HZ BOOKS
华章IT

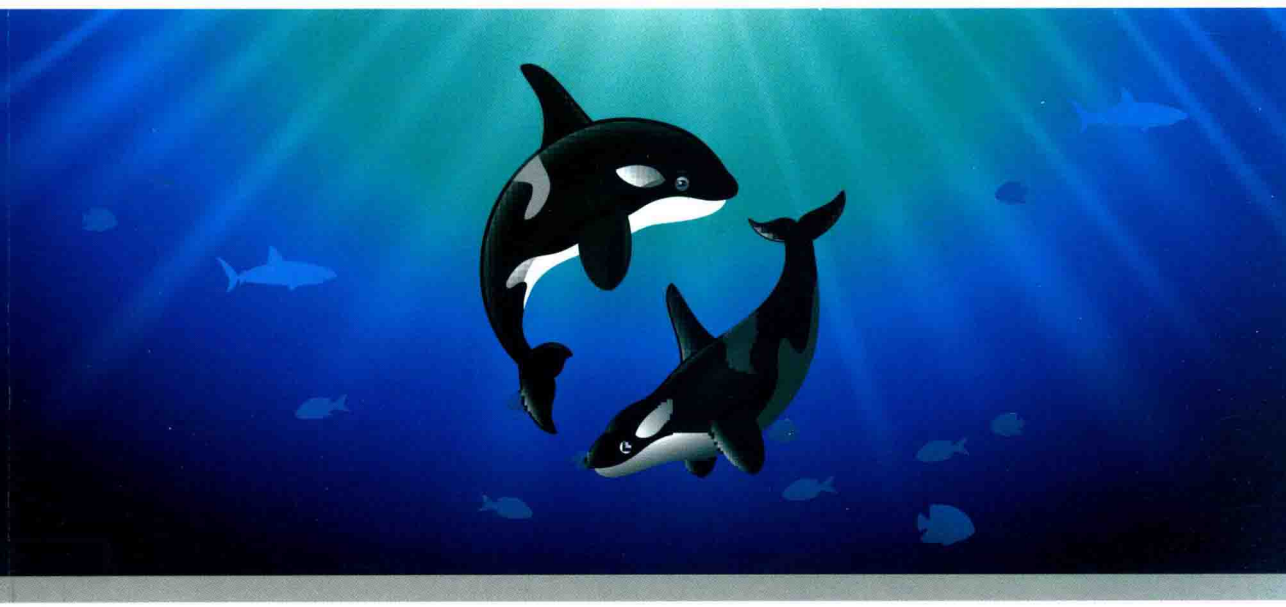
HBase PMC成员与网易资深工程师倾力打造，多位技术专家联袂推荐。

深入HBase内核，抽丝剥茧，剖析HBase数据库的基础理论与开发运维。

数据库技术丛书

HBase Principle and Practice

HBase原理与实践



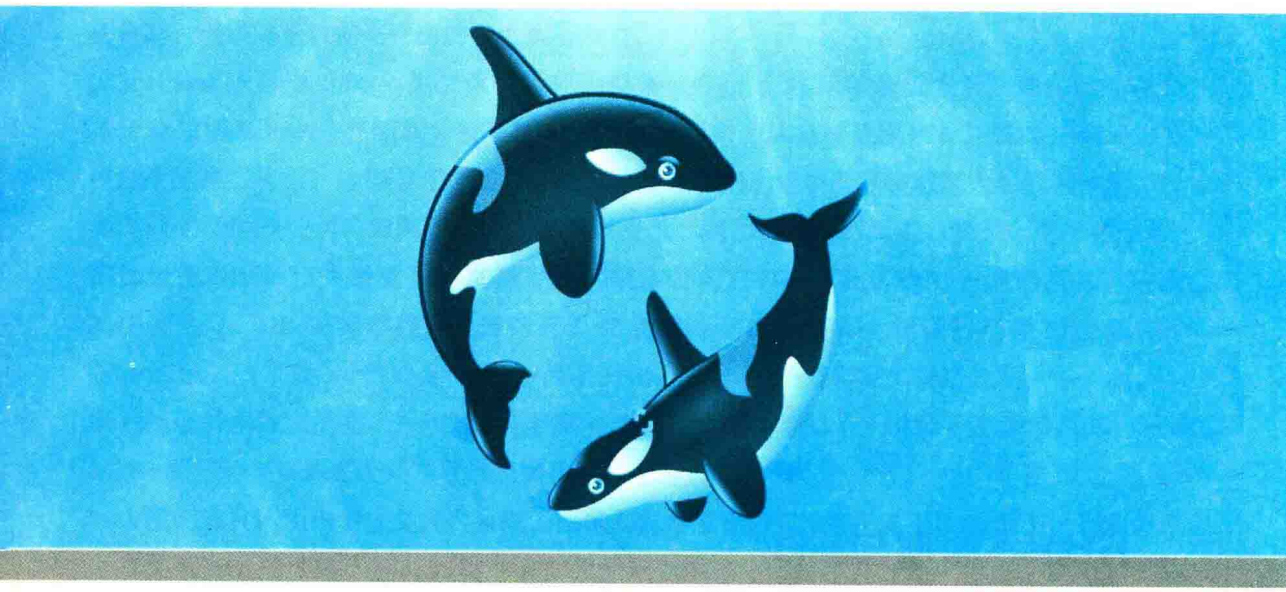
胡争 范欣欣 著

机械工业出版社
China Machine Press

数据库 技术丛书

HBase Principle and Practice

HBase原理与实践



胡争 范欣欣 著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

HBase 原理与实践 / 胡争，范欣欣著 . —北京：机械工业出版社，2019.8
（数据库技术丛书）

ISBN 978-7-111-63495-9

I. H… II. ①胡… ②范… III. 计算机网络－信息存贮－研究 IV. TP393.071

中国版本图书馆 CIP 数据核字（2019）第 176442 号

HBase 原理与实践

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：吴 怡 责任校对：李秋荣
印 刷：北京诚信伟业印刷有限公司 版 次：2019 年 9 月第 1 版第 1 次印刷
开 本：186mm×240mm 1/16 印 张：20
书 号：ISBN 978-7-111-63495-9 定 价：129.00 元

客服电话：（010）88361066 88379833 68326294 投稿热线：（010）88379604
