华章科技

世界级软件开发大师和软件开发“教父”Martin Fowler与Jolt生产效率大奖图书作者Pramod J. Sadalage最新力作，权威性毋庸置疑

全方位比较关系型数据库与NoSQL数据库的异同，详细讲解4大主流NoSQL数据库的优劣势、用法和适用场合，深入探讨实现NoSQL数据库系统的各种细节

华章程序员书库

PEARSON

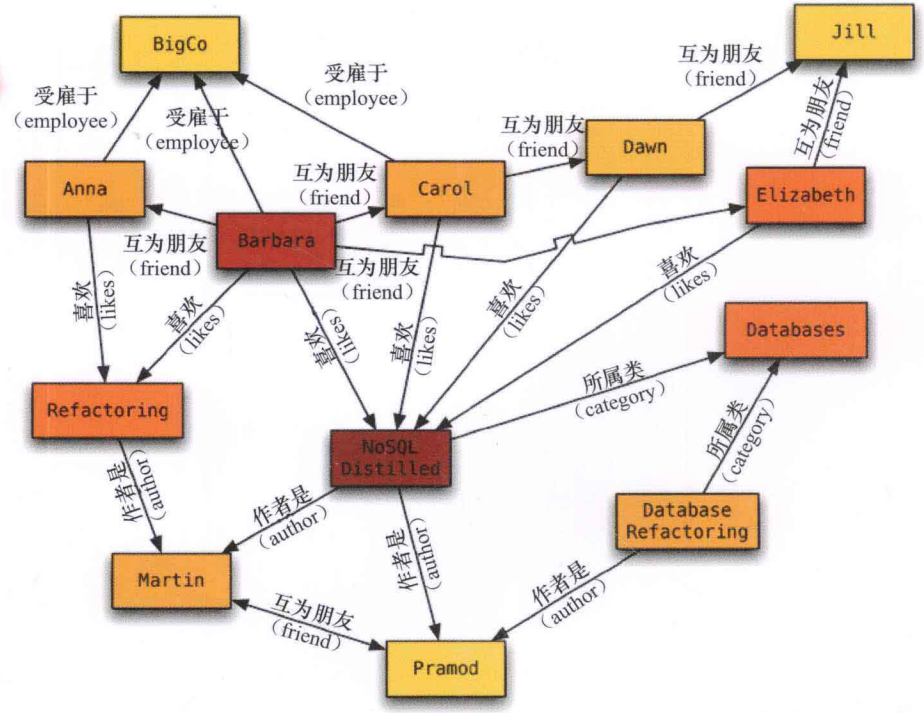
NoSQL Distilled

A Brief Guide to the Emerging World of Polyglot Persistence

NoSQL精粹

（美） Pramod J. Sadalage Martin Fowler 著

爱飞翔 译



机械工业出版社
China Machine Press

华章程序员书库

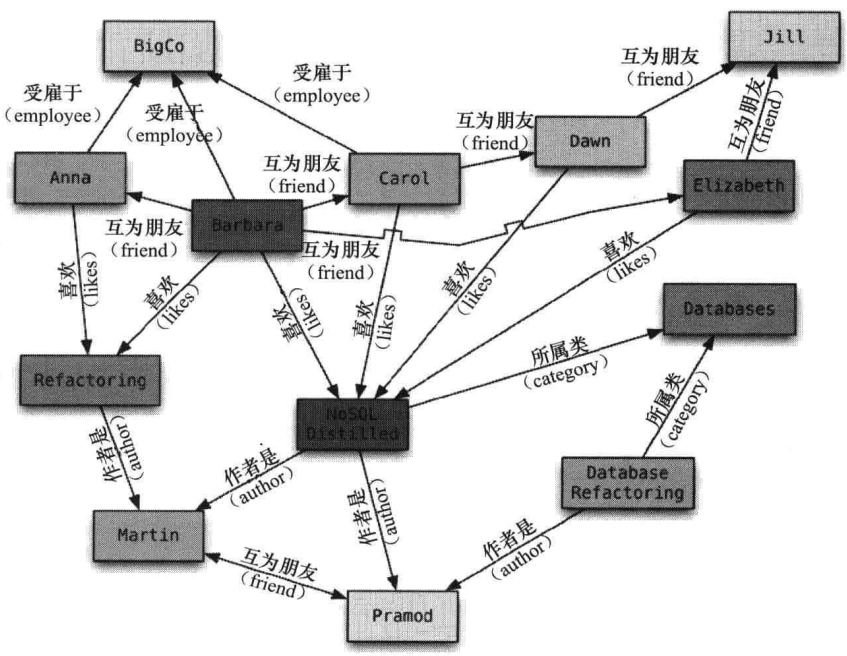
NoSQL Distilled

A Brief Guide to the Emerging World of Polyglot Persistence

NoSQL精粹

（美） Pramod J. Sadalage　Martin Fowler　著

爱飞翔　译



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

NoSQL 精粹 /（美）塞得拉吉（Sadalage，P. J.），（美）福勒（Fowler，M.）著；爱飞翔译 . —北京：机械工业出版社，2013.9
（华章程序员书库）
书名原文：NoSQL Distilled：A Brief Guide to the Emerging World of Polyglot Persistence

ISBN 978-7-111-43303-3

I. N… II. ①塞… ②福… ③爱… III. 数据库－系统 IV. TP311.138

中国版本图书馆 CIP 数据核字（2013）第 161584 号

