

面向新手，研读Redis源码，构建高性能数据库，有讲解、有实例、有解析！



从零开始学Redis

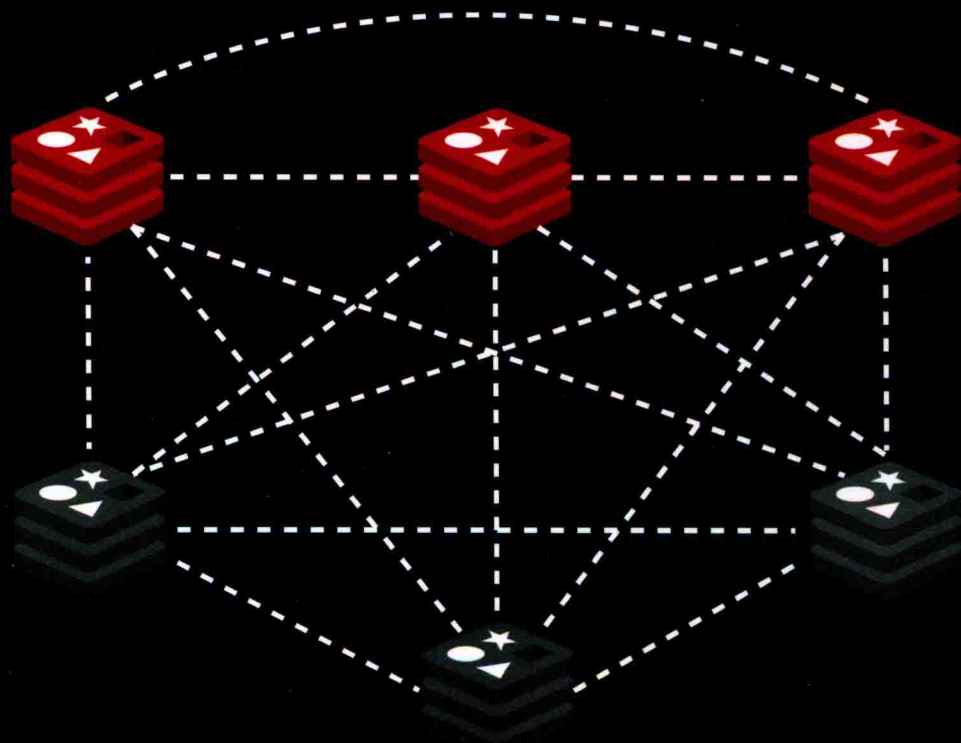
高洪涛 刘河飞 编著

系统全面：全书讲解150多个命令，涵盖大部分应用场景

层层深入：从基本命令到实战应用，逐步提高

实例丰富：每个命令的讲解都结合实例，随学随用

面向实战：配有Java、SpringBoot、Python操作Redis的方法



非
外
借



中国工信出版集团



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

内容简介

从零开始学 Redis

高洪涛 刘河飞 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

Redis 数据库是目前比较热门的数据库，拥有巨大的用户量。本书主要分 3 部分讲解 Redis 数据库。第一部分 Redis 初始篇，详细介绍了 Redis 的数据类型，以及 Redis 的大部分命令，并结合实际操作进行了演示。第二部分 Redis 进阶篇，深入讲解了 Redis 的客户端、服务器端、数据结构的底层，以及 Redis 的排序、事务、持久化、集群等相关功能，同时讲解了 Redis 的其他高级功能，如慢查询、流水线、地理位置、位图等，并结合实际操作，步步演示。第三部分 Redis 实战篇，分别介绍了使用 Java、SpringBoot、Python 来操作 Redis 的实例，以帮助读者更好地学习 Redis。通过阅读本书，读者可以快速掌握 Redis 的相关命令及功能用法，同时结合实战学习，可以将 Redis 熟练应用于实际的生产开发中。

本书面向大多数软件开发者，如 Redis 初学者，或者具有相关后台开发经验的开发者。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