华章网站：www.hzbook.com 读者信箱：hzit@hzbook.com

Preface 前　言

Apache HBase是基于Apache Hadoop构建的一个高可用、高性能、多版本的分布式NoSQL数据库，是Google BigTable的开源实现，通过在廉价服务器上搭建大规模结构化存储集群，提供海量数据高性能的随机读写能力。

HBase项目自2006年提交第一行代码以来，经历了13年的蓬勃发展。现在已经有大量企业采用HBase来存储和分析飞速增长的业务数据。从全球范围来看，国内HBase的关注度更是高居榜首，这得益于国内互联网、移动互联网、物联网等领域庞大的数据体量。诸多国内大型科技公司，如阿里巴巴、小米、腾讯、网易、华为、滴滴、快手、中国移动等，都已经把HBase作为极重要的基础设施，很多公司对HBase社区也有长期的投入。截至2019年8月，HBase全球社区已经拥有了83位HBase Committer，而国内就有20位左右的Committer，占了近1/4的比例[⊖]。近一两年，HBase在国内更是得到了长足的发展，2018年中国HBase技术社区成立，一年时间里社区在多个城市相继组织了9次线下技术沙龙活动，为HBase更好地在国内各公司茁壮成长做出了卓越的贡献。

我们和社区用户多次交流后发现，很多人都希望我们能推荐一本HBase的书。当前市面上有关HBase的书籍大部分都集中于如何使用HBase，例如部署HBase集群，使用客户端API进行读写操作以及协处理器等，诚然，这些内容对快速掌握和使用HBase非常有好处，但是许多HBase使用者并不满足于此，他们更希望能了解和掌握其内部运行原理。因此，当机械工业出版社的吴怡编辑询问我们是否有想法为HBase写一本书时，我们毫不犹豫地答应了。