本书为考虑是否可以使用和如何使用 NoSQL 数据库的企业提供了可靠的决策依据。它由世界级软件开发大师和软件开发“教父”Martin Fowler 与 Jolt 生产效率大奖图书作者 Pramod J. Sadalage 共同撰写。书中全方位比较了关系型数据库与 NoSQL 数据库的异同；分别以 Riak、MongoDB、Cassandra 和 Neo4J 为代表，详细讲解了键值数据库、文档数据库、列族数据库和图数据库这 4 大类 NoSQL 数据库的优劣势、用法和适用场合；深入探讨了实现 NoSQL 数据库系统的各种细节，以及与关系型数据库的混用。

全书分为两部分，共 15 章：第一部分（第 1 ～ 7 章）主要讲述 NoSQL 的核心概念。其中第 1 章解释了 NoSQL 发展迅速的原因；第 2 章描述了在 NoSQL 领域的三种主要的数据模型中如何体现“聚合”这一概念；第 3 章介绍了聚合的缺点；第 4 章描述了数据库如何在集群中分布数据；第 5 章论及了更新与读取操作对一致性的影响；第 6 章讨论了版本戳；第 7 章描述了适合用在 NoSQL 系统中的“映射 – 化简”操作。第二部分（第 8 ～ 15 章）讲述了如何实现 NoSQL 数据库系统。其中第 8 章～第 11 章每章各以一种 NoSQL 数据库为例，演示了如何实现第一部分介绍的概念；第 12 章解释了数据如何在强模式系统与无模式系统之间迁移；第 13 章着眼于混合持久化领域的趋势；第 14 章探讨了在混合持久化领域中会考虑到的其他一些技术；第 15 章提供了选择数据库时可以参考的一些建议。

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：关 敏
三河市杨庄长鸣印刷装订厂印刷
2013 年 9 月第 1 版第 1 次印刷
186mm×240mm · 11 印张
标准书号：ISBN 978-7-111-43303-3
定 价：49.00 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：hzjsj@hzbook.com

译 者 序

数据库技术是企业级应用程序经常会用到的，在习惯于传统的关系型数据库多年之后，一群先驱开始探索新的解决方案。随着待处理的数据量逐渐增多，大家越来越需要一种在集群环境中易于编程且执行效率高的大数据处理技术，在此情势下，NoSQL 数据库应运而生。

