大数据应用与技术丛书

The Definitive Guide to MongoDB: A Complete Guide to Dealing with Big Data Using MongoDB, Third Edition

MongoDB 大数据处理权威指南 (第3版)

[美] David Hows
Peter Membrey
Eelco Plugge
Tim Hawkins 著

周连科 译

清华大学出版社

大数据应用与技术丛书

MongoDB 大数据处理权威指南(第3版)

[美] David Hows
Peter Membrey
Eelco Plugge
Tim Hawkins 著

周连科 译

清华大学出版社

北 京

The Definitive Guide to MongoDB: A Complete Guide to Dealing with Big Data Using MongoDB, Third Edition
By David Hows, Peter Membrey, Eelco Plugge, Tim Hawkins
EISBN：978-1-4842-1183-0

图书在版编目(CIP)数据

MongoDB 大数据处理权威指南(第 3 版) / (美) 戴维·豪斯(David Hows) 等著；周连科　译. —北京：清华大学出版社，2017
(大数据应用与技术丛书)
书名原文：The Definitive Guide to MongoDB: A Complete Guide to Dealing with Big Data Using MongoDB, Third Edition
ISBN 978-7-302-46387-0

Ⅰ. ①M…　Ⅱ. ①戴…　②周…　Ⅲ. ①关系数据库系统—指南　Ⅳ. ①TP311.138-62

中国版本图书馆 CIP 数据核字(2017)第 021748 号

责任编辑：王　军　韩宏志
装帧设计：牛艳敏
责任校对：曹　阳
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京密云胶印厂
经　　销：全国新华书店
开　　本：185mm×260mm　　印　　张：19　　字　　数：511 千字
版　　次：2017 年 3 月第 1 版　　印　　次：2017 年 3 月第 1 次印刷
印　　数：1～3000
定　　价：49.80 元

产品编号：071424-01

译者序

MongoDB 是一个高性能、开源、无模式的文档型数据库，由 C++语言编写，旨在为 Web 应用提供可扩展的高性能数据存储解决方案，是当前 NoSql 数据库中比较热门的一种。MongoDB 是一个介于关系数据库和非关系数据库之间的产品，是非关系数据库中功能最丰富、最像关系数据库的。它支持的数据结构非常松散，是类似于 json 的 bson 格式，因此可存储较复杂的数据类型。Mongo 最大的特点是它支持的查询语言非常强大，其语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且支持对数据建立索引，在许多场景下可用于替代传统的关系型数据库或键/值存储方式。

本书循序渐进地讲解 Mongo 的基础知识、用法和高级功能。每章内容都将展示一个单独的样例数据库，读者可按模块或线性方式阅读本书。本书讲解的主题包括：

- 如何在各种不同的平台上安装和配置 MongoDB。
- 如何使用各种不同的开发语言访问 MongoDB。
- 如何访问产品相关的社区，包括如何获得帮助和建议。
- 如何设计和构建利用 MongoDB 独特优势的应用。
- 如何优化、管理基于 MongoDB 的数据存储，以及如何进行故障检测。
- 如何创建跨多个服务器的可扩展、容错的服务器配置。

本书对于每一个知识点，都是先给出一个简要综述，然后通过例题来讲解。即使对 MongoDB 还不了解，也可以在读完本书后，独立搭建起自己的开发环境。对于有一定开发经验的读者，本书关于访问速度、文档结构的存储方式、大容量的存储、分片、大数据量下任意字段的查询等方面的讲解会令人大开眼界。总之，本书是一本学习 MongoDB 的不可多得的精品之作。

在这里要感谢清华大学出版社的编辑，他们为本书的翻译投入了巨大的热情并付出了很多心血。没有你们的帮助和鼓励，本书不可能顺利付梓。

对于这本经典之作，译者在翻译过程中力求忠于原文、通俗易懂，但是鉴于译者水平有限，错误和失误在所难免，如有任何意见和建议，请不吝指正。本书全部章节由周连科翻译，参与本次翻译的还有孔祥亮、陈跃华、杜思明、熊晓磊、曹汉鸣、陶晓云、王通、方峻、李小凤、曹晓松、蒋晓冬、邱培强、洪妍、李亮辉、高娟妮、曹小震、陈笑等。