本书从设计的角度对HBase的整个体系架构和各核心组件进行系统的分析和讲解。与此

⊖ 目前中国区总共有18位HBase Committer，其中6位HBase PMC成员（张铎@小米、张洸豪、胡争@小米，沈春辉、李钰、杨文龙@阿里巴巴）。值得一提的是，张铎于2019年7月18日当选为Apache HBase项目的主席，他将在接下来的时间里带领Apache HBase社区实现HBase项目的更新迭代。

同时，还介绍常用的性能调优策略以及问题诊断的方法和技巧，帮助读者更好地在实际生产环境中实践。另外，本书最后章节集中介绍 HBase 2.x 版本的核心特性，例如 Procedure v2、In Memory Compaction 以及 MOB 等。

本书主要内容

本书不是一本入门级读物，本书面向那些使用 HBase 作为数据库后端存储的应用程序开发者、有一定经验的运维人员和对 HBase 内核设计感兴趣的人。

如果你想深入了解 HBase 的每个组件是如何工作的，如果你想更好地运维或者调优你的 HBase 集群，如果你想了解 HBase 2.x 版本的核心特性，就请阅读本书。想要更好地阅读本书，需要具备如下先决条件：

- 了解 HBase 的基本操作。
- 了解 C、Java 等高级语言。
- 对一些基本算法有所了解，因为本书会从源代码层面分析 HBase 的工作机制，如果你能了解这些算法，会使你更深入地理解 HBase。

本书共有 16 章，可以分为 6 个部分。

第一部分：HBase 基础部分，包含第 1、2 章。其中，第 1 章主要介绍 HBase 系统的发展历史、数据模型以及体系结构，第 2 章主要介绍 HBase 系统中常用的数据结构以及基础算法。

第二部分：HBase 系统相关组件，包含第 3、4、5 章。其中，第 3 章重点介绍 HBase 所依赖的核心组件，包括 ZooKeeper、HDFS 等，第 4 章介绍 HBase 客户端组件实现，第 5 章介绍 RegionServer 内部组件的实现。

第三部分：HBase 核心工作原理，包含第 6、7、8、9、10、11 章。其中，第 6 章详细分析 HBase 读写流程，第 7 章介绍 HBase Compaction 的实现原理，第 8 章介绍 HBase 中 Region 的迁移、合并以及分裂等操作是如何实现的，第 9 章介绍 RegionServer 宕机后如何通过 HLog 进行数据恢复，第 10 章介绍 HBase 不同集群之间的复制是如何实现的，第 11 章介绍 HBase 如何通过 Snapshot 机制完成数据的备份和恢复。

第四部分：HBase 运维调优实践，包含第 12、13、14 章。其中，第 12 章介绍 HBase 集群常用的运维管理操作，包括集群如何有效监控，基准性能如何测试等，第 13 章集中介绍 HBase 集群的常用调优技巧，第 14 章重点分析几个 HBase 实际运维案例，通过案例分析介绍 HBase 集群定位和处理问题的技巧。

第五部分：HBase 2.x 核心特性（第 15 章），介绍 HBase 最新 2.x 版本的核心功能与特性。

第六部分：HBase 高级话题（第 16 章），介绍社区中比较热门的二级索引话题，以及

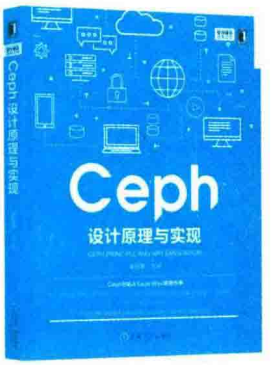
HBase 内核的开发与测试。

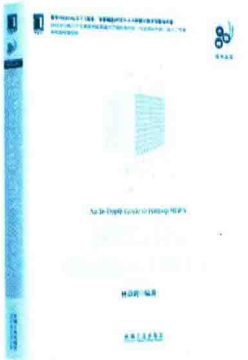
本书的六个部分都是相互独立的话题，读者完全可以从书中任何一个部分开始阅读。当然，如果你想更加系统地学习 HBase，建议你从前往后逐章阅读。