新技术诞生后，我们应该以既稳健又前瞻的心态看待它。一方面不宜在尚未充分理解时就盲目跟风，另一方面却也要紧密关注业界动向，顺应发展趋势。本书正是这样一本具有指导意义的手册，它虽篇幅短小，内涵却非常丰富。

书中首先分析了传统关系型数据库所要解决的问题，以及在解决手段方面存在的待改进之处。然后引入 NoSQL 这一新概念，并从数据模型、分布模型、一致性、版本戳、映射 - 化简操作等角度逐一详细比较了它与关系型数据库的异同。读者可以领略到 NoSQL 传承并发展了关系型数据库的哪些优秀特性，放弃了哪些不适合的特性，又新增了哪些内容，同时还将了解到产生这些异同的原因。

以上就是本书的第一部分。读者在理解了上述概念后，会在第二部分看到同为 NoSQL 数据库的四种子类型之间的关系。本书分别以 Riak、MongoDB、Cassandra 和 Neo4J 为代表，举例讲解了键值数据库、文档数据库、列族数据库和图数据库这四大类 NoSQL 数据库的用法，分别说明了其优势与劣势。

其后，书中又系统地讲解了关系型数据库与 NoSQL 数据库的数据模式迁移问题，描述了混合持久化这一新兴领域的样貌，并简述了此领域还将用到的一系列新技术。

通过分析与对比，我们可以发现，关系型数据库与 NoSQL 数据库并不矛盾，它们是从两个不同的角度来解决数据存储问题。在混合持久化的新环境下，二者互为补充，相辅相成，若运用得当，其整体效果将好于单用一门技术。

本书最后得出结论：鉴于 NoSQL 技术尚未成熟，所以大部分企业级应用程序开发者目前还应以现有关系型数据库为主，但是在前瞻性项目中可以先试先行；与此同时，

不论是否打算开始运用 NoSQL 数据库，都应该将传统项目中与数据库相关的代码抽离，便于将来 NoSQL 技术成熟后迅速切换。

通过阅读本书，我们可以掌握数据库技术的新动向，将现有的数据库技术细节从应用程序业务中提取出来，把它按数据用法分别封装到各个模块里，然后，根据书中所讲的知识，通过实践找出项目中适合改用 NoSQL 数据库的地方，并在 NoSQL 日臻成熟的过程中逐渐迁移过去。

如果我们能在 NoSQL 技术发展之初就把握其脉动，积极尝试并及时总结经验，那么待其成熟时，就能在混合持久化领域占得先机了，这也正是本书的一大意义。

由于本书作者乃业界巨擘，其观点影响颇广，为求谨慎，译文写出了很多中文术语的英文原名，以便读者查对，个别容易引发误解的称谓，加引号以强调其特殊含义。

在翻译过程中，得到了机械工业出版社华章公司诸位编辑与工作人员的帮助，在此深表谢意。同时还要感谢乔晓萌女士对我翻译工作的支持和鼓励。

本书由爱飞翔翻译，舒亚林与张军也参与了部分翻译工作。由于时间仓促，译者水平有限，错误与疏漏之处敬请读者批评指正。若您对本书有意见和建议，请通过电子邮件 eastarstormlee@gmail.com 联系译者，或访问网址 http://agilemobidev.com/eastarlee/book/nosql_distilled/ 留言。

爱飞翔

2013 年 6 月 15 日

前　言

我们已经在企业级计算领域研究了 20 余年，编程语言、架构、平台、软件开发流程等技术都在改变，然而这期间有一件事却一直没变，那就是：大家依然使用关系型数据库来存储数据。虽说也出现了一些挑战关系型数据库的产品，而且有的还在某些领域成功了，但是总体来说，留给架构师的数据存储问题仍然是选择使用哪款关系型数据库的问题。

稳定性在此领域颇受重视。企业的数据比程序存储的时间要长很多（至少大家都是这么说的。当然啦，我们也见过许多非常老的程序）。拥有一个既稳定，又容易理解，而且还能让许多应用程序编程平台访问的数据库，是非常有价值的。