可以说，选择本书是读者一个明智的选择。也希望通过学习本书，读者能够熟练掌握 MongoDB 的运用，使自己的工作更高效，为自己的职业发展增添一种技能。

作者简介

David Hows 以优异的成绩毕业于澳大利亚新南威尔士州的卧龙岗大学。他第一次接触计算机，是在尝试不花钱的情况下改进家庭 PC 的性能。这促使他加入 IT 行业，David 曾担任过系统管理员、性能工程师、软件开发者、解决方案架构师和数据库工程师等职务。David 也曾徒劳地尝试练过多年足球，并且他的咖啡杯上写着“Grumble Bum”。

Peter Membrey 是一位特许 IT 研究员，他拥有 15 年使用 Linux 和开源解决方案解决现实问题的经验。从 17 岁起他就是一位红帽认证工程师，也有幸在 Red Hat 工作过，并撰写了几本与开源解决方案相关的书籍。他拥有利物浦大学的信息安全硕士学位，目前是香港理工大学的博士生，他的研究方向包括时间同步、云计算、大数据和安全。他与自己贤惠的妻子 Sara 和儿子 Kaydyn 一起居住在香港。

Eelco Plugge 是一个工作和生活在荷兰的技术人员。目前是移动设备管理行业的工程师，他把大部分时间花在分析日志、配置和错误上，他之前是 McAfee 的一位数据加密专家，完成一些 IT/系统工程工作。Eelco 撰写了不少有关 MongoDB 和负载平衡的书，这位技术攻关人员对与 IT 安全相关的主题很有兴趣，与 IT 安全中的 MSc 相互补充。

Eelco 是两个孩子的父亲，闲暇时会离开电脑屏幕，偶尔看书。他感兴趣的事情有科学和自然方面的奇闻趣事、外汇交易(外汇)、编程、安全性和寿司。

Tim Hawkins 曾在 1993 年创建了世界上第一个在线分类广告门户网站 loot.com，之后负责管理雅虎欧盟的许多非媒体属性产品，例如搜索、本地搜索、邮件、消息和社交网络。他目前正在管理美国主要电子零售商的一个大型离岸团队，负责开发和部署下一代电子商务应用。他喜欢礼帽，讨厌复杂性。

技术编辑和贡献者简介

Stephen Steneker(又名 Stennie)是一位经验丰富的全能软件开发者、咨询师和讲师。Stephen 曾长期工作于澳大利亚的技术公司，是雅虎(澳大利亚和新西兰)、HomeScreen Entertainment 和 Grox 的技术负责人。他获得了英属哥伦比亚大学的学士学位(计算机科学)。

他目前的职位是 MongoDB Inc.的技术服务工程师，为 MongoDB 提供支持、咨询和培训。他经常在用户组和会议上发言，也是悉尼 MongoDB 用户组的创始人和管理者(http://www.meetup.com/ SydneyMUG/)。

在 Twitter、StackOverflow 或 Github 上可以找到他，他的 ID 为@stennie。

A. Jesse Jiryu Davis 是 MongoDB 在纽约的主管工程师，擅长 C、Python 和异步 I/O。他是 MongoDB C 驱动程序的首席开发人员、Motor 的作者，Python、PyMongo 和 Tornado 的贡献者。他与人合著了 A Web Crawler With asyncio Coroutines 一章的 Guido van Rossum，是开源应用程序的体系结构系列的第四本书。

致　谢

感谢 MongoDB 团队曾经和现有的所有成员。没有他们的帮助我们就无法完成本书的撰写，人们对数据存储的观点也会出现极大差异。我要特别感谢悉尼 MongoDB 团队的同事们，他们为本书的撰写提供了极大帮助。

