前言 PREFACE

丛书说明

本套“软件开发魔典”系列图书，是专门为编程初学者量身打造的编程基础学习与项目实践用书。

本丛书针对“零基础”和“中级”学习者，通过案例引导读者深入技能学习和项目实践。为满足初学者在框架知识方面的基础入门、扩展学习、编程技能、项目实践 4 个方面的职业技能需求，特意采用“知识基础→知识提高→核心技术→高级操作→项目实践”的结构和“由浅入深，由深到精”的学习模式进行讲解。

MongoDB 最佳学习线路

本书以 MongoDB 最佳的学习模式来设计内容结构，第 1～3 篇可使读者掌握 MongoDB 数据库安装、存储、查询、优化等基础知识和应用技能，第 4～5 篇可使读者拥有多个行业项目开发经验。读者如果遇到问题，可观看本书同步微视频，也可以通过在线技术支持让老程序员答疑解惑。

本书内容

全书分为 5 篇共 17 章。

第 1 篇（第 1～3 章）为基础篇，主要讲解对 MongoDB 的初步认识、MongoDB 的安装与配置以及 MongoDB 数据库的使用。读者在学完本篇后将会了解 MongoDB 的基本概念以及数据库的简单操作。

第 2 篇（第 4～7 章）为提高篇，主要讲解 MongoDB 存储、MongoDB 查询、聚合以及 MongoDB 的管理。通过本篇的学习，读者将对如何使用 MongoDB 有个深度的了解，为后面的开发奠定基础。

第 3 篇（第 8～11 章）为核心技术篇，主要讲解 MongoDB 数据库高级查询优化和大数据复制。学完本篇，读者将对 MongoDB 索引与优化、复制、分片以及使用 MongoDB 数据库进行综合性编程具有一定的综合应用能力。

第 4 篇（第 12～14 章）为高级操作篇，主要讲解 MongoDB 数据库在 Java、Node.js、Python 等语言开发中的应用。学完本篇，读者将能够贯通前面所学的各项知识和技能，学会在不同语言开发中应用 MongoDB 数据库的技能。

第 5 篇（第 15～17 章）为项目实践篇，主要讲解商品管理系统、舞蹈培训管理系统、网站帖子爬取及数据展示 3 个实战案例。通过本篇的学习，读者将对 MongoDB 数据库编程在项目开发中的实际应用有切身体会，为日后进行软件开发积累项目管理及实践开发的经验。

全书不仅融入了作者丰富的工作经验和多年的使用心得，还提供了大量来自工作现场的实例，具有较强的实战性和可操作性。读者系统学习后可以掌握 MongoDB 的基础知识，拥有全面的数据库操作能力、优良的团队协同技能和丰富的项目实战经验。编写本书的目的就是让数据库初学者快速成长为一名合格的中级程序员，通过演练积累项目开发经验和团队合作技能，在未来的职场中获取一个较高的起点，并能迅速融入软件开发团队。

本书特色

1. 结构科学，易于自学

本书在内容组织和范例设计中充分考虑初学者的特点，由浅入深，循序渐进，无论读者是否接触过框架，都能从本书中找到最佳的起点。

2. 视频讲解，细致透彻

为降低学习难度，提高学习效率。本书录制了同步微视频（模拟培训班模式），通过视频除了能轻松学会专业知识外，还能获取老师的软件开发经验，使学习变得更轻松有效。

3. 超多、实用、专业的范例和实践项目

本书结合实际工作中的应用范例逐一讲解 MongoDB 数据库的各种知识和技术，在项目实践篇中更以 3 个项目实践来总结本书前 14 章介绍的知识和技能，使读者在实践中掌握知识，轻松拥有项目开发经验。

4. 随时检测自己的学习成果

每章首页中，均提供了“学习指引”和“重点导读”，以指导读者重点学习及学后检查；章后的“就业面试技巧与解析”均根据当前最新求职面试（笔试）题精选而成，读者可以随时检测自己的学习成果，做到融会贯通。