不过，关系型数据库现在碰上新对手了，它的名字叫 NoSQL。由于我们需要处理的数据量越来越大，必须以商用服务器集群来构建大型硬件平台，因此 NoSQL 就应运而生了。这也使大家要再次考虑那个存在已久的难题，即代码如何才能同关系型数据库良好地结合起来。

“NoSQL”这个词的定义是非常不明确的。它泛指那些最近诞生的非关系型数据库，诸如 Cassandra、MongoDB、Neo4J 和 Riak 等。它们主张使用无模式（schemaless）[⊖]的数据，可以运行在集群环境中，并且能够牺牲传统数据库所具备的一致性，以换取另外一些有用的特性。NoSQL 的倡导者声称，使用它们可以构建出性能更高、扩展度更好且更易编程的系统。

这会不会敲响了关系型数据库即将灭亡的第一声警钟呢？还是说 NoSQL 要抢走数据库领域的头把交椅？我们的回答是：“这两种情况都不会出现。”关系型数据库是一个非常强大的工具，我们希望能长时间使用下去；然而大家也要看到一场深远的变革，那就是：关系型数据库不再是唯一的选择了。我们认为，数据库领域正进入混合持久

⊖ schema 也译作“纲要”、“大纲”、“概要”等，本书统称其为模式，与表示“处理技术问题的固定范式”之“模式”（Pattern）一词不同。——译者注

化（Polyglot Persistence）时代，由企业乃至个人研发的应用程序，可以使用多种技术来管理数据。因此架构师需要熟悉这些技术，并且能根据不同的需求做出适当的选择。若非如此，笔者怎会花那么多时间和精力来写这本书呢？

本书给诸位读者提供足够多的信息，协助大家在以后的研发过程中思考：项目是否真的值得使用 NoSQL 数据库。每个项目都是不同的，我们不可能写出一个简单的决策树，用它来选出合适的数据存储方式。与之相反，本书力求讲解大量的背景知识，以便大家了解 NoSQL 的工作原理，这样的话，你不用在互联网上四处寻找，就能够做出适合自己项目的决定了。笔者刻意将本书写得很短，以便读者能够快速阅览它。虽说本书不会回答各种具体问题，但是，它可以帮你缩小考虑的范围，让你明白自己当前应该提出哪些问题。

NoSQL 数据库为何引人关注

我们来看一下大家选用 NoSQL 数据库的两个主要原因。

- 应用程序的开发效率。在很多应用程序的开发过程中，大量精力和时间都放在了内存（in-memory）数据结构和关系型数据库之间的映射上面。NoSQL 数据库可以提供一种更加符合应用程序需求的数据模型，从而简化了数据交互，减少了所需编写、调试并修改的代码量。
- 大规模的数据。企业所重视的是，数据库要能够快速获取并处理数据。他们发现，即便关系型数据库能达成这一目标，其成本也很高。主要原因在于，关系型数据库是为独立运行的计算机而设计的，但是现在大家通常使用由更小、更廉价的计算机所组成的集群来计算数据，这样更实惠些。许多 NoSQL 数据库正是为集群环境而设计，因此它们更适合大数据量的应用场景。

本书内容

本书分为两个部分。第一部分主要讲述核心概念，让读者能够判断出 NoSQL 数据库是否适合自己，并且了解各种 NoSQL 数据库之间的差别。第二部分更加专注于实现 NoSQL 数据库系统。

第 1 章解释了 NoSQL 发展如此迅速的原因：由于需要处理的数据量越来越多，所以大型系统的扩展方式，由原来在单一计算机上的纵向扩展，转变为在计算机集群上

的横向扩展。这也印证了许多 NoSQL 数据库的数据模型所具备的一个重要特性，那就是：可以把内容密切相关的数据组织成一种丰富的结构，并将其显式存储起来，以便作为一个单元（unit）来访问。本书中，我们将这种类型的结构称为聚合（aggregate）。

