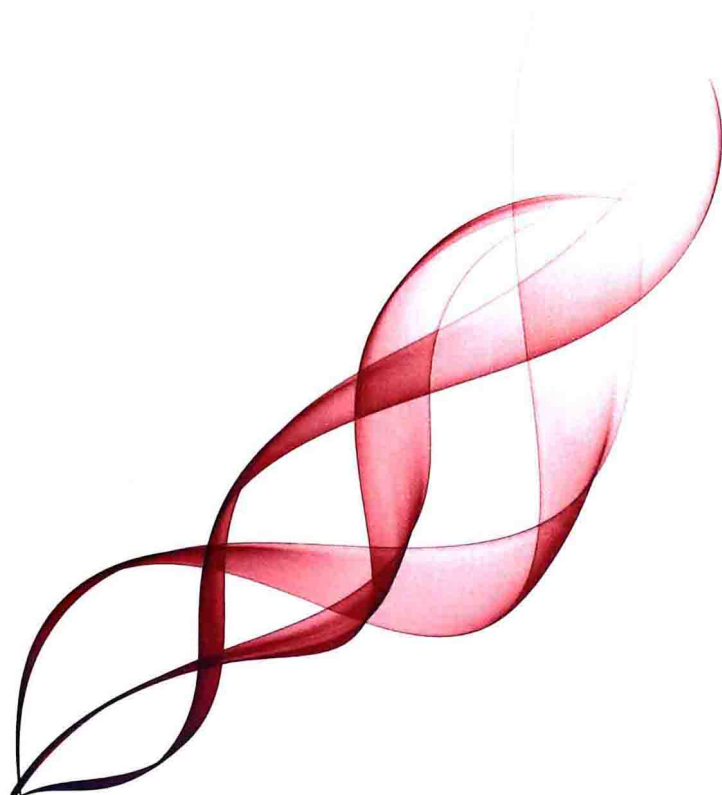




Enterprise Application Development with HBase

HBase企业应用 开发实战

马延辉 孟鑫 李立松◎著



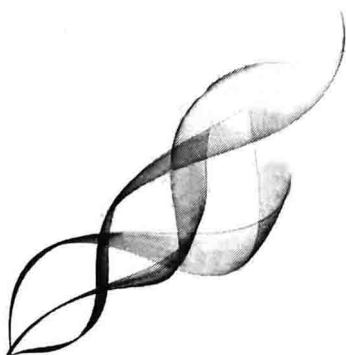
机械工业出版社
China Machine Press

技术丛书

Enterprise Application Development with HBase

HBase企业应用 开发实战

马延辉 孟鑫 李立松◎著



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

HBase 企业应用开发实战 / 马延辉, 孟鑫, 李立松著. —北京: 机械工业出版社, 2014.9
(大数据技术丛书)

ISBN 978-7-111-47832-4

I. H… II. ①马… ②孟… ③李… III. 计算机网络—信息存贮 IV. TP393.07

中国版本图书馆 CIP 数据核字 (2014) 第 204102 号



HBase 企业应用开发实战

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: 姜 影

责任校对: 殷 虹

印 刷: 北京市荣盛彩色印刷有限公司

版 次: 2014 年 9 月第 1 版第 1 次印刷

开 本: 186mm × 240mm 1/16

印 张: 30.25

书 号: ISBN 978-7-111-47832-4

定 价: 89.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzsj@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

为什么要写这本书

大数据是继云计算、物联网之后 IT 行业又一次颠覆性的技术革命。大数据在互联网、军事、金融、通信和物理学等领域已经有不少落地案例，而 Hadoop 技术的快速发展也引起业界广泛关注。可以说，现在 Hadoop 是大数据处理的关键技术，也是迄今为止，最成熟、应用最广泛的技术。

HBase 原型是 Google 的 BigTable 论文，从基因上讲，HBase 已经是 Hadoop 生态系统不可或缺的一部分。HBase 是完全开源的，同时存在多个版本，并且版本升级非常快，其学习成本比较高，学习周期比较长，所以现在大部分公司的工作人员很难在短时间内快速掌握并使用 HBase 框架。