——David Hows

撰写书籍是一个团队性工作。即使有时只有一个作者，在幕后也会有许多人帮助将所有素材整合在一起。因此我非常感谢 MongoDB 社区和 Apress 的所有人，感谢他们的辛勤工作、耐心和支持。特别要感谢 Dave 和 Eelco，他们帮助完成了本书的第 3 版。

我还想谢谢 Dou Yi，他是香港理工大学的博士生(关注安全与加密基础研究)，帮助我保持理智，耐心地解释我很久以前就应该掌握的数学概念。她帮我节省了大量时间，避免我走弯路。

特别要感谢 Rocky Chang 博士同意监督我的 EngD 研究，把我引入互联网测量世界(包括时间同步)。非常感谢他一直以来的支持、耐心和理解。

——Peter Membrey

感谢 9gag 社区，没有他们本书就无法在数月前完成。

——Eelco Plugge

我想感谢 mongodb-user 和 mongodb-dev 邮件列表中的所有成员，感谢他们能够忍耐我无休止地提出问题。

——Tim Hawkins

前　言

我接触数据库的时间相对较晚，从 2006 年才开始使用 MySQL。在学完所有计算机本科都提供的逻辑课程之后，我开始使用 MySQL 构建出一个完整的 LAMP 架构，其中用到了一些基本表。此时，我并未对 SQL 表管理的复杂性进行深入思考。不过，随着时间的流逝，我看到了存储越来越多异构数据的需要，并了解到随着时间的推移，简单的模式可以如何在它的生命周期中成长和演变。

我第一次接触 MongoDB 是在 2011 年，当时 Peter Membrey 建议我不要使用包含 30 个键行以及 30 个值行的上下文表，而是应该使用 MongoDB 实例来存储数据。就像所有开发者在面对一项新技术时的感觉一样，我对之嗤之以鼻并固执地且坚持我原来的计划。直到我使用糟糕的设计已经完成一半代码时，Peter 仍然坚持坚称我应该尝试使用 MongoDB，此时我才接受意见。如同所有来自 SQL 阵营的开发者一样，MongoDB 能够接受任何类型数据并且可以根据任何搜索条件返回这些数据的能力让我感到震惊。直到现在我也仍然大呼过瘾。

——David Hows

本书的组织方式

在本书中，Peter、Eelco Plugge、Tim Hawkins 和我都希望能够完全地展示出我们在学习 MongoDB 时的经验：在保持设计简单和清晰的同时，教会你如何使用 MongoDB。每章内容都将展示一个单独的样例数据库，因此你可以按照模块或线性的方式阅读本书；这完全取决于你自己。这意味着如果愿意，你可以略过某些特定的章节，而不会影响你对其他内容的学习。

本书的样例命令将显示在它们的后跟输出之前。，它们将以等宽的“代码体”字体出现，。而且命令并且会以加粗的方式显示，以便与其他结果输出加以予以区分。在大多数章节中，你都会遇到“提示”、“警告”和“注意”，它们包含有用的(有时甚至极其重要的)信息。

——David Hows

目　录

第 1 章

MongoDB 简介

想象一下这样的世界：数据库的使用是如此简单，以至于你忘记了正在使用它。再想象一下这样的世界：不需要任何复杂配置或设置，数据库仍然能够快速运行，并且具有良好的扩展性。想一下，如何可以只关注手上的任务，完成它，并可以按时下班。这听起来有点神奇，但是 MongoDB 承诺帮助你完成所有这些事情(甚至更多)。

MongoDB(源自单词 humongous)是一种较新的数据库，它没有表、模式、SQL 或行的概念。它没有事务、ACID 兼容性、连接、外键或其他许多容易在凌晨引起问题的特性。简单地说，MongoDB 是一个非常特别的数据库，它不同于你之前所使用的数据库，特别是关系数据库管理系统(Relational DataBase Management System，RDBMS)。事实上，当你知道 MongoDB 缺少对这些所谓的“标准”特性的支持时，你甚至可能已经在摇头了。