致谢

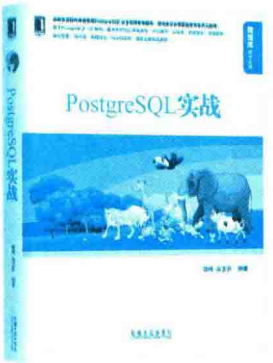
在编写本书的过程中，我们非常感谢给予了我们如此多帮助和鼓励的朋友、家人以及同事们。首先感谢 HBase 官方社区的开发者，是他们极其卓越的工作让我们有机会写这样一本书。另外，还要感谢许许多多中国 HBase 技术社区的小伙伴，感谢他们提供丰富的 HBase 使用场景和相关解决方案，他们的经验和分享对推广和普及 HBase 项目做出了重大贡献。同时感谢我们的家人，没有他们的鼓励和支持，本书不可能完成。最后，一份特别的感谢要送给本书策划编辑吴怡，感谢她在全书撰写过程中所给予的详细指点以及有用的建议。

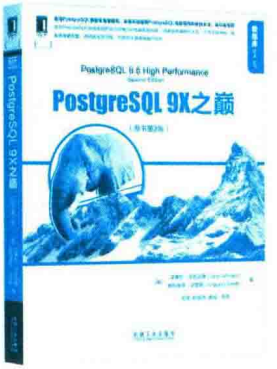
推荐阅读


Ceph
设计原理与实现

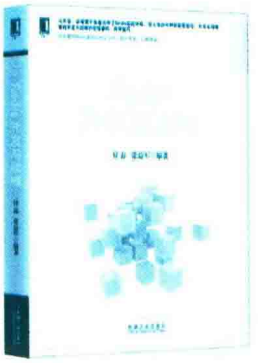


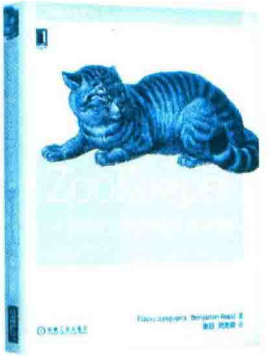

大规模分布式存储系统
原理解析与架构实战


PostgreSQL实战

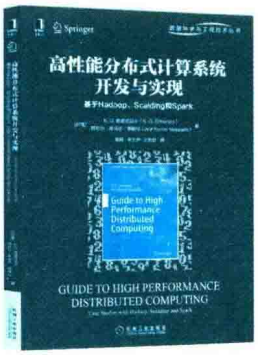

PostgreSQL 9.6 High Performance
PostgreSQL 9X之巅


Redis
设计与实现





推荐阅读


高性能分布式计算系统
开发与实现
基于Hadoop、Scalding和Spark
Guide to High Performance Distributed Computing
GUIDE TO HIGH PERFORMANCE DISTRIBUTED COMPUTING


分布式实时系统
原理与设计方法
Real-Time Systems
Real-Time Systems


数据库系统
设计、实现与管理（进阶篇）
Database Systems
DATABASE SYSTEMS


数据库系统
设计、实现与管理（基础篇）
Database Systems
DATABASE SYSTEMS


分布式控制系统设计
模式语言方法
Designing Distributed Control Systems
DESIGNING DISTRIBUTED CONTROL SYSTEMS


系统编程
分布式应用的设计与开发
Systems Programming
SYSTEMS PROGRAMMING

目　录 *Contents*

第 1 章 HBase 概述

HBase 是目前非常热门的一款分布式 KV（KeyValue，键值）数据库系统，无论是互联网行业还是其他传统 IT 行业都在大量使用。尤其是近几年随着国内大数据理念的普及，HBase 凭借其高可靠、易扩展、高性能以及成熟的社区支持，受到越来越多企业的青睐。许多大数据系统都将 HBase 作为底层数据存储服务，例如 Kylin、OpenTSDB 等。

本章作为全书的开篇，将从 HBase 的历史发展、数据模型、体系结构、系统特性几个方面，向读者介绍这位主角。

1.1 HBase 前生今世

1. HBase 历史发展

要说清楚 HBase 的来龙去脉，还得从 Google 当年风靡一时的“三篇论文”——GFS、MapReduce、BigTable 说起。2003 年 Google 在 SOSP 会议上发表了大数据历史上第一篇公认的革命性论文——《GFS: The Google File System》，之所以称其为“革命性”是有多方面原因的：首先，Google 在该论文中第一次揭示了如何在大量廉价机器基础上存储海量数据，这让人们第一次意识到海量数据可以在不需要任何高端设备的前提下实现存储，换句话说，任何一家公司都有技术实力存储海量数据，这为之后流行的海量数据处理奠定了坚实的基础。其次，GFS 体现了非常超前的设计思想，以至于十几年之后的今天依然指导着大量的分布式系统设计，可以说，任何从事分布式系统开发的人都有必要反复阅读这篇经典论文。