从零开始学 Redis / 高洪涛, 刘河飞编著. —北京: 电子工业出版社, 2019.5

ISBN 978-7-121-36311-5

I. ①从… II. ①高… ②刘… III. ①数据库—基本知识 IV. ①TP311.138

中国版本图书馆 CIP 数据核字 (2019) 第 068268 号

策划编辑: 李 冰

责任编辑: 李 冰 特约编辑: 田学清

印 刷: 三河市华成印务有限公司

装 订: 三河市华成印务有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱

邮编: 100036

开 本: 787×1092 1/16 印张: 24.75 字数: 634 千字

版 次: 2019 年 5 月第 1 版

印 次: 2019 年 5 月第 1 次印刷

定 价: 89.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010) 88254888, 88258888。

质量投诉请发邮件至 zltz@phei.com.cn，盗版侵权举报请发邮件到 dbqq@phei.com.cn。

本书咨询联系方式: libing@phei.com.cn。

前言

2016年10月，在工作的过程中我偶然接触到 Redis，便开始自学，从学习 Redis 的安装，到熟悉它的数据类型及相关命令，再到它的实际应用。在企业工作的这段时间里，我也常用到 Redis 做缓存系统，实现高并发的存储与读/写，以及 Redis 相关的高级功能，觉得非常实用。经过不断整理与总结，2018年我决定写本书，与诸位爱好 Redis 并能实际应用 Redis 的读者进行分享。

有所得，必有所失。通常我白天正常上班，晚上或周末开始撰写本书。感谢坚持不懈的自己，多少个日夜的坚持，才换来本书的完稿。在得到的同时，我也失去了许多。为了完成本书的编写，我放弃了大量的休息时间，也很少锻炼身体，以致常常生病，同时变成了一个“宅男”，缺少了与人交流沟通的机会。一句话总结就是：沉迷写书，日渐消瘦。

在写作本书的过程中，我深刻地体会到：做事之所以会半途而废，往往不是因为难度较大，而是因为觉得成功离我们较远。确切地说，我们不是因为失败而放弃，而是因为倦怠而失败。在人生的旅途中，我们多思考一下，多坚持一下，同时也多鼓励一下自己，一生中也许会少许多懊悔与惋惜，我们离成功也就会越来越近。在此，我希望阅读本书的读者坚持学习，不断进步。累了，就休息一下，但是不要懈怠；迷茫了，就调整一下自己努力的方向，但是不要放弃努力。既然选择了，就要坚持下去，相信未来的自己一定会感谢现在努力的自己。

夜已深，茶已凉，就不再多叙，望诸君谨记：学虽易，学好难，且学且珍惜。

到目前为止，Redis 还在不断更新，用户量也在不断扩大，这也足以说明 Redis 的强大之处。希望诸君学习愉快，技术更上一层楼。

本书内容简介

全书分3部分。

第一部分（第1~5章）为 Redis 初始篇，首先介绍了对 NoSQL 的认识，然后介绍了 Redis 环境的搭建与启动，以及它的相关客户端，最后介绍了 Redis 的数据类型命令、必备命令及 Redis 数据库的相关知识。

第二部分（第6~13章）为 Redis 进阶篇，首先讲解了 Redis 客户端与服务器的相关属性与函数，然后结合 Redis 的底层源码深入讲解了 Redis 的底层实现和相关的 API 函数，最后讲解了 Redis 的相关功能，如排序、事务、消息订阅、持久化、集群，以及其他高级功能，如慢

查询、流水线、地理位置、位图等，旨在帮助读者深入理解 Redis，并掌握其精髓。

第三部分（第 14~16 章）为 Redis 实战篇，这部分结合实际应用，讲解了对 Redis 数据库的操作，以 Java 语言、最流行的 SpringBoot 框架及 Python 语言为主，并提供了大量的实例代码，旨在提高读者的动手能力，帮助读者真正掌握 Redis 数据库。

