往往代价是高延迟

2) 无法高效存储大量小文件

Example：一个 1GB 文件，分成 8 个 128mb 的 split，并分配 8 个 map 任务管理，而 10000 个 100kb 的文件，会被 10000 个 map 任务管理（不值得）

3) 不支持多用户写入及任意修改文件

只有写入者，写操作只能在文件末尾完成

1.3 使用场景

1. 平台类

捕获来自各种数据源的增量数据（细水长流、日积月累），如日志

收集系统 Flume

2. 内容服务类

直接存放、读取，不仅指人，也指物，比如购物收藏夹、交易数据、聊天记录

3. 信息展示类

快速查询、高吞吐量，如天猫双十一

第二章　Hbase 模型与系统架构

2.1 相关概念

1. Table（表）

数据组织进一张表里，表包含行、列，一个 Hbase 表由多行组成

2. Row（行）

行键来唯一标识（Row Key），行包含多个 key 和一到多个包含值的列

3. Column（列）

包含分隔开的列族和列限定符

4. Column Family（列族）

列族包含一个或多个相关列，列都以列族作为前缀，如：anchor:name ,anchor:tel 都属于 anchor 列族

5. Column Qualifier（列限定符）

   列的标识符是列族中数据的索引，如：给定一个列族 content，那么标识符是 content：html，也可以是 content：pdf。(列族是创建表格时确定的，但是列的标识符是动态的)

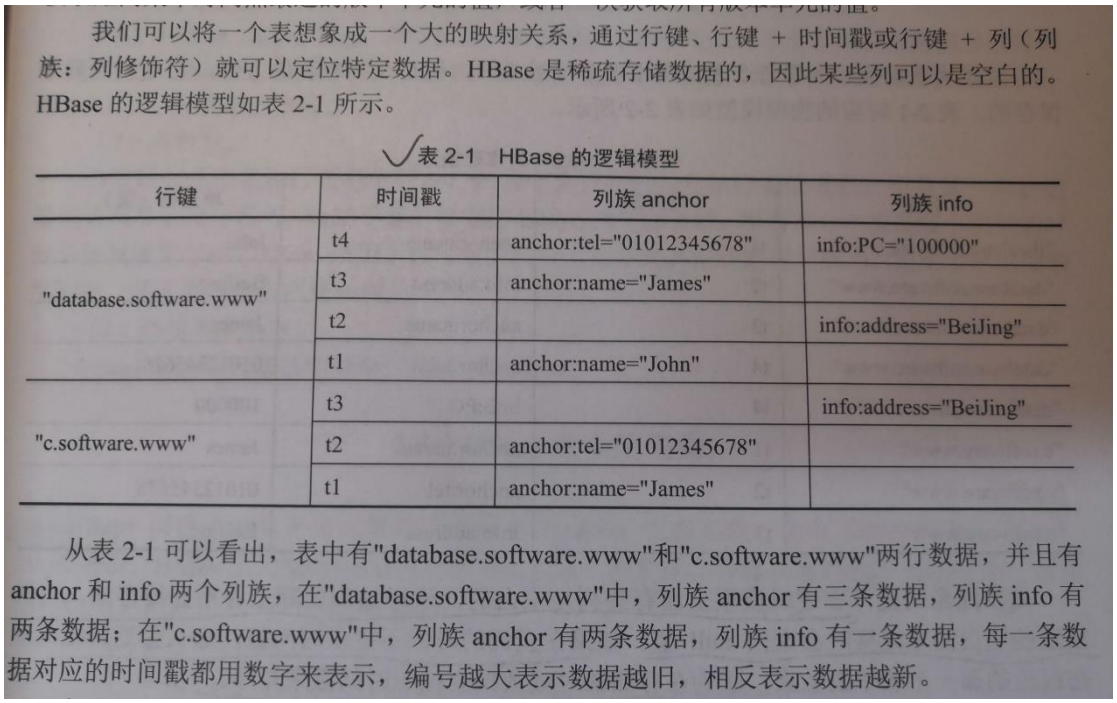
6. Cell（单元格）

   由 行、列、列标识符、值和时间戳（每个 cell 都保存同一份数据的多个版本，默认 3)

7. Timestamp（时间戳）

   写在值旁边用于区分值的版本的数据（默认情况下，每个单元中数据插入时都会用时间戳来进行版本标识)