2004 年，Google 又发表了另一篇非常重要的论文——《MapReduce: Simplefied Data Processing on Large Clusters》，这篇论文论述了两个方面的内容，其中之一是 MapReduce 的编程模型，在后来的很多讨论中，人们对该模型褒贬不一，该编程模型在之后的技术发展中接受了大量的架构性改进，演变成了很多其他的编程模型，例如 DAG 模型等。当然，

MapReduce 模型本身作为一种基础模型得到了保留并依然运行在很多特定领域（比如，Hive 依然依赖 MapReduce 处理长时间的 ETL 业务）。MapReduce 在 GFS 的基础上再一次将大数据往前推进了一步，论文论述了如何在大量廉价机器的基础上稳定地实现超大规模的并行数据处理，这无疑是非常重要的进步。这篇论文无论在学术界还是在工业界都得到了极度狂热的追捧。原因无非是分布式计算系统可以套用于大量真实的业务场景，几乎任何一套单机计算系统都可以用 MapReduce 去改良。

2006 年，Google 发布了第三篇重要论文——《 BigTable: A Distributed Storage System for Structured Data 》，用于解决 Google 内部海量结构化数据的存储以及高效读写问题。与前两篇论文相比，这篇论文更难理解一些。这是因为严格意义上来讲，BigTable 属于分布式数据库领域，需要读者具备一定的数据库基础，而且论文中提到的数据模型（多维稀疏排序映射模型）对于习惯了关系型数据库的工程师来说确实不易理解。但从系统架构来看，BigTable 还是有很多 GFS 的影子，包括 Master-Slave 模式、数据分片等。

这三篇论文在大数据历史上，甚至整个 IT 界的发展历史上都具有革命性意义。但真正让大数据“飞入寻常百姓家”，是另一个科技巨头——Yahoo。Google 的三篇论文论证了在大量廉价机器上存储、处理海量数据（结构化数据、非结构化数据）是可行的，然而并没有给出开源方案。2004 年，Doug Cutting 和 Mike Cafarella 在为他们的搜索引擎爬虫（Nutch）实现分布式架构的时候看到了 Google 的 GFS 论文以及 MapReduce 论文。他们在之后的几个月里按照论文实现出一个简易版的 HDFS 和 MapReduce，这也就是 Hadoop 的最早起源。最初这个简易系统确实可以稳定地运行在几十台机器上，但是没有经过大规模使用的系统谈不上完美。所幸他们收到了 Yahoo 的橄榄枝。在 Yahoo，Doug 领导的团队不断地对系统进行改进，促成了 Hadoop 从几十台到几百台再到几千台机器规模的演变，直到这个时候，大数据才真正在普通公司实现落地。

至于 BigTable，没有在 Yahoo 内得到实现，原因不明。一家叫做 Powerset 的公司，为了高效处理自然语言搜索产生的海量数据实现了 BigTable 的开源版本——HBase，并在发展了 2 年之后被 Apache 收录为顶级项目，正式入驻 Hadoop 生态系统。HBase 成为 Apache 顶级项目之后发展非常迅速，各大公司纷纷开始使用 HBase，HBase 社区的高度活跃性让 HBase 这个系统发展得更有活力。有意思的是，Google 在将 BigTable 作为云服务对外开放的时候，决定提供兼容 HBase 的 API。可见在业界，HBase 已经一定程度上得到了广泛的认可和使用。

2. HBase 使用现状

HBase 在国外起步很早，包括 Facebook、Yahoo、Pinterest 等大公司都大规模使用 HBase 作为基础服务。在国内 HBase 相对起步较晚，但现在各大公司对于 HBase 的使用已经越来越普遍，包括阿里巴巴、小米、华为、网易、京东、滴滴、中国电信、中国人寿等公司都使用 HBase 存储海量数据，服务于各种在线系统以及离线分析系统，其中阿里巴