此外，现在有关 HBase 的中文学习资料非常少，也给研发人员的学习带来了很大的难度。尽管现在市面上已经有几本关于 HBase 的中文书，但是，这几本书都是翻译作品，语言的组织不符合国人的习惯，并且实例讲解部分也不能切合本土国情。还有，这几本书分别侧重了某一个方面，如理论、实战、运维等，还没有一本书能够非常系统地阐述 HBase 框架。本书正是为了解决以上各种问题而编写的，也是国内第一本系统讲解 HBase 理论、实战和运维调优的书籍。

本书以 HBase 0.94 为基础，不仅深入探讨了 HBase 的原理架构和数据模型，更重要的是通过实际案例教会读者如何运用 HBase 框架来设计、搭建及运行大数据应用系统，同时结合生产案例剖析 HBase 系统运维和性能调优的技巧。

读者对象

本书适合以下读者阅读。

(1) 云计算、大数据处理技术和 NoSQL 数据库爱好者

“大数据”无疑是继“云计算”之后 IT 业界最热门的词汇。而云计算、大数据、NoSQL

技术本身存在交集，现在不少研究云计算的公司或机构都开始涉猎大数据和 NoSQL 领域，本书讲解的 HBase 数据库是 NoSQL 的一种，同时是大数据处理的关键技术，本书可以帮助这部分读者快速且全面地了解 HBase 的原理、架构、使用场景和细节知识点，理解 HBase 在云计算、大数据和 NoSQL 中的位置。

（2）对 Hadoop 及 HBase 感兴趣的开发人员

Hadoop 技术在近几年非常热，它已经是大数据处理的关键技术，而 HBase 作为 Hadoop 生态系统的重要组件，已经被越来越多的公司使用。本书详细介绍了 HBase 与 Hadoop 的关系、HBase 的基本概念、核心知识点和高级特性，并且结合实战案例讲解，使得读者可以快速掌握 HBase 的使用。

（3）使用 HBase 进行数据库开发或运维的高级 DBA

HBase 作为 NoSQL 数据库的一种，被越来越多的企业应用用作底层存储或者中间存储。本书不但讲解了 HBase 的原理和架构，更重要的是详细介绍了 HBase 的使用方法、运维监控和系统调优方法，能够帮助该部分读者快速掌握大型分布式数据库的安装、运维和调优技巧。

（4）开源软件爱好者

HBase 作为 Apache 基金会的顶级优秀开源项目，其实现过程中吸收了很多开源领域的优秀思想，同时也值得我们深入研究和学习。本书在讲解过程中剖析了不少 HBase 源代码，可以帮助该部分读者了解和掌握 HBase 框架源代码的设计方法和技巧。

（5）开设相关课程的高等院校学生

现在越来越多的高等院校已经开设了大数据方向的学生培养课程。在这些课程中，Hadoop 生态系统技术是核心课程，本书详细介绍 Hadoop 生态系统重要组件——HBase，这部分读者可以将本书作为参考教材使用。

如何阅读本书

本书分为三大部分。

第一部分为基础篇（第 1 ~ 5 章），介绍了大数据背景、HBase 基本原理、模式设计、HBase 的安装部署和所支持客户端 API 及使用方法。

第二部分为实战篇（第 6 ~ 8 章），通过三个典型的应用案例和代码示例，结合实践技巧和理论知识，深入讲解如何使用 HBase 设计大型数据应用系统。

第三部分为高级篇（第 9 ~ 12 章），重点介绍 HBase 的整体架构、高级特性、运维监控和性能调优等，并结合生产系统的性能优化和运维经验进行讲解，旨在提升读者的实际操作经验。

最后本书列出了三个附录供读者参考。

附录 A 为 HBase 框架所有配置参数的介绍。

附录 B 为基于 HBase 的 SQL 引擎工具 Phoenix 的 SQL 语法详解。

附录 C 为 HBase 性能测试工具 YCSB 编译安装介绍。

如果你是一名已经具备一定 Hadoop、HBase 基础知识和使用经验的用户，那么可以直接阅读第二部分和第三部分。第二部分侧重实战，第三部分侧重运维，请读者自行选择阅读。但是，如果你是一名初学者，请一定从第 1 章的基础理论知识开始学习。