2.2 Hbase 的逻辑模型和物理模型



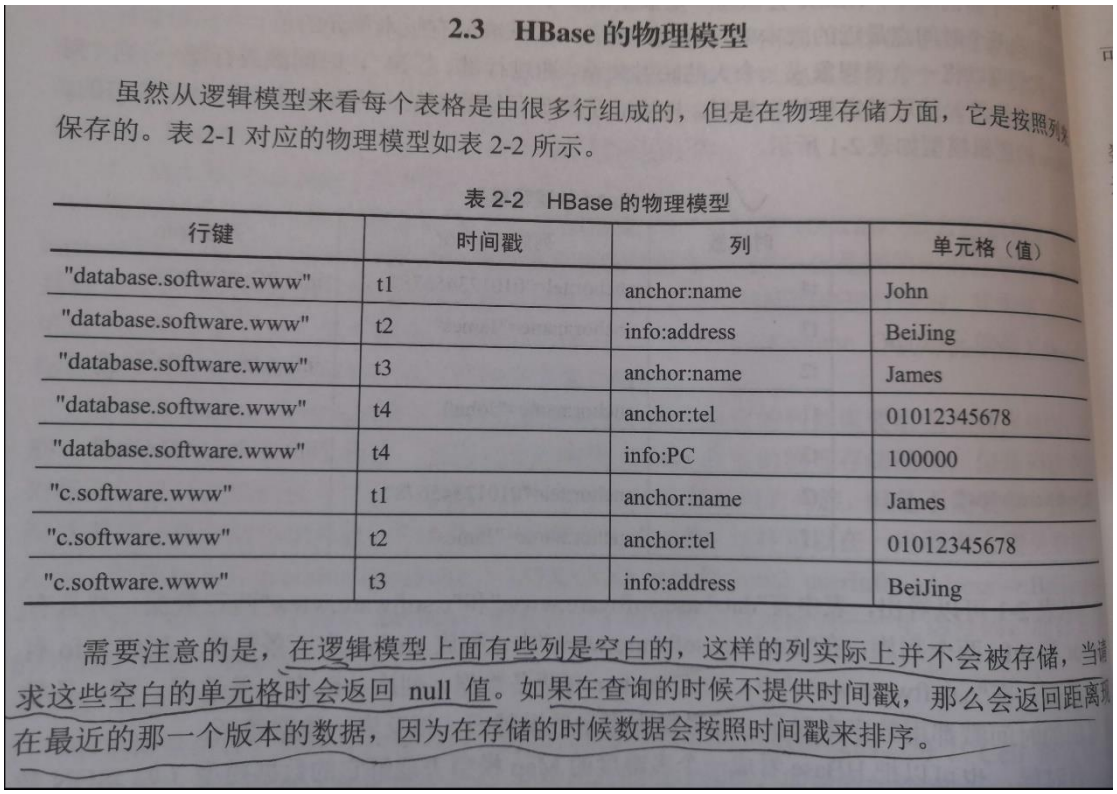
我们可以将一个表想象成一个大的映射关系，通过行键、行键 + 时间戳或行键 + 列（列族：列修饰符）就可以定位特定数据。HBase 是稀疏存储数据的，因此某些列可以是空白的。HBase 的逻辑模型如表 2-1 所示。

表 2-1 HBase 的逻辑模型

行键	时间戳	列族 anchor	列族 info
"database.software.www"	t4	anchor:tel="01012345678"	info:PC="100000"
	t3	anchor:name="James"	
	t2		info:address="BeiJing"
	t1	anchor:name="John"	
"c.software.www"	t3		info:address="BeiJing"
	t2	anchor:tel="01012345678"	
	t1	anchor:name="James"	

从表 2-1 可以看出，表中有"database.software.www"和"c.software.www"两行数据，并且有 anchor 和 info 两个列族，在"database.software.www"中，列族 anchor 有三条数据，列族 info 有两条数据；在"c.software.www"中，列族 anchor 有两条数据，列族 info 有一条数据，每一条数据对应的时间戳都用数字来表示，编号越大表示数据越旧，相反表示数据越新。

（逻辑模型）



2.3 HBase 的物理模型

虽然从逻辑模型来看每个表格是由很多行组成的，但是在物理存储方面，它是按照列保存的。表 2-1 对应的物理模型如表 2-2 所示。

表 2-2 HBase 的物理模型

行键	时间戳	列	单元格（值）
"database.software.www"	t1	anchor:name	John
"database.software.www"	t2	info:address	BeiJing
"database.software.www"	t3	anchor:name	James
"database.software.www"	t4	anchor:tel	01012345678
"database.software.www"	t4	info:PC	100000
"c.software.www"	t1	anchor:name	James
"c.software.www"	t2	anchor:tel	01012345678
"c.software.www"	t3	info:address	BeiJing

需要注意的是，在逻辑模型上面有些列是空白的，这样的列实际上并不会被存储，当请求这些空白的单元格时会返回 null 值。如果在查询的时候不提供时间戳，那么会返回距离现在最近的那一个版本的数据，因为在存储的时候数据会按照时间戳来排序。