本书的特点

本书以模块化为主，从了解、熟悉 Redis，到 Redis 的进阶学习，最后结合实际应用，为读者展示了 Redis 数据库的使用。本书前面的章节详细介绍了关于 Redis 的 200 多个命令，并结合实际操作为读者演示；后面的章节结合相关的原理图、流程图，为读者介绍了 Redis 的相关功能，如排序、事务、消息订阅、持久化、集群，以及其他高级功能，如慢查询、流水线、地理位置、位图等。最后以实际应用为主，提供了 Java、SpringBoot、Python 操作 Redis 等相关实例。

致谢

首先，非常感谢张增强老师对我的肯定与支持，并给予我宽松的时间，让我得以完成本书的编写。其次，感谢坚持不懈的自己，在多少个黑夜与周末，我不断地坚持，换来了此书的完稿。最后，感谢郭豪、闫凯峰等好友的帮助，在他们的支持下，我不断地向前，不断地进步着。

目 录

第一部分 Redis 初始篇

| | |
|--------------------------------------|----|
| 第 1 章 初识 NoSQL | 2 |
| 1.1 什么是 NoSQL | 2 |
| 1.2 NoSQL 与传统关系型数据库的比较 | 3 |
| 1.3 在什么应用场景下使用 NoSQL | 4 |
| 1.4 NoSQL 的数据模型 | 5 |
| 1.5 NoSQL 数据库的分类 | 6 |
| 1.5.1 NoSQL 数据库分类简介 | 6 |
| 1.5.2 各类 NoSQL 数据库的比较 | 6 |
| 第 2 章 认识 Redis | 8 |
| 2.1 Redis 简介 | 8 |
| 2.1.1 Redis 的由来 | 8 |
| 2.1.2 什么是 Redis | 8 |
| 2.1.3 Redis 的特性 | 8 |
| 2.1.4 Redis 的使用场景 | 9 |
| 2.2 搭建 Redis 环境 | 10 |
| 2.2.1 在 Window 环境下搭建 | 10 |
| 2.2.2 在 Linux 环境下搭建 | 13 |
| 2.3 Redis 客户端 | 14 |
| 2.3.1 命令行客户端 | 14 |
| 2.3.2 可视化客户端 | 15 |
| 2.3.3 编程客户端 | 17 |
| 2.4 Redis 的启动方式 | 18 |
| 2.4.1 在 Window 环境下的启动方式 | 18 |
| 2.4.2 在 Linux 环境下的启动方式 | 19 |
| 第 3 章 Redis 数据类型 | 21 |
| 3.1 Redis 数据类型之字符串 (String) 命令 | 21 |

| | | |
|-------------------------------|---------------------------------------|-----------|
| 3.1.1 | 设置键值对 | 22 |
| 3.1.2 | 获取键值对 | 24 |
| 3.1.3 | 键值对的偏移量 | 26 |
| 3.1.4 | 设置键的生存时间 | 26 |
| 3.1.5 | 键值对的值操作 | 27 |
| 3.1.6 | 键值对的计算 | 29 |
| 3.1.7 | 键值对的值增量 | 31 |
| 3.2 | Redis 数据类型之哈希 (Hash) 命令 | 34 |
| 3.2.1 | 设置哈希表域的值 | 34 |
| 3.2.2 | 获取哈希表中的域和值 | 36 |
| 3.2.3 | 哈希表统计 | 38 |
| 3.2.4 | 为哈希表中的域加上增量值 | 39 |
| 3.2.5 | 删除哈希表中的域 | 40 |
| 3.3 | Redis 数据类型之列表 (List) 命令 | 41 |
| 3.3.1 | 向列表中插入值 | 41 |
| 3.3.2 | 获取列表元素 | 44 |
| 3.3.3 | 删除列表元素 | 46 |
| 3.3.4 | 移动列表 | 50 |
| 3.3.5 | 列表模式 | 52 |
| 3.4 | Redis 数据类型之集合 (Set) 命令 | 53 |
| 3.4.1 | 向集合中添加元素 | 53 |
| 3.4.2 | 获取集合元素 | 54 |
| 3.4.3 | 集合运算 | 57 |
| 3.4.4 | 删除集合元素 | 60 |
| 3.5 | Redis 数据类型之有序集合 (Sorted Set) 命令 | 61 |
| 3.5.1 | 添加元素到有序集合中 | 62 |
| 3.5.2 | 获取有序集合元素 | 63 |
| 3.5.3 | 有序集合排名 | 69 |
| 3.5.4 | 有序集合运算 | 71 |
| 3.5.5 | 删除有序集合元素 | 72 |
| 第 4 章 Redis 必备命令 | | 76 |
| 4.1 | 键 (key) 命令 | 76 |
| 4.1.1 | 查询键 | 76 |
| 4.1.2 | 修改键 | 79 |
| 4.1.3 | 键的序列化 | 81 |
| 4.1.4 | 键的生存时间 | 82 |
| 4.1.5 | 键值对操作 | 85 |