巴、小米以及京东更是有着数千台 HBase 的集群规模。业务场景包括订单系统、消息存储系统、用户画像、搜索推荐、安全风控以及物联网时序数据存储等。最近，阿里云、华为云等云提供商先后推出了 HBase 云服务，为国内更多公司低门槛地使用 HBase 服务提供了便利。

另外，相比其他技术社区，HBase 社区非常活跃，每天都会有大量的国内外工程师参与 HBase 系统的开发维护，大部分问题都能在社区得到快速积极的响应。近几年，HBase 社区中，国内开发者的影响力开始慢慢扩大，在某些功能领域甚至已经占据主导地位。

3.HBase 版本变迁

HBase 从 2010 年开始前前后后经历了几十个版本的升级，不断地对读写性能、系统可用性以及稳定性等方面进行改进，如图 1-1 所示。在这些版本中，有部分版本在 HBase 的发展历程中可谓功勋卓著。

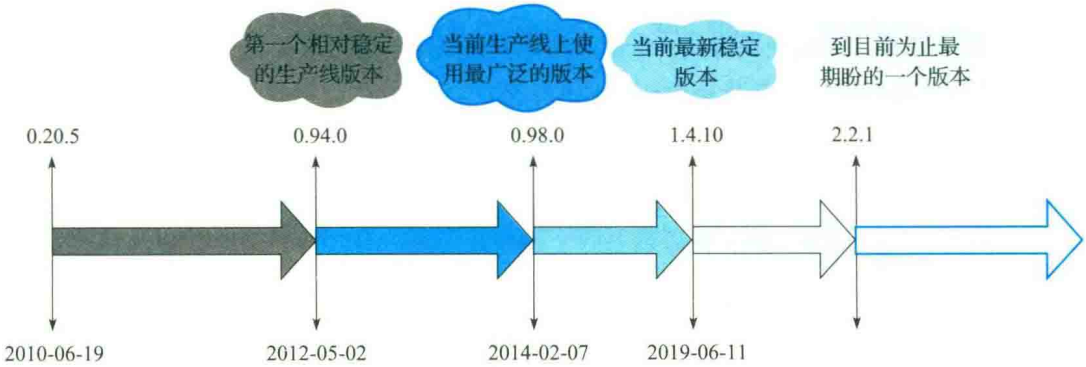


图 1-1　HBase 版本变迁

0.94.x 版本是 HBase 历史上第一个相对稳定的生产线版本，国内最早使用 HBase 的互联网公司（小米、阿里、网易等）都曾在生产线上大规模使用 0.94.x 作为服务版本，即使在当前，依然还有很多公司的业务运行在 0.94.x 版本，可见 0.94.x 版本在过去的几年时间里是多么辉煌。

之后的 2 年时间，官方在 0.94 版本之后发布了两个重要版本：0.96 版本和 0.98 版本，0.96 版本实现了很多重大的功能改进，比如 BucketCache、MSLAB、MTTR 优化等，但也因为功能太多而引入了很多 bug，导致生产线上真正投入使用的并不多。直至 0.98 版本发布。0.98 版本修复了大量的 bug，大大提升了系统可用性以及稳定性。不得不说，0.98 版本是目前业界公认的 HBase 历史上最稳定的版本之一，也是目前生产线上使用最广泛的版本之一。

2015 年 2 月，社区发布了 1.0.0 版本，这个版本带来的最大改变是规范了 HBase 的版本号，此后的版本号都统一遵循 semantic versioning 语义，如图 1-2 所示。

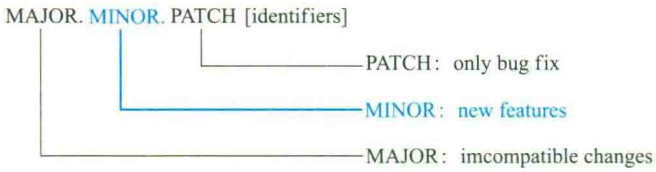


图 1-2 HBase 版本规则

比如 1.2.6 版本中 MAJOR 版本是 1，MINOR 版本是 2，PATCH 是 6。不同 MAJOR 版本不保证功能的兼容性，比如 2.x 版本不保证一定兼容 1.x 版本。MINOR 版本表示会新增新的功能，比如 1.2.x 会在 1.1.x 的基础上新增部分功能。而 PATCH 版本只负责修复 bug，因此可以理解为 MAJOR、MINOR 相同的情况下，PATCH 版本越大，系统越可靠。