5. 专业创作团队和技术支持

本书由聚慕课教育研发中心编著和提供在线服务。读者在学习过程中遇到任何问题，可加入图书读者（技术支持）QQ 群（661907764）进行提问，作者和资深程序员为读者在线答疑。

本书附赠超值王牌资源库

本书附赠了极为丰富、超值的王牌资源库，具体内容如下：

（1）王牌资源 1：随赠本书“配套学习与教学”资源库，提升读者的学习效率。

- 本书 316 节同步微视频（扫描二维码观看）。
- 本书 3 个大型项目案例以及 100 个实例源代码。
- 本书配套上机实训指导手册及本书教学 PPT 课件。

（2）王牌资源 2：随赠“职业成长”资源库，突破读者职业规划与发展瓶颈。

- 求职资源库：100 套求职简历模板、600 套毕业答辩模板和 80 套学术开题报告模板。

- 面试资源库：程序员面试技巧、200 道求职常见面试（笔试）真题与解析。
- 职业资源库：100 套岗位竞聘模板、程序员职业规划手册、开发经验及技巧集、软件工程师技能手册。

（3）王牌资源 3：随赠“软件开发魔典”资源库，拓展读者学习本书的深度和广度。

- 案例资源库：100 个实例及源码注释。
- 软件开发文档模板库：10 套八大行业软件开发文档模板。
- 编程水平测试系统：计算机水平测试、编程水平测试、编程逻辑能力测试、编程英语水平测试。
- 软件学习电子书资源库：MongoDB 常用工具查询电子书、MongoDB 常用命令查询电子书、MongoDB 数据库运维手册、MongoDB 可视化工具使用技巧和 MongoDB 吃内存问题及解决方案。

（4）王牌资源 4：编程代码优化纠错器。

- 本纠错器能让软件开发更加便捷和轻松，无须安装配置复杂的软件运行环境即可轻松运行程序代码。
- 本纠错器能一键格式化，让凌乱的程序代码更加规整美观。
- 本纠错器能对代码精准纠错，让程序查错不再难。

上述资源获取及使用

注意：由于本书不配送光盘，书中所用及上述资源均需借助网络下载才能使用。

1. 资源获取

采用以下任意途径，均可获取本书所附赠的超值王牌资源库。

（1）加入本书微信公众号“聚慕课 jumooc”，下载资源或者咨询关于本书的任何问题。

（2）加入本书图书读者服务（技术支持）QQ 群（661907764），读者可以打开群“文件”中对应的 Word 文件，获取网络下载地址和密码。

2. 使用资源

读者可通过 PC 端、App 端、微信端学习和使用本书微视频和资源。

本书适合哪些读者阅读

本书非常适合以下人员阅读。

- 没有任何 MongoDB 基础的初学者。
- 有一定的 MongoDB 开发基础，想精通编程的人员。
- 有一定的 MongoDB 开发基础，没有项目实践经验的人员。
- 正在进行软件专业相关毕业设计的学生。
- 大中专院校及培训学校的老师和学生。

创作团队

本书由聚慕课教育研发中心编著，高淼任主编，王峰、陈长生老师任副主编。参与本书编写的人员还

有蒋楠、陈梦、李良、陈献凯和裴垚等。

在编写过程中，我们尽己所能将最好的讲解呈现给读者，但也难免有疏漏和不妥之处，敬请读者不吝指正。

编　者

目录 CONTENTS

第 1 篇　基础篇

第 2 篇 提高篇

第 4 篇　高级操作篇

第 1 篇

基础篇

本篇是 MongoDB 的基础知识篇。从基本的对 MongoDB 的认识到基本用法的使用，结合一些我们日常生活中常用的案例的编写和分析带领读者简单快速地进入 MongoDB 的探索世界。