| | | |
|-------|----------------------------|-----|
| 4.1.6 | 删除键 | 89 |
| 4.2 | HyperLogLog 命令 | 90 |
| 4.2.1 | 添加键值对到 HyperLogLog 中 | 90 |
| 4.2.2 | 获取 HyperLogLog 的基数 | 91 |
| 4.2.3 | 合并 HyperLogLog | 92 |
| 4.3 | 脚本命令 | 92 |
| 4.3.1 | 缓存中的 Lua 脚本 | 92 |
| 4.3.2 | 对 Lua 脚本求值 | 93 |
| 4.3.3 | 杀死或清除 Lua 脚本 | 95 |
| 4.4 | 连接命令 | 96 |
| 4.4.1 | 解锁密码 | 96 |
| 4.4.2 | 断开客户端与服务器的连接 | 97 |
| 4.4.3 | 查看服务器的运行状态 | 97 |
| 4.4.4 | 输出打印消息 | 97 |
| 4.4.5 | 切换数据库 | 98 |
| 4.5 | 服务器命令 | 98 |
| 4.5.1 | 管理客户端 | 98 |
| 4.5.2 | 查看 Redis 服务器信息 | 101 |
| 4.5.3 | 修改并查看相关配置 | 108 |
| 4.5.4 | 数据持久化 | 111 |
| 4.5.5 | 实现主从服务 | 112 |
| 4.5.6 | 服务器管理 | 114 |
| 第 5 章 | Redis 数据库 | 116 |
| 5.1 | Redis 数据库切换 | 116 |
| 5.2 | Redis 数据库中的键操作 | 117 |
| 5.2.1 | 添加键 | 118 |
| 5.2.2 | 修改键 | 118 |
| 5.2.3 | 删除键 | 120 |
| 5.2.4 | 取键值 | 121 |
| 5.3 | Redis 数据库通知 | 121 |
| 5.3.1 | 数据库通知分类 | 122 |
| 5.3.2 | 数据库通知的实现原理 | 124 |

第二部分 Redis 进阶篇

| | | |
|-------|---------------------|-----|
| 第 6 章 | Redis 客户端与服务器 | 126 |
| 6.1 | Redis 客户端 | 126 |