（物理模型）

2.3 Hbase 的特点

1. **容量巨大**

   纵向和横向上支持大数据存储，可达百亿行、列

2. **面向列**

   列（族）存储和权限控制、列（族）独立检索

   （在查询少数几个字段时候能大大减少读取的数据量）

3. **稀疏性**

   不存储值为空的列，因此 Hbase 的表是稀疏的，节省空间

4. **数据多版本**

   每个单元中的数据多个版本，可查询历史版本数据

5. **可扩展性、**

   存储在 HDFS 上，由于 HDFS 具有动态增加节点的特性，因此 Hbase 容易集群扩展

6. **高可靠性**

   WAL（预写日志）机制保证了数据写入不会因为集群故障导致数据丢失；

   Hbase 位于 HDFS 上，而 HDFS 有数据备份功能

   Hbase 引入 ZooKeeper，避免 Master 出现单点故障

7. **高性能**

   基于列的数据库会将每列单独存放，当查找一个数量较小的时候查找速度快

   读写缓存存机制，具有高并发读写能力

8. **数据类型单一**

   Hbase 的数据都是字符串 string

2.4 Hbase 的系统架构

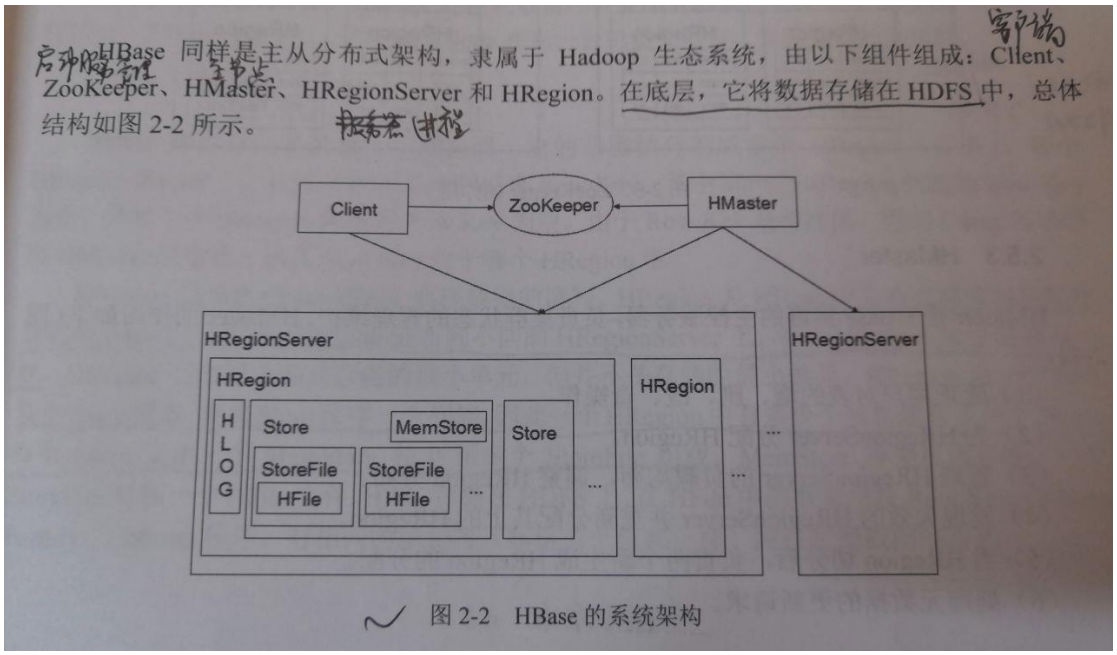


图 2-2 HBase 的系统架构

Client、ZooKeeper、HMaster、HReginServer 和 HRegion

（底层存储在 HDFS 中）

1. Client（客户端）：**包含访问 HBase 的接口，使用 RPC 机制与 HMaster 和 HReginServer 进行通信并维护 Cache 来加快 Hbase 的访问**