第 2 章描述了在 NoSQL 领域的三种主要数据模型中，如何体现“聚合”这一概念。这三种数据库模型是：“键值模型”（key-value，参见 2.2 节），“文档模型”（document，参见 2.2 节）和“列族模型”（column family，参见 2.3 节）。聚合为许多种应用提供了一个自然的交互单元，既改善了集群的运行状况，又使编写程序来访问数据库变得更为容易。第 3 章转到聚合的缺点上面：难以处理位于不同聚合的实体之间的关系（参见 3.1 节）。这自然就引出了图数据库（参见 3.2 节），它是一个不属于面向聚合（aggregated-oriented）阵营的 NoSQL 数据模型。我们也会讲到 NoSQL 数据库的共同特性：它们都是以“无模式”的形式来操作的（参见 3.3 节）。模式的这种特性确实提供了更大的灵活性，但是它并不像大家想象的那么万能。

在讲完 NoSQL 数据模型方面的内容之后，我们接下来要讲分布模型。第 4 章描述了数据库如何在集群中分布数据。这个问题又细分为“分片”（sharding，参见 4.2 节）和“复制”（replication），复制方式可以是“主从复制”（master-slave replication，参见 4.3 节）或者“对等复制”（peer-to-peer replication，参见 4.4 节）。了解完分布模型的概念后，接下来要讲“一致性”（consistency）问题。与关系型数据库相比，NoSQL 数据库在一致性方面提供了更多选择，这么做是因为 NoSQL 要更好地支持集群。于是，第 5 章谈到了更新与读取操作对一致性的影响（分别参见 5.1 节和 5.2 节），如何在一致性与持久性之间进行仲裁（参见 5.5 节），以及如何放宽对持久性的约束以提升其他特性（参见 5.4 节）。如果之前听过 NoSQL，那么就应该听过“CAP 定理”（The CAP Theorem）。5.3 节中介绍了 CAP 定理的相关知识，告诉大家如何根据该理论来权衡一致性与其他特性。

前面这些章节主要侧重于如何分布数据并保持其一致性，接下来的两章讨论了要完成这项工作所需的一些重要工具。第 6 章讲述了版本戳（version stamp），它用来记录数据库的内容变更，并且可以检测数据是否一致。第 7 章概述了“映射 – 化简”（Map-Reduce）操作，这种计算方式很适合在集群中组织并行计算，因而也适用于 NoSQL 系统。

讲完这些概念后，我们针对以下 4 种数据库各举一些例子，来演示如何实现上述

概念。第 8 章使用 Riak 来演示“键值数据库”，第 9 章使用 MongoDB 作为“文档数据库”的示例，第 10 章选用 Cassandra 来探讨“列族数据库”，第 11 章选择了 Neo4J 作为“图数据库”的示例。此处必须强调：要想全面学习数据库，只依靠这些章节是不够的。因为除此之外还有很多内容，没办法写在这本书中，而且还有更多东西必须尝试之后才能学会。本书选择这些示例，并不是建议大家在工作中使用它们，其目的是让读者知道数据的各种存储方式，明白不同的数据库技术如何使用前面提到的概念。读者会看到这些数据库系统都需要何种程序代码，并且简单了解使用它们时所应遵循的开发思路。

有些人经常会觉得：因为 NoSQL 数据库没有模式，所以在应用程序的生命期中，可以毫无困难地改变其数据结构。本书不同意此观点，因为无模式的数据库其实隐含了一种模式，在实现数据结构变更时，也必须修改其规则。所以，第 12 章解释了数据如何在强模式与无模式系统之间迁移。

所有这一切都清楚地表明：NoSQL 不是独立存在的，也不会取代关系型数据库。第 13 章着眼于混合持久化领域的发展趋势：多种数据存储方式将共存，有时甚至会存在于同一个应用中。第 14 章将大家的视野扩展至本书之外，在混合持久化领域中，考虑一些前面没有涉及的技术。