| | | |
|--------------|----------------------------------|------------|
| 6.1.1 | 客户端的名字、套接字、标志和时间属性 | 126 |
| 6.1.2 | 客户端缓冲区 | 129 |
| 6.1.3 | 客户端的 authenticated 属性 | 131 |
| 6.1.4 | 客户端的 argv 和 argc 属性 | 131 |
| 6.1.5 | 关闭客户端 | 132 |
| 6.2 | Redis 服务器 | 132 |
| 6.2.1 | 服务器处理命令请求 | 132 |
| 6.2.2 | 服务器发送命令 | 133 |
| 6.2.3 | 服务器执行命令 | 134 |
| 6.2.4 | 服务器返回命令结果 | 135 |
| 6.3 | 服务器函数 | 136 |
| 6.3.1 | serverCron 函数 | 136 |
| 6.3.2 | trackOperationsPerSecond 函数 | 137 |
| 6.3.3 | sigtermHandler 函数 | 137 |
| 6.3.4 | clientsCron 函数 | 138 |
| 6.3.5 | databasesCron 函数 | 138 |
| 6.4 | 服务器属性 | 138 |
| 6.4.1 | cronloops 属性 | 138 |
| 6.4.2 | rdb_child_pid 与 aof_child_pid 属性 | 138 |
| 6.4.3 | stat_peak_memory 属性 | 139 |
| 6.4.4 | lruclock 属性 | 140 |
| 6.4.5 | mstime 与 unixtime 属性 | 141 |
| 6.4.6 | aof_rewrite_scheduled 属性 | 141 |
| 6.5 | Redis 服务器的启动过程 | 141 |
| 6.5.1 | 服务器状态结构的初始化 | 142 |
| 6.5.2 | 相关配置参数的加载 | 142 |
| 6.5.3 | 服务器数据结构的初始化 | 142 |
| 6.5.4 | 数据库状态的处理 | 143 |
| 6.5.5 | 执行服务器的循环事件 | 144 |
| 第 7 章 | Redis 底层数据结构 | 145 |
| 7.1 | Redis 简单动态字符串 | 145 |
| 7.1.1 | SDS 的实现原理 | 145 |
| 7.1.2 | SDS API 函数 | 147 |
| 7.2 | Redis 链表 | 148 |
| 7.2.1 | 链表的实现原理 | 148 |
| 7.2.2 | 链表 API 函数 | 150 |

| | | |
|--------------|---------------------|------------|
| 7.3 | Redis 压缩列表 | 151 |
| 7.3.1 | 压缩列表的实现原理 | 151 |
| 7.3.2 | 压缩列表 API 函数 | 153 |
| 7.4 | Redis 快速列表 | 154 |
| 7.4.1 | 快速列表的实现原理 | 154 |
| 7.4.2 | 快速列表 API 函数 | 156 |
| 7.5 | Redis 字典 | 157 |
| 7.5.1 | 字典的实现原理 | 157 |
| 7.5.2 | 字典 API 函数 | 160 |
| 7.6 | Redis 整数集合 | 161 |
| 7.6.1 | 整数集合的实现原理 | 161 |
| 7.6.2 | 整数集合 API 函数 | 163 |
| 7.7 | Redis 跳表 | 164 |
| 7.7.1 | 跳表的实现原理 | 164 |
| 7.7.2 | 跳表 API 函数 | 166 |
| 7.8 | Redis 中的对象 | 167 |
| 7.8.1 | 对象类型 | 167 |
| 7.8.2 | 对象的编码方式 | 171 |
| 第 8 章 | Redis 排序 | 174 |
| 8.1 | SORT 排序命令 | 174 |
| 8.2 | 升序 (ASC) 与降序 (DESC) | 176 |
| 8.3 | BY 参数的使用 | 177 |
| 8.4 | LIMIT 参数的使用 | 180 |
| 8.5 | GET 与 STORE 参数的使用 | 181 |
| 8.6 | 多参数执行顺序 | 185 |
| 第 9 章 | Redis 事务 | 187 |
| 9.1 | Redis 事务简介 | 187 |
| 9.2 | Redis 事务的 ACID 特性 | 188 |
| 9.2.1 | 事务的原子性 | 188 |
| 9.2.2 | 事务的一致性 | 190 |
| 9.2.3 | 事务的隔离性 | 192 |
| 9.2.4 | 事务的持久性 | 193 |
| 9.3 | Redis 事务处理 | 194 |
| 9.3.1 | 事务的实现过程 | 194 |
| 9.3.2 | 悲观锁和乐观锁 | 197 |
| 9.3.3 | 事务的 WATCH 命令 | 198 |