读者在学完本篇后将会了解到 MongoDB 数据库的基本概念和常用用法，掌握 MongoDB 的安装，创建删除简单的数据库、集合等知识，为后面更深入地学习 MongoDB 编程打下坚实的基础。

- **第 1 章** 初识 MongoDB 世界——认识 MongoDB
- **第 2 章** MongoDB 使用基础——MongoDB 的安装与配置
- **第 3 章** 数据库程序的操作——MongoDB 数据库的使用

第1章

初识 MongoDB 世界——认识 MongoDB

本章概述

本章主要介绍 NoSQL 数据库的发展以及它其中最热门的 MongoDB 数据库基础知识。通过本章内容的学习，读者可以学习到 MongoDB 数据库的体系结构、特征、核心服务以及 MongoDB 的数据模型等。

本章要点

- NoSQL 的种类及特征
- MongoDB 体系结构
- MongoDB 键特性
- MongoDB 的核心服务和工具
- MongoDB 数据模型

1.1 NoSQL

NoSQL，泛指非关系型数据库。随着互联网 Web 2.0 网站的兴起，传统的关系型数据库在应付 Web 2.0 网站，特别是超大规模和高并发的 SNS 类型的 Web 2.0 纯动态网站时已经显得力不从心，暴露了很多难以克服的问题，而非关系型数据库则由于其自身的特点得到了非常迅速的发展。NoSQL 数据库的产生就是为了应对大规模数据集合以及多重数据种类带来的挑战，尤其是大数据应用难题。

1.1.1 NoSQL 简史

NoSQL 一词最早出现于 1998 年，是 Carlo Strozzi 开发的一个轻量、开源、不提供 SQL 功能的关系型数据库。

2009 年，Last.fm 的 Johan Oskarsson 发起了一次关于分布式开源数据库的讨论，来自 Rackspace 的 Eric Evans 再次提出了 NoSQL 的概念，这时的 NoSQL 主要指非关系型、分布式、不提供 ACID 的数据库设计模式。

2009 年在亚特兰大举行的 no:sql(east)讨论会是一个里程碑，其口号是 select fun, profit from real_world where relational=false;。因此，对 NoSQL 最普遍的解释是“非关联型的”，强调 Key-Value Stores 和文档数据库的优点，而不是单纯地反对 RDBMS。

1.1.2　NoSQL 的种类及其特性

NoSQL 数据库有多种类型，每种类型都有各自的特点，如表 1-1 所示。

表 1-1　NoSQL 数据库分类

类　型	部 分 代 表	特　点
列存储	HBase Cassandra HyperTable	顾名思义，是按列存储数据的。最大的特点是方便存储结构化和半结构化数据，方便做数据压缩，对针对某一列或者某几列的查询有非常大的 IO 优势
文档存储	MongoDB CouchDB	文档存储一般用类似 JSON 的格式存储，存储的内容是文档型的。这样也就有机会对某些字段建立索引，实现关系型数据库的某些功能
键值存储	Tokyo Cabinet/Tyrant Berkeley DB MemcacheDB Redis	可以通过键快速查询到其值。一般来说，存储不管值的格式，照单全收
图存储	Neo4j FlockDB	图形关系的最佳存储。使用传统关系型数据库来解决的话性能低下，而且设计使用不方便
对象存储	db4o Versant	通过类似面向对象语言的语法操作数据库，通过对象的方式存取数据
XML 数据库	Berkeley DB XML BaseX	高效的存储 XML 数据，并支持 XML 的内部查询语法，比如 XQuery、Xpath

下面介绍两种不同类型的数据库。

1）面向列的数据库。Cassandra、HBase、HyperTable 属于这种类型。

普通的关系型数据库都是以行为单位来存储数据的，擅长以行为单位读入数据，比如特定条件数据的获取。因此，关系型数据库也被称为面向行的数据库。相反，面向列的数据库是以列为单位来存储数据的，擅长以列为单位读入数据。