掌握了前面所讲的全部内容之后，读者就应该明白如何选择合适的数据存储技术了。所以最后一章（第 15 章）提供了一些选择数据库时可以参考的建议。笔者认为，有两个关键因素：找到一种高效的编程模型，其数据存储模型要非常符合待开发的应用程序，并且确保其获取数据的效率与弹性均符合开发者的需求。从 NoSQL 诞生之初，我们就担心没有一套定义明确的流程可以遵循，现在，你仍然需要结合自己的需求，来验证自己所选择的数据库技术是否合适。

本书只是个简要的概述，所以笔者一直在尽力压缩篇幅。我们精选了自己认为最重要的信息，这部分内容读者就不必再去找了。如果打算认真研究这些技术，那就需要进一步研读本书以外的知识了，不过，我们还是希望本书能为你的探索之路开个好头。

还需要强调的是：计算机领域中的这些技术是日新月异的，存储技术的某些重要方面在不断变化，每年都会出现新的特性与新的数据库。笔者投入了巨大的精力来专门讲述概念，因为就算底层技术变了，对这些概念的理解也依然有价值。我们非常确信，

本书所讲的大部分概念都会历久绵长，但绝不能保证所有概念都会如此。

谁应该阅读本书

如果正在考虑选用某种形式的 NoSQL 数据库，那就应该阅读本书。选用 NoSQL 的原因可能是你打算做一个新的项目，也可能是既有项目遭遇瓶颈，所以要将其数据库迁移到 NoSQL 数据库上。

本书致力于给读者提供足够的信息，以判断自己所选的 NoSQL 技术是否符合需求，如果符合的话，应该深入研究哪些工具。我们设想本书的主要读者是架构师或技术主管，然而那些想大概了解这门新技术的软件管理人员也可以阅读本书。此外，对于想大概了解这项技术的开发人员来说，这也是本很好的入门读物。

本书不讲编程细节，也不去部署某个特定的数据库，那些内容留待更为专业的教材来写吧。我们还严格限制了本书的篇幅。笔者认为，这种书应该在坐飞机的时候读：它不会回答你提出的所有问题，但却会激发你提出一堆好问题来。

若是之前已经深入研究了 NoSQL 领域，那么本书可能不会增加你的知识储备。不过，它仍然有助于你将之前学到的东西解释给别人听。把围绕着 NoSQL 的争论理解清楚是很重要的，尤其当你要劝说别人在项目中也采用 NoSQL 技术时更是如此。

本书要讲的数据库类型

本书遵循常见的分类方式，也就是按照数据模型来划分各种 NoSQL 数据库。下表列出了 4 种数据模型，以及归属于每种数据模型的数据库。这份列表并不完整，其中只列出了较为常见的数据库。撰写本书时，在 http://nosql-database.org 与 http://nosql.mypopescu.com/kb/nosql 都可查阅到更为完整的列表。每个分类中，以斜体标出的数据库，都会在相关章节中作为范例来讲解。

数据模型	范例数据库	数据模型	范例数据库
键值（参见第 8 章）	BerkeleyDB LevelDB Memcached Project Voldemort Redis *Riak*	文档（参见第 9 章）	CouchDB *MongoDB* OrientDB RavenDB Terrastore

（续）

数据模型	范例数据库	数据模型	范例数据库
列族（参见第 10 章）	Amazon SimpleDB *Cassandra* HBase Hypertable	图（参见第 11 章）	FlockDB HyperGraphDB Infinite Graph *Neo4J* OrientDB

这样划分的目的是从每一类数据库中，选出一个最有代表性的工具来讲。尽管每个分类下列出的那些数据库各不相同，不可像这样一概而论，但是，书中提到的那些具体示例，其实大多数情况下也适用于此分类中的其他数据库。我们会从“键值数据库”、“文档数据库”、“列族数据库”和“图数据库”这 4 类中各选一个作为范例，此外，在必要时，还会提到可以满足某个特定功能的其他产品。