勘误和支持

本书第 2、3 章由孟鑫书写，第 4 章由李立松书写，其他章由马延辉书写。由于作者的水平有限，编写时间仓促，书中难免会出现一些错误或者不准确的地方，恳请读者批评指正。读者可以将书中的错误发布在站点页面 <http://www.adintellig.com/hbase-in-action/> 的留言中。同时如果你遇到任何问题，也可以访问该页面，我们将尽量在线上为你提供最满意的解答。如果你有更多的宝贵意见，也欢迎发送邮件至邮箱 binma85@gmail.com，期待能够得到你们的真挚反馈。

致谢

首先要感谢 Google 公司，它开放了如此优秀的论文 BigTable——HBase 的原型。

感谢 Apache 基金会以及所有对开源软件和开源社区作出贡献的朋友。感谢一直以来帮助过我们的数盟社区和 EasyHadoop 社区，他们提供的平台让我们认识了很多圈子里的同行，交流中技术和思想的碰撞让我们受益匪浅。

感谢查礼老师百忙之中抽出时间为本书写推荐。

感谢对我们有过帮助的 Ted Yu、叶刚、史彪、向磊、魏兴宝、卢亿磊、赵修湘、史东杰、廖旻可、陈书美、傅杰、费伟、孙利兵等人。

感谢机械工业出版社华章公司的编辑杨福川、姜影和白宇，在这一年多的时间中始终支持我们的写作，他们的鼓励和帮助引导着我们顺利完成本书。

最后感谢我们的爸爸、妈妈、爷爷、奶奶，感谢他们的养育之恩，并时时刻刻为我们灌输着信心和力量！

谨以此书献给我们最亲爱的家人，以及众多热爱 HBase 的朋友们！

马延辉于中国北京

目 录 *Contents*

前 言

第一部分 基础篇

第 1 章 认识 HBase 2

1.1 理解大数据背景 2

1.1.1 什么是大数据 3

1.1.2 为何大数据至关重要 4

1.1.3 NoSQL 在大数据中扮演的角色 4

1.2 HBase 是什么 6

1.2.1 HBase 的发展历史 6

1.2.2 HBase 的发行版本 7

1.2.3 HBase 的特性 9

1.3 HBase 与 Hadoop 的关系 10

1.4 HBase 的核心功能模块 12

1.4.1 客户端 Client 12

1.4.2 协调服务组件 ZooKeeper 13

1.4.3 主节点 HMaster 13

1.4.4 Region 节点 HRegionServer 13

1.5 HBase 的使用场景和经典案例 14

1.5.1 搜索引擎应用 15

1.5.2 增量数据存储 15

1.5.3 用户内容服务 17

1.5.4 实时消息系统构建 18

1.6 本章小结 18

第 2 章 HBase 安装与配置 19

2.1 先决条件 19

2.2 HBase 运行模式 23

2.2.1 单机模式 23

2.2.2 分布式模式 24

2.3 HBase 的 Web UI 31

2.4 HBase Shell 工具使用 31

2.5 停止 HBase 集群 33

2.6 本章小结 33

第 3 章 数据模型 34

3.1 两类数据模型 34

3.1.1 逻辑模型 35

3.1.2 物理模型 35

3.2 数据模型的重要概念 36

3.2.1 表 36

3.2.2 行键 37

3.2.3	列族	38	4.4.1	实例 1: 动物分类	54
3.2.4	单元格	38	4.4.2	实例 2: 店铺与商品	56
3.3	数据模型的操作	38	4.4.3	实例 3: 网上商城用户消费 记录	57
3.3.1	读 Get	39	4.4.4	实例 4: 微博用户与粉丝	58
3.3.2	写 Put	39	4.5	本章小结	60
3.3.3	扫描 Scan	39			
3.3.4	删除 Delete	40			
3.4	数据模型的特殊属性	40	第 5 章 HBase 客户端	61	
3.4.1	版本	40	5.1	精通原生 Java 客户端	61
3.4.2	排序	42	5.1.1	客户端配置	62
3.4.3	列的元数据	42	5.1.2	创建表	69
3.4.4	连接查询	43	5.1.3	删除表	70
3.4.5	计数器	43	5.1.4	插入数据	70
3.4.6	原子操作	43	5.1.5	查询数据	72