在 1.0 的基础上官方先后发布了 1.1.x、1.2.x、1.3.x 以及 1.4.x 等多个版本。因为稳定性的原因，并不建议在生产线上使用 1.0.0 ～ 1.1.2 中间的版本。目前，HBase 社区推荐使用的稳定版本为 1.4.10。[⊖]

2.x 版本是接下来最受期待的一个版本（升级要慎重，请参考社区中的实践），因为最近一两年社区开发的新功能都将集中在 2.x 版本发布，2.x 包含的核心功能特别多，包括：大幅度减小 GC 影响的 offheap read path/write path 工作，极大提升系统稳定性的 Procedure V2 框架，支持多租户隔离的 RegionServer Group 功能，支持大对象存储的 MOB 功能等。

1.2 HBase 数据模型

从使用角度来看，HBase 包含了大量关系型数据库的基本概念——表、行、列，但在 BigTable 的论文中又称 HBase 为“sparse, distributed, persistent multidimensional sorted map”，即 HBase 本质来看是一个 Map。那 HBase 到底是一个什么样的数据库呢？

实际上，从逻辑视图来看，HBase 中的数据是以表形式进行组织的，而且和关系型数据库中的表一样，HBase 中的表也由行和列构成，因此 HBase 非常容易理解。但从物理视图来看，HBase 是一个 Map，由键值（KeyValue，KV）构成，不过与普通的 Map 不同，HBase 是一个稀疏的、分布式的、多维排序的 Map。接下来，笔者首先从逻辑视图层面对 HBase 中的基本概念进行介绍，接着从稀疏多维排序 Map 这个视角进行深入解析，最后从物理视图层面说明 HBase 中的数据如何存储。

1.2.1 逻辑视图

在具体了解逻辑视图之前有必要先看看 HBase 中的基本概念。

- table：表，一个表包含多行数据。
- row：行，一行数据包含一个唯一标识 rowkey、多个 column 以及对应的值。在

⊖ 随着 HBase 项目的不断更新迭代，社区推荐的稳定版也会不断更新。用户可以在这里看到当前 HBase 社区推荐使用的稳定版本：https://www.apache.org/dist/hbase/stable/。

HBase 中，一张表中所有 row 都按照 rowkey 的字典序由小到大排序。

- column：列，与关系型数据库中的列不同，HBase 中的 column 由 column family（列簇）以及 qualifier（列名）两部分组成，两者中间使用 ":" 相连。比如 contents:html，其中 contents 为列簇，html 为列簇下具体的一列。column family 在表创建的时候需要指定，用户不能随意增减。一个 column family 下可以设置任意多个 qualifier，因此可以理解为 HBase 中的列可以动态增加，理论上甚至可以扩展到上百万列。
- timestamp：时间戳，每个 cell 在写入 HBase 的时候都会默认分配一个时间戳作为该 cell 的版本，当然，用户也可以在写入的时候自带时间戳。HBase 支持多版本特性，即同一 rowkey、column 下可以有多个 value 存在，这些 value 使用 timestamp 作为版本号，版本越大，表示数据越新。
- cell：单元格，由五元组（row, column, timestamp, type, value）组成的结构，其中 type 表示 Put/Delete 这样的操作类型，timestamp 代表这个 cell 的版本。这个结构在数据库中实际是以 KV 结构存储的，其中（row, column, timestamp, type）是 K，value 字段对应 KV 结构的 V。