按数据模型来分类是可行的，但却失之武断。不同数据模型之间的界限往往是模糊的，比如键值和文档数据库（参见 2.2 节）之间的区别就不是很明显。许多数据库并不能明确地归入某一类。例如，OrientDB 称自己既是文档数据库又是图数据库。

致谢

首先感谢 ThoughtWorks ⊖ 的诸位同仁，在过去的几年中，很多同事在交付的项目中应用了 NoSQL。笔者写作本书的动机主要来源于他们的经验，而这些经验亦是能印证 NoSQL 技术价值的实用信息。目前为止通过使用 NoSQL 数据存储积累了一些有益的经验，基于这些经验，我们认为：NoSQL 是一项重要的数据存储技术，它正引发该领域内的一场重大变革。

我们也要感谢举办公开讲座、发表文章和博客来分享 NoSQL 使用心得的各种社群。若是大家都不愿意与同行分享研究成果的话，那么许多软件开发领域的发展就不为人知了。特别感谢谷歌及亚马逊的 BigTable 和 Dynamo 技术规范论文，它们对 NoSQL 的发展影响深远。也要感谢为开源 NoSQL 数据库的开发提供赞助及技术贡献的公司。这一次发生在数据存储领域的变革，与以往相比有一个较为有趣的差别：

⊖ ThoughtWorks 是一家在全球诸多国家都有分公司的软件设计与定制领袖企业。主要业务模式是通过咨询来改善企业 IT 组织及软件开发方法，以软件带动企业业务发展。详情参见：http://www.thoughtworks.com/。——译者注

NoSQL 的发展深度植根于开源工作。

特别感谢 ThoughtWorks 公司给予笔者时间来写作本书。我们两个大约同一时间加入 ThoughtWorks，并且在这里工作了 10 余年。ThoughtWorks 对我们来说一直是个非常友好的大家庭，同时也是知识和实践的来源。在这个良好的环境中，大家可以公开分享各自所学的知识，这与传统的系统交付公司（System Delivery Organization）非常不同。

Bethany Anders-Beck、Ilias Bartolini、Tim Berglund、Duncan Craig、Paul Duvall、Oren Eini、Perryn Fowler、Michael Hunger、Eric Kascic、Joshua Kerievsky、Anand Krishnaswamy、Bobby Norton、Ade Oshineye、Thiyagu Palanisamy、Prasanna Pendse、Dan Pritchett、David Rice、Mike Roberts、Marko Rodriquez、Andrew Slocum、Toby Tripp、Steve Vinoski、Dean Wampler、Jim Webber 和 Wee Witthawaskul 审阅了本书初稿，并提出了改进建议。

此外，Pramod 要感谢绍姆堡图书馆（Schaumburg Library）提供的一流服务和安静的写作空间；感谢爱女 Arhana 和 Arula，你们知道爸爸到图书馆是为了写书，而没有带你们同去；感谢爱妻 Rupali，你给了我巨大的支持和帮助，让我能够集中精力完成本书。