面向列的数据库具有高扩展性，即使数据增加也不会降低相应的处理速度（特别是写入速度），所以它主要应用于需要处理大量数据的情况。另外，把它作为批处理程序的存储器来对大量数据进行更新也是非常有用的。但由于面向列的数据库跟现行数据库存储的思维方式有很大不同，故应用起来十分困难。

2）面向文档的数据库。MongoDB、CouchDB 属于这种类型，它们属于 NoSQL 数据库，但与键值存储相异。

（1）不定义表结构。

即使是不定义表结构，也可以像定义表结构一样使用，还省去了变更表结构的麻烦。

（2）可以使用复杂的查询条件。

与键值存储不同的是，面向文档的数据库可以通过复杂的查询条件来获取数据，虽然不具备事务处理

和 Join 这些关系型数据库所具有的处理能力，但除此以外的其他处理基本上都能实现。

3）键值存储的数据库。

它的数据是以键值的形式存储的，虽然它的速度非常快，但基本上只能通过键的完全一致查询获取数据，根据数据的保存方式可以分为临时性、永久性和两者兼具三种。

（1）临时性。

所谓临时性就是数据有可能丢失，memcached 把所有数据都保存在内存中，这样保存和读取的速度非常快，但是当 memcached 停止时，数据就不存在了。由于数据保存在内存中，所以无法操作超出内存容量的数据，旧数据会丢失。总体来说，在内存中保存数据、可以进行非常快速的保存和读取处理、数据有可能丢失。

（2）永久性。

所谓永久性就是数据不会丢失，这里的键值存储是把数据保存在硬盘上，与临时性比起来，由于必然要发生对硬盘的 IO 操作，所以性能上还是有差距的，但数据不会丢失是它最大的优势。总体来说，在硬盘上保存数据、可以进行非常快速的保存和读取处理（但无法与 memcached 相比）、数据不会丢失。

（3）两者兼具。

Redis 属于这种类型。Redis 有些特殊，临时性和永久性兼具。Redis 首先把数据保存在内存中，在满足特定条件（默认是 15 分钟一次以上，5 分钟内 10 个以上，1 分钟内 10000 个以上的键发生变更）的时候将数据写入到硬盘中，这样既确保了内存中数据的处理速度，又可以通过写入硬盘来保证数据的永久性，这种类型的数据库特别适合处理数组类型的数据。总体来说，同时在内存和硬盘上保存数据、可以进行非常快速的保存和读取处理、保存在硬盘上的数据不会消失（可以恢复）、适合于处理数组类型的数据。

1.1.3 NoSQL 特点

关系型数据库经过几十年的发展，各种优化工作已经做得很深了，而 NoSQL 系统也从中吸收了关系型数据库的技术，我们从系统设计的角度来了解一下 NoSQL 数据库的四大特点。

1. 索引支持

关系型数据库创立之初没有想到今天的互联网应用对可扩展性提出如此高的要求，因此，设计时主要考虑的是简化用户的工作，SQL 语言的产生促成数据库接口的标准化，从而形成了 Oracle 这样的数据库公司并带动了上下游产业链的发展。关系型数据库的单机存储引擎支持索引，比如 MySQL 的 InnoDB 存储引擎需要支持索引，而 NoSQL 系统的单机存储引擎是纯粹的，只需要支持基于主键的随机读取和范围查询。NoSQL 系统在系统层面提供对索引的支持，比如有一个用户表，主键为 user_id，每个用户有很多属性，包括用户名，照片 ID（photo_id），照片 URL，在 NoSQL 系统中如果需要对 photo_id 建立索引，可以维护一张分布式表，表的主键为形成的二元组。关系型数据库由于需要在单机存储引擎层面支持索引，大大降低了系统的可扩展性，使得单机存储引擎的设计变得很复杂。