3.4.7	事务特性 ACID	43	5.1.6	删除数据	76
3.4.8	行锁	45	5.1.7	过滤查询	77
3.4.9	自动分区	45	5.2	使用 HBase Shell 工具操作 HBase	79
3.5	CAP 原理与最终一致性	46	5.2.1	命令分类	79
3.6	本章小结	47	5.2.2	常规命令	80
			5.2.3	DDL 命令	81
第 4 章 HBase 表结构设计	48		5.2.4	DML 命令	82
4.1	模式创建	48	5.2.5	工具命令 Tools	86
4.2	Rowkey 设计	49	5.2.6	复制命令	87
4.3	列族定义	51	5.2.7	安全命令	87
4.3.1	可配置的数据块大小	51	5.3	使用 Thrift 客户端访问 HBase	88
4.3.2	数据块缓存	52	5.3.1	Thrift 与 Thrift2 区别	88
4.3.3	布隆过滤器	52	5.3.2	安装与部署 Thrift2	89
4.3.4	数据压缩	53	5.3.3	Python 使用案例	93
4.3.5	单元时间版本	53	5.4	通过 REST 客户端访问 HBase	95
4.3.6	生存时间	54	5.4.1	启动服务	95
4.4	模式设计实例	54	5.4.2	使用 REST 访问 example 表	96

5.5	使用 MapReduce 批量操作 HBase	97
5.5.1	三种访问模式	98
5.5.2	实现 MapReduce API	98
5.5.3	HBase 作为输入源示例	99
5.5.4	HBase 作为输出源示例	101
5.5.5	HBase 作为共享源示例	103
5.6	通过 Web UI 工具查看 HBase 状态	106
5.6.1	Master 状态界面	106
5.6.2	RegionServer 状态界面	107
5.6.3	ZooKeeper 统计信息页面	109
5.7	其他客户端	110
5.8	本章小结	110

第二部分 实战篇

第 6 章 整合 SQL 引擎层 114

6.1	NoSQL 背景知识	114
6.1.1	什么是 NoSQL	114
6.1.2	将 SQL 整合到 HBase 的原因	115
6.1.3	基于 HBase 的 SQL 引擎实现	116
6.2	Hive 整合 HBase 的实现	119
6.2.1	认识 Hive	119
6.2.2	Hive 整合 HBase 的环境准备	122
6.2.3	Linux 环境下重新编译 Hive	123
6.2.4	Hive 参数配置	125
6.2.5	启动 Hive	127
6.2.6	Hive 与 HBase 整合后的框架如何使用	127
6.2.7	HBase 到 Hive 的字段映射	133

6.2.8	多列与 Hive Map 类型	134
6.3	查询引擎 Phoenix	137
6.3.1	认识 Phoenix	138
6.3.2	Phoenix 安装环境准备	141
6.3.3	Phoenix 安装部署	142
6.3.4	Phoenix 源码编译	143
6.3.5	Phoenix 中 SQLLine 的快速使用	149
6.3.6	使用 JDBC 访问 Phoenix	153
6.4	对象映射框架 Kundera	155
6.4.1	认识 Kundera	155
6.4.2	Kundera 的客户端 API 快速使用	158
6.4.3	Kundera 模块介绍	161
6.4.4	Kundera 的 REST 访问方式	162
6.5	分布式 SQL 引擎 Lealone	165
6.5.1	认识 Lealone	165
6.5.2	Lealone 的安装部署	166
6.5.3	通过 JDBC 访问 Lealone	168
6.5.4	通过 Python 访问 Lealone	169
6.5.5	Lealone 特有的建表语法	170
6.6	本章小结	171

第 7 章 构建音乐站用户属性库 173

7.1	案例背景	173
7.1.1	音乐站	173
7.1.2	需求概述	175
7.1.3	需求范围和系统边界	175
7.1.4	需求详述	176
7.1.5	名词解释	180
7.2	概要设计	181