目　录

译者序

前言

第一部分　概　念

第 1 章　为什么使用 NoSQL ····· 2

1.1　关系型数据库的价值 ····· 3

1.1.1　获取持久化数据 ····· 3

1.1.2　并发 ····· 3

1.1.3　集成 ····· 4

1.1.4　近乎标准的模型 ····· 4

1.2　阻抗失谐 ····· 4

1.3　“应用程序数据库”与“集成数据库” ····· 6

1.4　蜂拥而来的集群 ····· 8

1.5　NoSQL 登场 ····· 9

1.6　要点 ····· 13

第 2 章　聚合数据模型 ····· 15

2.1　聚合 ····· 16

2.1.1　关系模型与聚合模型示例 ····· 16

2.1.2　面向聚合的影响 ····· 20

2.2　键值数据模型与文档数据模型 ····· 22

2.3　列族存储 ····· 23

2.4　面向聚合数据库总结 ····· 25

2.5　延伸阅读 ····· 26

2.6 要点……26
第 3 章 数据模型详解……27
3.1 关系……28
3.2 图数据库……29
3.3 无模式数据库……31
3.4 物化视图……33
3.5 构建数据存取模型……34
3.6 要点……39
第 4 章 分布式模型……40
4.1 单一服务器……41
4.2 分片……41
4.3 主从复制……43
4.4 对等复制……45
4.5 结合“分片”与“复制”技术……47
4.6 要点……48
第 5 章 一致性……49
5.1 更新一致性……50
5.2 读取一致性……51
5.3 放宽“一致性”约束……55
5.4 放宽“持久性”约束……60
5.5 仲裁……62
5.6 延伸阅读……63
5.7 要点……64
第 6 章 版本戳……65
6.1 “商业事务”与“系统事务”……66
6.2 在多节点环境中生成版本戳……68
6.3 要点……70
第 7 章 映射 – 化简……71
7.1 基本“映射 – 化简”……72

7.2 分区与归并……73
7.3 组合“映射－化简”计算……76
7.3.1 举例说明两阶段“映射－化简”……77
7.3.2 增量式“映射－化简”……80
7.4 延伸阅读……81
7.5 要点……81

第二部分 实 现

第 8 章 键值数据库……84
8.1 何谓“键值数据库”……85
8.2 键值数据库特性……86
8.2.1 一致性……86
8.2.2 事务……87
8.2.3 查询功能……87
8.2.4 数据结构……89
8.2.5 可扩展性……89
8.3 适用案例……90
8.3.1 存放会话信息……90
8.3.2 用户配置信息……90
8.3.3 购物车数据……90
8.4 不适用场合……90
8.4.1 数据间关系……90
8.4.2 含有多项操作的事务……91
8.4.3 查询数据……91
8.4.4 操作关键字集合……91
第 9 章 文档数据库……92
9.1 何谓文档数据库……93
9.2 特性……94
9.2.1 一致性……94

9.2.2 事务 95
9.2.3 可用性 96
9.2.4 查询功能 97
9.2.5 可扩展性 99
9.3 适用案例 100
9.3.1 事件记录 100
9.3.2 内容管理系统及博客平台 101
9.3.3 网站分析与实时分析 101
9.3.4 电子商务应用程序 101
9.4 不适用场合 101
9.4.1 包含多项操作的复杂事务 101
9.4.2 查询持续变化的聚合结构 101
第 10 章 列族数据库 102
10.1 何谓列族数据库 103
10.2 特性 103
10.2.1 一致性 105
10.2.2 事务 107
10.2.3 可用性 107
10.2.4 查询功能 108
10.2.5 可扩展性 110
10.3 适用案例 110
10.3.1 事件记录 110
10.3.2 内容管理系统与博客平台 111
10.3.3 计数器 111
10.3.4 限期使用 111
10.4 不适用场合 112
第 11 章 图数据库 113
11.1 何谓图数据库 114
11.2 特性 115

11.2.1 一致性 116
11.2.2 事务 117
11.2.3 可用性 117
11.2.4 查询功能 118
11.2.5 可扩展性 121
11.3 适用案例 122
11.3.1 互联数据 122
11.3.2 安排运输路线、分派货物和基于位置的服务 123
11.3.3 推荐引擎 123
11.4 不适用场合 123
第 12 章 模式迁移 124
12.1 模式变更 125
12.2 变更关系型数据库的模式 125
12.2.1 迁移全新项目 126
12.2.2 迁移既有项目 127
12.3 变更 NoSQL 数据库的模式 129
12.3.1 增量迁移 131
12.3.2 迁移图数据库的模式 132
12.3.3 改变聚合结构 132
12.4 延伸阅读 133
12.5 要点 133
第 13 章 混合持久化 134
13.1 各异的数据存储需求 135
13.2 混用各类数据库 135
13.3 将直接数据库操作封装为服务 137
13.4 扩展数据库以增强其功能 138
13.5 选用合适的数据库技术 139
13.6 企业使用混合持久化技术时的考量 139
13.7 部署复杂度 140