图 1-3 是 BigTable 中一张示例表的逻辑视图，表中主要存储网页信息。示例表中包含两行数据，两个 rowkey 分别为 com.cnn.www 和 com.example.www，按照字典序由小到大排列。每行数据有三个列簇，分别为 anchor、contents 以及 people，其中列簇 anchor 下有两列，分别为 cnnsi.com 以及 my.look.ca，其他两个列簇都仅有一列。可以看出，根据行 com.cnn.www 以及列 anchor:cnnsi.com 可以定位到数据 CNN，对应的时间戳信息是 t9。而同一行的另一列 contents:html 下却有三个版本的数据，版本号分别为 t5、t6 和 t7。

rowkey	anchor		contents	people
	cnnsi.com	my.look.ca	html	author
"com.cnn.www"	t9:CNN	t8:CNN.com	t7:"<html>..." t6:"<html>..." t5:"<html>..."	
"com.example.www"				t5:John Doe

图 1-3　HBase 逻辑视图

总体来看，HBase 的逻辑视图是比较容易理解的，需要注意的是，HBase 引入了列簇的概念，列簇下的列可以动态扩展；另外，HBase 使用时间戳实现了数据的多版本支持。

1.2.2　多维稀疏排序 Map

使用关系型数据库中表的概念来描述 HBase，对于 HBase 的入门使用大有裨益，然

而，对于理解 HBase 的工作原理意义不大。要真正理解 HBase 的工作原理，需要从 KV 数据库这个视角重新对其审视。BigTable 论文中称 BigTable 为 "sparse, distributed, persistent multidimensional sorted map"，可见 BigTable 本质上是一个 Map 结构数据库，HBase 亦然，也是由一系列 KV 构成的。然而 HBase 这个 Map 系统却并不简单，有很多限定词——稀疏的、分布式的、持久性的、多维的以及排序的。接下来，我们先对这个 Map 进行解析，这对于之后理解 HBase 的工作原理非常重要。

大家都知道 Map 由 key 和 value 组成，那组成 HBase Map 的 key 和 value 分别是什么？和普通 Map 的 KV 不同，HBase 中 Map 的 key 是一个复合键，由 rowkey、column family、qualifier、type 以及 timestamp 组成，value 即为 cell 的值。举个例子，上节逻辑视图中行 "com.cnn.www" 以及列 "anchor:cnnsi.com" 对应的数值 "CNN" 实际上在 HBase 中存储为如下 KV 结构：

```
{"com.cnn.www","anchor","cnnsi.com","put","t9"} -> "CNN"
```

同理，其他的 KV 还有：

```
{"com.cnn.www","anchor","my.look.ca","put","t8"} -> "CNN.com"
{"com.cnn.www","contents","html","put","t7"} -> "<html>..."
{"com.cnn.www","contents","html","put","t6"} -> "<html>..."
{"com.cnn.www","contents","html","put","t5"} -> "<html>..."
{"com.example.www","people","author","put","t5"} -> "John Doe"
```

至此，读者对 HBase 中数据的存储形式有了初步的了解，在此基础上再来介绍多维、稀疏、排序等关键词。

- 多维：这个特性比较容易理解。HBase 中的 Map 与普通 Map 最大的不同在于，key 是一个复合数据结构，由多维元素构成，包括 rowkey、column family、qualifier、type 以及 timestamp。
- 稀疏：稀疏性是 HBase 一个突出特点。从图 1-3 逻辑表中行 "com.example.www" 可以看出，整整一行仅有一列（people:author）有值，其他列都为空值。在其他数据库中，对于空值的处理一般都会填充 null，而对于 HBase，空值不需要任何填充。这个特性为什么重要？因为 HBase 的列在理论上是允许无限扩展的，对于成百万列的表来说，通常都会存在大量的空值，如果使用填充 null 的策略，势必会造成大量空间的浪费。因此稀疏性是 HBase 的列可以无限扩展的一个重要条件。
- 排序：构成 HBase 的 KV 在同一个文件中都是有序的，但规则并不是仅仅按照 rowkey 排序，而是按照 KV 中的 key 进行排序——先比较 rowkey，rowkey 小的排在前面；如果 rowkey 相同，再比较 column，即 column family:qualifier，column 小的排在前面；如果 column 还相同，再比较时间戳 timestamp，即版本信息，timestamp 大的排在前面。这样的多维元素排序规则对于提升 HBase 的读取性能至关重要，在后面读取章节会详细分析。