7.2.1	设计目标	181
7.2.2	数据规模假设	181
7.2.3	功能指标	182
7.2.4	系统流程	182
7.3	表结构设计	183
7.3.1	功能抽象	183
7.3.2	逻辑结构	184
7.3.3	Rowkey 设计	188
7.3.4	列族设计	188
7.3.5	版本定义	188
7.3.6	优化属性定义	188
7.4	数据加载	189
7.4.1	加载流程	189
7.4.2	Mapper 类	190
7.4.3	Main 类	192
7.4.4	运行	193
7.5	数据检索	193
7.5.1	HBaseTable	193
7.5.2	HBaseAdmin	193
7.5.3	几种检索类型	195
7.6	后台查询	198
7.6.1	二级索引实现	198
7.6.2	后台查询系统	205
7.7	本章小结	206

第 8 章 构建广告实时计算系统 208

8.1	理解广告数据和流处理框架	208
8.1.1	网络广告的几大特性	209
8.1.2	网络广告的数据类型	210
8.1.3	流处理框架	211
8.1.4	背景与需求描述	217

8.2	概要设计	218
8.2.1	设计目标	219
8.2.2	主要功能	219
8.2.3	系统架构	219
8.3	详细设计	221
8.3.1	表结构设计	221
8.3.2	功能模块设计	222
8.4	核心功能实现	223
8.4.1	规划集群环境部署	223
8.4.2	安装 ZooKeeper 集群	225
8.4.3	安装 Kafka 分布式集群	228
8.4.4	实现 Kafka 生产者	231
8.4.5	安装 Storm 分布式集群	233
8.4.6	查看集群节点部署情况	240
8.4.7	基于 Storm-kafka 中间件 实现计算逻辑	240
8.4.8	如何使用 HBase 中统计数据	251
8.5	本章小结	252

第三部分 高级篇

第 9 章 核心概念 254

9.1	核心结构	254
9.1.1	B+ 树	255
9.1.2	LSM 树	255
9.1.3	两种结构本质区别	257
9.2	底层持久化	258
9.2.1	存储基本架构	258
9.2.2	HDFS 文件	259
9.2.3	Region 切分	264

9.2.4	合并	265	10.1.7	Thrift 使用过滤器	304
9.2.5	HFile 格式	266	10.1.8	过滤器总结	309
9.2.6	KeyValue 格式	269	10.2	计数器	310
9.3	预写日志	270	10.2.1	使用 Shell 操作计数器	310
9.3.1	概要流程	270	10.2.2	基于单列的计数器	312
9.3.2	相关 Java 类	271	10.2.3	多列计数器	313
9.3.3	日志回放	274	10.3	协处理器	314
9.3.4	日志一致性	275	10.3.1	认识协处理器	315
9.4	写入流程	276	10.3.2	观察者 Observer	316
9.4.1	客户端	276	10.3.3	终端 EndPoint	318
9.4.2	服务器端	281	10.3.4	协处理器部署	320
9.5	查询流程	286	10.4	Schema 设计要点	323
9.5.1	两种查询操作	286	10.4.1	行键设计	323
9.5.2	客户端	286	10.4.2	列族设计	325
9.5.3	服务器端	287	10.5	二级索引	325
9.6	数据备份	291	10.5.1	Client-managed 方式	326
9.6.1	备份机制架构	292	10.5.2	ITHBase 实现	326
9.6.2	故障恢复	292	10.5.3	IHBase 实现	329
9.7	数据压缩	294	10.5.4	Coprocessor 方式	329
9.7.1	支持的压缩算法	295	10.5.5	MapReduce 两种方式	330
9.7.2	使用配置	295	10.6	布隆过滤器	330
9.8	本章小结	296	10.6.1	基本概念	331
10.6.2	配置布隆过滤器	332	10.6.3	使用布隆过滤器	333
第 10 章 HBase 高级特性	297	10.7	负载均衡	333	
10.1	过滤器	297	10.7.1	全局计划	334
10.1.1	过滤器的两类参数	297	10.7.2	随机分配计划	337
10.1.2	比较器	298	10.7.3	批量启动分配计划	337
10.1.3	列值过滤器	300	10.7.4	通过 Shell 控制负载均衡	338
10.1.4	键值元数据过滤器	300	10.8	批量加载	338
10.1.5	行键过滤器	303	10.8.1	准备数据: importtsv	338
10.1.6	功能过滤器	303			