2. ZooKeeper：

   存储所有 HRegion 的寻址入口，完成数据读写

监控 HReginServer 的上线、下线信息，并通知 HMaster

存放 HBase 集群的元数据以及集群的状态信息

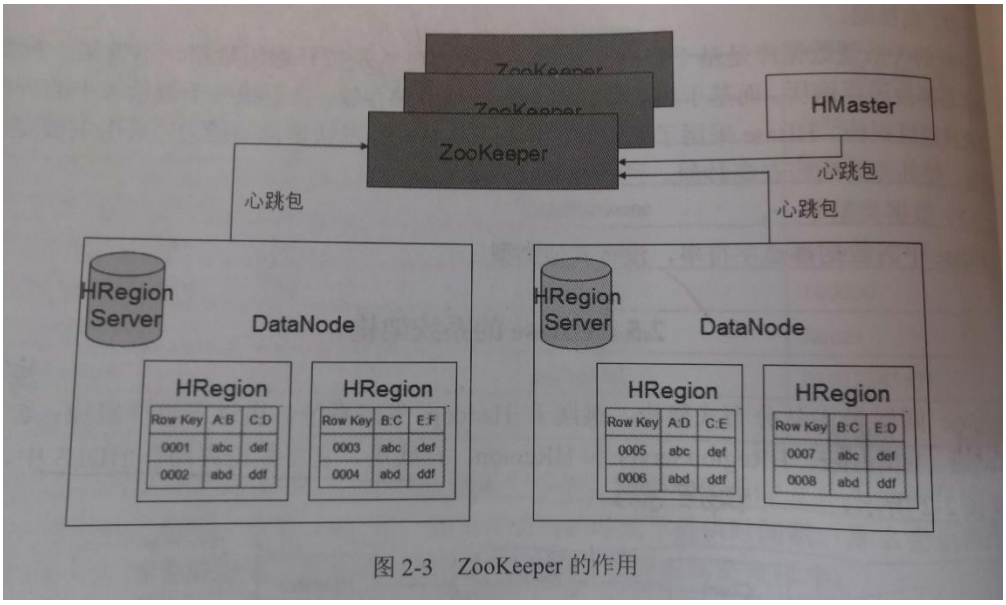


图 2-3　ZooKeeper 的作用

3. HMaster：

- 管理增删改查操作
- 为 HReginServer 分配 HRegin
- 管理 HReginServer 的负债均衡，调整 HRegin 分布
- 发现失效的 HReginServer 并重新分配 HRegin
- 当 HRegin 切分后，负责两个新 HRegin 的分配
- 处理元数据的更新请求

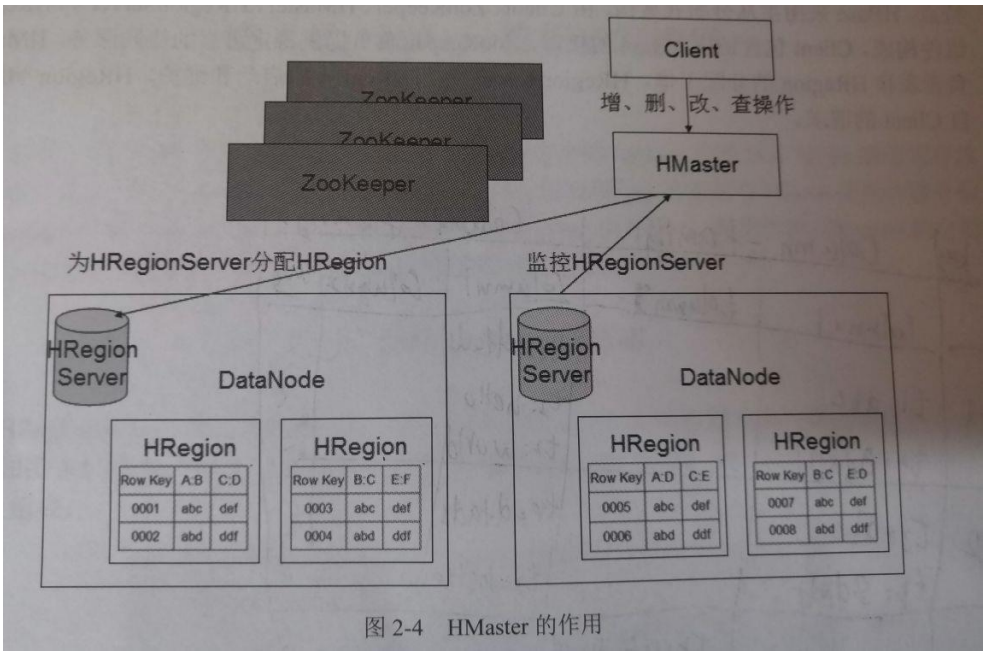


图 2-4　HMaster 的作用

4. HReginServer：

具体对外提供服务的进程，主要负责维护 HMaster 分配给它的 HRegin 的启动和管理（一个 HReginServer 包含 HRegin）

5. HReginServer：

HRegin 是 Hbase 中分布式存储和负债均衡的最小单位，不同的 HRegin 分配到不同 HReginServer 上。

HRegin 由一个或多个 Store 组成，每个 Store 保存一个列族，因此一个 HRegin 中有多少个列族就有多少个 Store。

每个 Store 又由一个 MemStore 和 0~多个 StoreFile 组成