2. 并发事务处理

关系型数据库有一整套的关于事务并发处理的理论，比如锁的粒度是表级、页级还是行级，多版本并发控制机制 MVCC，事务的隔离级别，死锁检测，回滚，等等。然而，互联网应用大多数的特点都是多读少写，比如读和写的比例是 10∶1，并且很少有复杂事务需求，因此，一般可以采用更为简单的 copy-on-write 技术：单线程写，多线程读，写的时候执行 copy-on-write，写不影响读服务。NoSQL 系统这样的假设简化

了系统的设计，减少了很多操作的 overhead，提高了性能。

3. 数据结构

关系型数据库的存储引擎的数据结构是一棵磁盘 *B*+树，为了提高性能，可能需要有 Insert Buffer 聚合写，Query Cache 缓存读，经常需要实现类似 Linux page cache 的缓存管理机制。数据库中的读和写是互相影响的，写操作也因为时不时需要将数据输出到磁盘而性能不高。简而言之，关系型数据库存储引擎的数据结构是通用的动态更新的 *B*+树。然而，在 NoSQL 系统中，比如 Bigtable 中采用 SSTable+MemTable 的数据结构，数据先写入到内存的 MemTable，达到一定大小或者超过一定时间才会备份到磁盘生成 SSTable 文件，SSTable 是只读的。如果说关系型数据库存储引擎的数据结构是一棵动态的 *B*+树，那么 SSTable 就是一个排好序的有序数组。很明显，实现一个有序数组比实现一棵动态 *B*+树且包含复杂的并发控制机制要简单高效得多。

4. Join 操作

关系型数据库需要在存储引擎层面支持 Join，而 NoSQL 系统一般根据应用来决定 Join 实现的方式。举个例子，有两张表：用户表和商品表，每个用户下可能有若干个商品，用户表的主键为 user_id，用户和商品的关联属性存放在用户表中，商品表的主键为 item_id，商品属性包括商品名和商品 URL，等等。假设应用需要查询一个用户的所有商品并显示商品的详细信息，普通的做法是先从用户表查找指定用户的所有 item_id，然后对每个 item_id 去商品表查询详细信息，即执行一次数据库 Join 操作，这必然带来了很多的磁盘随机读，并且由于 Join 带来的随机读的局部性不好，缓存的效果往往也是有限的。在 NoSQL 系统中，我们往往可以将用户表和商品表集成到一张宽表中，这样虽然存储了额外的信息，但却换来了查询的高效。

1.1.4　NoSQL 的优缺点

业界为了解决更多用户的需求，推出了多款新类型的数据库，并且由于它们在设计上和传统的 NoSQL 数据库相比有很大的不同，所以被统称为 NoSQL 系列数据库。总的来说，在设计上，它们非常关注对数据高并发读写和对海量数据的存储等，与关系型数据库相比，它们在架构和数据模型方面做了“减法”，而在扩展和并发等方面做了“加法”。现在主流的 NoSQL 数据库有 BigTable、HBase、Cassandra、SimpleDB、CouchDB、MongoDB 和 Redis 等。接下来，我们了解一下 NoSQL 数据库到底存在哪些优缺点。

在优势方面，主要体现在下面这三点：

（1）简单的扩展。典型例子是 Cassandra，由于其架构是类似于经典的 P2P，所以能通过轻松地添加新的节点来扩展这个集群。

（2）快速的读写。主要例子有 Redis，由于其逻辑简单，而且纯内存操作，使得其性能非常出色，单节点每秒可以处理超过 10 万次读写操作。

（3）低廉的成本。这是大多数分布式数据库共有的特点，因为主要都是开源软件，没有昂贵的许可证成本。

虽然有以上优势，但 NoSQL 数据库还存在着很多的不足，常见的主要有下面这几个：

（1）不提供对 SQL 的支持。如果不支持 SQL 这样的工业标准，将会对用户产生一定的学习和应用迁移成本。

（2）支持的特性不够丰富。现有产品所提供的功能都比较有限，大多数 NoSQL 数据库都不支持事务，