10.8.2 加载数据: completebulkload	340	11.5.2 基于 JMX 的监控工具 Ella	368
10.9 本章小结	340	11.6 报警工具 Nagios	371
第 11 章 集群运维管理	341	11.7 故障处理	376
11.1 HBase 常用工具	341	11.7.1 问题咨询渠道	377
11.1.1 文件检测修复工具 hbck	342	11.7.2 常用日志信息	377
11.1.2 文件查看工具 hfile	346	11.7.3 常用故障调试工具	379
11.1.3 WAL 日志查看工具 hlog	348	11.7.4 客户端故障排查	384
11.1.4 压缩测试工具 CompressionTest	349	11.7.5 MapReduce 故障排查	386
11.1.5 数据迁移工具 CopyTable	350	11.7.6 网络故障排查	387
11.1.6 导出工具 export	351	11.7.7 RegionServer 相关问题 解决	387
11.1.7 导入工具 Import	351	11.7.8 Master 相关问题解决	391
11.1.8 日志回放工具 WALPlayer	351	11.7.9 ZooKeeper 相关问题解决	392
11.1.9 行数统计工具 RowCounter	352	11.8 集群备份	392
11.2 Region 和 RegionServer 管理	353	11.8.1 冷备份	393
11.2.1 大合并工具 major_compact	353	11.8.2 热备份之 Replication	393
11.2.2 Region 合并工具 Merge	354	11.8.3 热备份之 CopyTable	393
11.2.3 下线节点	354	11.8.4 热备份之 Export	393
11.2.4 滚动重启	355	11.9 本章小结	393
11.3 性能指标 Metrics	356	第 12 章 性能调优	395
11.3.1 Master Metrics	357	12.1 硬件和操作系统调优	395
11.3.2 RegionServer Metrics	357	12.1.1 配置内存	395
11.3.3 RPC Metrics	358	12.1.2 配置 CPU	396
11.3.4 JVM Metrics	359	12.1.3 操作系统	396
11.3.5 集群属性 Metrics	360	12.2 网络通信调优	399
11.4 监控系统 Ganglia	360	12.2.1 配置交换机	399
11.4.1 HBase 监控指标	360	12.2.2 添加机架感知	401
11.4.2 安装、部署和使用 Ganglia	361	12.3 JVM 优化	402
11.5 HBase 管理扩展 JMX	366	12.3.1 Java 垃圾回收算法	402
11.5.1 如何使用 JMX	366	12.3.2 Java 垃圾收集器	403

12.3.3	垃圾回收器的选择	405	12.7.2	调整 BlockCache 大小	425
12.3.4	JVM 参数设置	406	12.7.3	设置 MemStore 的上下限	426
12.4	HBase 查询优化	408	12.7.4	调整影响合并的文件数	427
12.4.1	设置 Scan 缓存	408	12.7.5	调整 MemStore 的 flush 因子	427
12.4.2	显式地指定列	409	12.7.6	调整单个文件大小	427
12.4.3	关闭 ResultScanner	410	12.7.7	调整 ZooKeeper Session 的 有效时长	428
12.4.4	禁用块缓存	410	12.8	分布式协调系统 ZooKeeper 优化	428
12.4.5	优化行键查询	410	12.8.1	配置 ZooKeeper 节点数	428
12.4.6	通过 HTableTool 访问	410	12.8.2	独立 ZooKeeper 集群	429
12.4.7	使用批量读	411	12.9	表设计优化	430
12.4.8	使用 Filter 降低客户端 压力	412	12.9.1	开启布隆过滤器	430
12.4.9	使用 Coprocessor 统计 行数	412	12.9.2	调整列族块大小	430
12.4.10	缓存查询结果	413	12.9.3	设置 In Memory 属性	432
12.5	HBase 写入优化	413	12.9.4	调整列族最大版本数	434
12.5.1	关闭写 WAL 日志	413	12.9.5	设置 TTL 属性	435
12.5.2	设置 AutoFlush	414	12.10	其他优化	436
12.5.3	预创建 Region	415	12.10.1	关闭 MapReduce 的预测 执行功能	436
12.5.4	延迟日志 flush	419	12.10.2	修改负载均衡执行周期	438
12.5.5	通过 HTableTool 访问	419	12.11	性能测试	438
12.5.6	使用批量写	420	12.12	本章小结	441
12.6	HBase 基本核心服务优化	421	附录 A	HBase 配置参数介绍	442
12.6.1	优化分裂操作	421	附录 B	Phoenix SQL 语法详解	451
12.6.2	优化合并操作	423	附录 C	YCSB 编译安装	468
12.7	HBase 配置参数优化	423			
12.7.1	设置 RegionServer Handler 数量	423			