不要担心！接下来你将学习 MongoDB 的背景和指导原则，以及 MongoDB 团队这样做的原因。我们也将浏览 MongoDB 的特性列表，并提供足够的细节，从而保证你会完全沉迷于本书其余部分的话题中。

首先我们将了解创建 MongoDB 背后的哲学和理念，以及一些有趣的和有争议的设计决策。我们将学习面向文档数据库的概念、文档的组织方式以及它们的优缺点。我们还将学习 JSON 以及如何在 MongoDB 中应用 JSON。最后，我们将学习一些最引人注目的 MongoDB 特性。

1.1 了解 MongoDB 哲学

如同所有项目一样，MongoDB 有一套自己的设计哲学用于帮助指导开发。本节内容将介绍一些 MongoDB 数据库的基本原则。

1.1.1 使用正确的工具处理正确的工作

MongoDB 中最重要的哲学概念是：一鞋难合众人脚。在过去许多年中，传统的关系(SQL)数据库(MongoDB 是面向文档的数据库)一直被用于存储所有类型的数据。无论该数据是否符合关系模型(被用在所有的 RDBMS 数据库中，例如 MySQL、PostgreSQL、SQLite、Oracle、MS SQL Server 等)都无所谓；无论如何，数据都将被填充到数据库中。一般来说，部分原因是读取和修改数据库相比操作文件系统更加简单(和安全)。如果选择一本 PHP 方面的书籍，例如 *PHP for Absolute Beginners 2nd Editor*，Jason Lengstorf 和 Thomas Blom Hansen 编著(Apress，2014)，它将会教你使用数据库存储信息，而不是文件系统。这样做只是因为它更简单。在使用数据库存储信息的时候，开发者必须一直遵守它的工作流程。很明显，我们并未按照数据库原有的意图使用它。任何尝试在数据库中存储复杂数据、创建几张表，然后将它们组合在一起的开发者，

都会明白我们在讲什么。

MongoDB 团队决定他们不会创建另一个试图为所有人做所有事情的数据库。相反，该团队希望创建一个只用于处理文档的数据库，而不是行，并且它的速度要快，还要具有强大的扩展性和易用性。为实现这个目标，他们不得不忽略一些特性，这意味着 MongoDB 在某些特定情况下并非最理想的选择。例如，它缺少事务支持，意味着无法使用 MongoDB 编写财务应用。也就是说，MongoDB 可能对于之前提到的部分应用(例如存储复杂数据)是非常合适的。不过这不是个问题，因为你完全可以在财务模块中使用传统的 RDBMS，而使用 MongoDB 存储文档。这样的混合解决方案十分常见，并且一些产品级应用(例如 New York Times 网站)已经这样做了。

一旦适应 MongoDB 可能无法解决所有问题的理念之后，你会发现对于某些问题，MongoDB 可以完美地解决它们，例如分析(例如网站中使用的实时 Google Analytics)和复杂数据结构(例如博客文章和评论)。如果你仍然无法接受 MongoDB 是一个正式的数据库工具这个观点，那么请提前跳到 1.3 节“了解特性列表”，该部分内容将展示 MongoDB 的一些强大特性。

注意：

缺少事务和其他传统数据库特性并不意味着 MongoDB 不稳定，或者不能用于管理重要数据。

MongoDB 设计背后的另一个关键概念是：数据库应该一直具有多个副本。如果单个数据库实例出现问题，那么它可以轻松地通过另一个服务器恢复到正常状态。因为 MongoDB 的目标是尽可能地快，所以它采取了一些捷径，导致它难以从系统崩溃中恢复。开发者认为最严重的系统崩溃可能就是从服务中移除一台计算机；这意味着即使数据库完全恢复了，也无法正常使用。记住：MongoDB 不会尝试为所有人完成所有事情。但对于许多目的(例如构建 Web 应用)，MongoDB 是一个能够实现解决方案的完美工具。

现在你应该已经明白了 MongoDB 的起源。它不会尝试在所有方面都表现完美，也乐于承认它不会适用于所有人。不过，对于选择使用它的开发者，MongoDB 提供了一个功能丰富的面向文档数据库，并且对运行速度和扩展性做了优化。它也几乎可运行在任何目标上。MongoDB 的网站上包含了可运行在 Linux、Mac OS、Windows 和 Solaris 中的安装文件。