第一部分 *Part 1*

基础篇

- 第1章 认识 HBase
- 第2章 HBase 安装与配置
- 第3章 数据模型
- 第4章 HBase 表结构设计
- 第5章 HBase 客户端



认识 HBase

本章将介绍大数据背景和 HBase 的基本概念，从大数据引申到 NoSQL，并阐述 HBase 出现的契机。随后，将介绍 HBase 的概念、发展历史、发行版本和基本特性。其中，HBase 的核心功能模块将作为一个小节单独重点介绍，最后通过介绍 HBase 的使用场景和经典案例，让读者朋友能够清晰地了解 HBase 可以做什么。

作为 NoSQL 家庭的一员，HBase 的出现弥补了 Hadoop 只能离线批处理的不足，同时能够存储小文件，提供海量数据的随机检索，并保证一定的性能。而这些特性也完善了整个 Hadoop 生态系统，泛化其大数据的处理能力，结合其高性能、稳定、扩展性好的特行，给使用大数据的企业带来了福音。

因为本章是全书的开篇，唯有简明扼要地介绍才能帮助正在学习和想要学习 HBase 的读者，所以本章将提纲挈领地介绍 HBase 的相关知识，重点介绍 HBase 是什么以及 HBase 能做什么两部分。

1.1 理解大数据背景

经美国权威机构 IDC 调查发现，现如今的公司正在以前所未有的速度和丰富的类型产生数据，并且也有能力存储这些数据，但是，如何关联这两方面以便产生最大的商业价值，是所有公司共同面临的挑战。这个问题非常复杂：虽然业务人员在技能提升和专业工具的帮助下，越来越了解数据，但由于数据的增长速度越来越快，积累量级越来越大，公司可以利用的数据比例正在迅速下降。

1.1.1 什么是大数据

Gartner 认为与过去相关概念相比,大数据强调 3V 特征,即 Volume(量级)、Variety(种类)和 Velocity(速度),如图 1-1 所示。

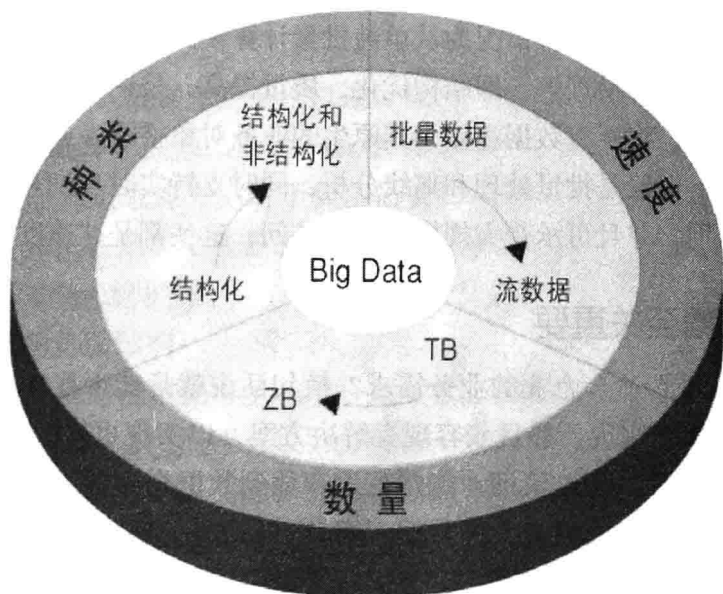


图 1-1 大数据三大特性

如今存储的数据量正在急剧增长,2000 年全球存储了 EB 级别的数据,预计到 2020 年,该值将变为 ZB 级别。仅 Twitter 每天就会生成超过 10TB 的数据,Facebook 的数据为几十 TB,一些特殊的企业在每小时就会产生 TB 级别的数据。

上面这些企业是一些典型的案例,其实我们生活的方方面面都会形成很多“轨迹”。例如,打开手机会生成一个事件;乘坐公共交通刷卡,这是一个事件;检票登机、打卡上班、App Store 上购买应用、更换电视频道、使用高速路电子收费系统等。每一项操作都会生成数据,并且该数据的量级与参与的人数相关,全球 60 亿人口,如果仅仅 1/10 的人参与进来,那么这个数据量级就已经非常惊人。就在 10 年前 IT 界超过 1TB 的数据仓库屈指可数,而现在则是“举不胜举”。

随着传感器、智能设备以及社交协作技术的激增,企业中的数据也变得更加复杂,因为它不仅包含传统的关系型数据,还包含来自网页、Web 日志文件、社交媒体论坛、电子邮件、文档、传感器数据等原始、半结构化和非结构化数据。

传统系统可能很难存储、分析这些数据的内容,更不要说挖掘有价值的信息。因为传统的数据库、数据仓库、联机事务处理等技术并不适合处理这些数据。尽管一些公司正在朝大数据方向大力发展,但总体而言,大部分公司只是刚开始理解大数据。当回首整个数据库发展的历程会发现,人们将大部分时间都花在仅 20% 的数据上:这些数据格式整齐且符合严格模式的关系类型。但事实是,全球 80% 的数据是非结构化的或者半结构化的。

视频和图片不能轻松或高效地存储在关系型数据库中，某些事件信息可能动态地更改（如气象），它们不太适合严格的模式。要利用大数据，企业必须能够分析所有类型的数据，包括关系和非关系数据：文本、传感器数据、音频和视频等。

有效处理大数据需要在数据变化的过程中对它的数量和种类进行分析，而不只是在“静止”状态进行分析。业界定义这种情况为从单纯批量计算模式到实时动态计算模式的内涵式转变。内涵式在这里也比较好理解，即结构优化、质量提高，是一种实现实质性的跨越式的进程。大数据平台允许用户将所有数据存储为其原生的业务对象格式，通过可用组件上的大规模并行计算实现价值，不仅仅是批量处理和离线分析，同时支持实时查询和处理等特征，甚至要求响应时间在毫秒级别，并且可承受大规模的并发访问，这些都是“速度”特征的范畴。

1.1.2 为何大数据至关重要

这种非传统分析是否适合企业的业务需求？换句话说就是能否找到一个大数据平台可为当前的分析工具提供补充实现，并且兼容现有解决方案，以实现更好的业务成果。

通常情况下，数据必须经过清理才能规范地存放到数据仓库中。相反大数据解决方案不仅会利用不适合传统仓库且数量庞大的数据，而且不需要改变原有数据格式，保留了数据的真实性，并能够快速访问海量的信息。对于不能使用传统关系型数据库方法处理的信息所带来的挑战，大数据解决方案非常适合。大数据之所以重要，是因为其具备解决现实问题的三个关键方面。

- ❑ 分析各种不同来源的结构化、结构化和非结构化数据的理想选择。
- ❑ 当需要分析所有或大部分数据，或者对一个数据抽样分析效果不明显时，大数据解决方案是理想的选择。
- ❑ 未预先确定数据的业务度量指标时，是进行迭代式和探索式分析的理想选择。

1.1.3 NoSQL 在大数据中扮演的角色

NoSQL，是 Not only SQL 的缩写，泛指非关系型的数据库。与关系型数据库相比，NoSQL 存在许多显著的不同点，其中最重要的是 NoSQL 不使用 SQL 作为查询语言。其数据存储可以不需要固定的表模式，也通常会避免使用 SQL 的 JOIN 操作，一般又都具备水平可扩展的特性。NoSQL 的实现具有两个特征：使用硬盘和把随机存储器作存储载体。

1. 传统关系型数据库的缺陷

随着互联网 Web 2.0 的兴起，传统的关系数据库在应付 Web 2.0 网站，特别是超大规模和高并发的 SNS 类型动态网站时已经力不从心，暴露了很多难以克服的问题。

（1）高并发读写的瓶颈

Web 2.0 网站要根据用户个性化信息来实时生成动态页面和提供动态信息，所以基本上无法使用静态化技术，因此数据库并发负载非常高，可能峰值会达到每秒上万次读写请求。