MongoDB 成功实现了这些目标，因此使用 MongoDB 有点像梦幻一样(至少对于我们来说)。不必担心如何将数据压缩到一张表中，只需要将数据组合在一起，然后将数据传递给 MongoDB。

考虑一个真实的例子。本书的合著者 Peter Membrey 最近开发了一个应用，用于存储一组 eBay 搜索结果。搜索结果的数量是不固定的(最多 100 个)，因此他需要一种简单的方式在数据库中将用户和搜索结果关联起来。Peter 曾尝试使用 MySQL，他不得不设计出一张表用于存储数据，并编写相应代码存储他的结果，然后再编写代码将结果组合在一起。这是一个相当常见的场景，大多数开发者在开发中经常会遇到。通常，我们不得不这样做；不过，对于该项目，他决定使用 MongoDB，因此事情就变得有点不同了。

具体地说，他添加了下面这样两行代码：

```
request['ebay_results'] = ebay_results_array
collection.save(request)
```

在本例中，request 是 Peter 的文档，ebay_result 是键，而 ebay_result_array 包含来自 eBay 的搜索结果。第二行保存了修改后的数据。将来当他访问该文档时，他将获得与之前格式完全相同的数据。他不需要任何 SQL；也不需要执行任何会话；更不需要创建任何新表或编写任何

特殊代码——MongoDB 就可以完成工作。他最终轻松地完成了工作，并按时回家。

1.1.2　天然缺少对事务的支持

MongoDB 开发者做出的另一个重要设计决定是：该数据库将不会包含事务的语义(用于保证数据一致性和存储的元素)。这是根据 MongoDB 的目标——简单、快速和可扩展做出的权衡之计。一旦移除这些重量级特性，数据库就可以更轻松地实现水平扩展。

通常在使用传统的 RDBMS 时，如果需要提高性能，就要购买更大、更快的机器。这是一种垂直扩展的方式，但只能做到这种程度。如果使用了水平扩展，就不需要使用一台大型机器，相反可以使用许多稍弱一些的小机器。从历史上讲，这样的服务器集群非常有利于负载均衡网站，但因为数据库的内部设计限制，这种方式一直存在着问题。

你可能会认为缺少事务的支持将构成致命缺陷；不过，许多人都忘了一种在 MySQL 中最流行的表类型(MYISAM——这也恰好是默认的表类型)也不支持事务。但这个事实并未阻止 MySQL 在之前的 10 年中变成并保持着主流开源数据库的地位。与开发解决方案时的大多数选择一样，是否使用 MongoDB 取决于个人的偏好，以及它是否符合你的项目。

注意：

MongoDB 在同时使用至少两台存储数据的服务器(成为 3 节点集群)时可以提供持久性，这也是为生产环境部署时推荐使用的基本配置。MongoDB 还支持“写顾虑(write concern)”的概念。即给定数量的节点确认写入成功，为数据安全存储提供了一个更强的保证。

自 MongoDB 1.8 版本以来，单个服务器的持久性通过事务日志来保证。这个日志只追加，每 100 毫秒刷新到磁盘中。

1.1.3　JSON 和 MongoDB

JSON(JavaScript Object Notation，JavaScript 对象记号)不止是一种交换数据的方式，它也是一种存储数据的良好方式。RDBMS 是高度结构化的，包含了多个文件(表)用于存储不同的数据。而 MongoDB 在单个文档中存储所有数据。在这方面，MongoDB 如同 JSON 一样，该模型为数据存储提供了丰富和高效的方式。而且，JSON 可高效地描述指定文档中的所有内容，所以不需要提前指定文档的结构。JSON 是没有模式的(不需要模式)，因为文档可以单独更新，或者独立于其他文档进行修改。另外，JSON 还通过将相关数据存储在同一位置的方式，提供了出色的性能。

实际上，MongoDB 并未使用 JSON 存储数据，而使用由 MongoDB 团队开发的一种称为 BSON(二进制 JSON 的英文简称)的开放数据格式。大多数情况下，使用 BSON 取代 JSON 并不会改变处理数据的方式。BSON 通过使计算机更容易处理和搜索文档的方式，使 MongoDB 处理速度变得更快。BSON 还添加了一些标准 JSON 不支持的特性，包括数字数据(例如 int32 和 int64)的许多扩展类型，以及支持处理二进制数据。在本章的 1.3.2 节“使用面向文档存储(BSON)”中将深入讲解 BSON。

RFC 7159 描述了 JSON 的初始规范，它由 Douglas Crockford 制订。JSON 允许在简单的、可读文本格式中展现复杂的数据结构，使用它通常比阅读和理解 XML 要简单得多。如同 XML 一样，JSON 被设计为一种在 Web 客户端(例如浏览器)和 Web 应用之间交换数据的方式。由于它描述对象的丰富方式和简单性，大多数开发者都选择它作为交换数据的格式。

复杂数据结构到底意味着什么？历史上，CSV(Comma-Separated Value，逗号分隔值)曾用于交换数据(确实，这种方式在今天也仍然非常常见)。CSV 是一种简单的文本格式，使用换行符分隔不同的行，使用逗号分隔不同的字段。例如下面的 CSV 文件：

```
Membrey, Peter, +852 1234 5678
Thielen, Wouter, +81 1234 5678
```

人们可以看懂这些信息，并迅速找到其中传递的信息。问题是，第 3 列的数字是电话号码还是传真号码？它们甚至可能是寻呼机的号码。为避免这种不确定性，CSV 文件通常会提供头字段，添加在文件的第一行。下面的片段显示了之前样例的更多细节：

```
Lastname, Firstname, Phone Number
Membrey, Peter, +852 1234 5678
Thielen, Wouter, +81 1234 5678
```

这样就好多了。但现在假设 CSV 文件提到的某些人拥有多个电话号码。可以添加另一个字段用作办公室电话号码，但当希望使用多个办公室电话号码时，会遇到同样的问题。当希望处理多个邮件地址时，也会遇到其他问题。大多数人都会有多个邮件地址，这些地址无法被清晰地标注为家庭或工作地址。此时，CSV 就暴露出了它的限制。CSV 文件只适于存储扁平且没有重复值的数据。通常会出现这种情况，提供几个 CSV 文件，每个都包含不同的信息。然后将这些文件结合到一起(通常使用 RDBMS)组成完整的数据。例如，一家大型零售公司在每天结束时，可能会从它的每个商店收到 CSV 文件格式的销售数据。这些文件将被结合到一起，这样公司才能看到某天的销售状况。这个处理并不简单，随着需要的文件数目增加，出错概率也会增大。

XML 很大程度上解决了这个问题，但使用 XML 未免“杀鸡用牛刀”：可以工作，但更像是大题小做，因为 XML 不仅用于机器的读取(而 JSON 用于人类的读取)，XML 是高度可扩展的。XML 并非定义特定的数据格式，而定义如何定义数据的格式。如果需要交换复杂并高度结构化的数据，XML 是非常有用的；不过对于简单数据交换，它通常会导致过多的工作。事实上，该场景就是“XML 地狱”这个词的来源。

JSON 提供了一个折中方案。与 CSV 不同，它可以存储结构化的内容；但与 XML 也不同，JSON 使内容易于理解和使用。下面使用 JSON 显示之前样例中的内容：

```
{
   "firstname": "Peter",
   "lastname": "Membrey",
   "phone_numbers": [
       "+852 1234 5678",
       "+44 1234 565 555"
   ]
}
```

在这个版本的例子中，每个 JSON 对象(或文档)包含所有必需的信息。例如 phone_numbers 中包含一个含有不同数字的列表。该列表可以变成任意大小。还可以指定数字的具体类型，如下所示：

```
{
    "firstname": "Peter",
    "lastname": "Membrey",
```