| | |
|----------------------------------|-----|
| 第 10 章 Redis 消息订阅 | 202 |
| 10.1 消息订阅发布概述 | 202 |
| 10.2 消息订阅发布实现 | 203 |
| 10.2.1 消息订阅发布模式命令 | 203 |
| 10.2.2 消息订阅功能之订阅频道 | 208 |
| 10.2.3 消息订阅功能之订阅模式 | 210 |
| 10.3 Redis 消息队列 | 211 |
| 10.3.1 消息订阅发布模式的原理 | 211 |
| 10.3.2 消息生产者/消费者模式的原理 | 212 |
| 第 11 章 Redis 持久化 | 213 |
| 11.1 Redis 持久化操作概述 | 213 |
| 11.2 Redis 持久化机制 AOF | 214 |
| 11.2.1 AOF 持久化的配置 | 214 |
| 11.2.2 AOF 持久化的实现 | 215 |
| 11.2.3 AOF 文件重写 | 216 |
| 11.2.4 AOF 文件处理 | 220 |
| 11.2.5 AOF 持久化的优劣 | 221 |
| 11.3 Redis 持久化机制 RDB | 222 |
| 11.3.1 RDB 持久化 | 222 |
| 11.3.2 RDB 文件 | 224 |
| 11.3.3 RDB 文件的创建与加载 | 226 |
| 11.3.4 创建与加载 RDB 文件时服务器的状态 | 228 |
| 11.3.5 RDB 持久化的配置 | 228 |
| 11.3.6 RDB 持久化的优劣 | 229 |
| 11.4 AOF 持久化与 RDB 持久化抉择 | 230 |
| 第 12 章 Redis 集群 | 231 |
| 12.1 Redis 集群的主从复制模式 | 231 |
| 12.1.1 什么是主从复制 | 231 |
| 12.1.2 主从复制配置 | 234 |
| 12.1.3 复制功能的原理 | 237 |
| 12.1.4 复制功能的实现步骤 | 242 |
| 12.1.5 Redis 读写分离 | 245 |
| 12.1.6 Redis 心跳机制 | 246 |
| 12.2 Redis 集群的高可用哨兵模式 | 247 |
| 12.2.1 什么是高可用哨兵模式 | 248 |

| | | |
|-----------------------|------------------------------------|------------|
| 12.2.2 | 哨兵模式的配置 | 249 |
| 12.2.3 | Sentinel 的配置选项 | 255 |
| 12.2.4 | 哨兵模式的实现原理 | 256 |
| 12.2.5 | 选择“合适”的 slave 节点作为 master 节点 | 263 |
| 12.2.6 | Sentinel 的下线状态 | 266 |
| 12.2.7 | Sentinel 内部的定时任务 | 267 |
| 12.3 | Redis 集群搭建 | 268 |
| 12.3.1 | 什么是 Redis 集群 | 268 |
| 12.3.2 | 集群中的节点和槽 | 269 |
| 12.3.3 | 集群搭建 | 274 |
| 12.3.4 | 使用 Redis 集群 | 285 |
| 12.3.5 | 集群中的错误 | 287 |
| 12.3.6 | 集群的消息 | 289 |
| 第 13 章 | Redis 高级功能 | 291 |
| 13.1 | 慢查询 | 291 |
| 13.1.1 | 配置慢查询 | 291 |
| 13.1.2 | 慢查询的生命周期 | 293 |
| 13.1.3 | 慢查询日志 | 294 |
| 13.1.4 | 慢查询命令 | 296 |
| 13.2 | 流水线 | 297 |
| 13.2.1 | 什么是 Pipeline 技术 | 297 |
| 13.2.2 | 如何使用 Pipeline 技术 | 298 |
| 13.3 | 地理位置的应用 | 298 |
| 13.3.1 | 存储地理位置 | 298 |
| 13.3.2 | 获取地理位置的经纬度信息 | 299 |
| 13.3.3 | 计算两地间的距离 | 300 |
| 13.3.4 | 获取指定范围内的位置信息 | 300 |
| 13.4 | 位图 | 302 |
| 13.4.1 | 二进制位数组 | 302 |
| 13.4.2 | 位数组的表示 | 304 |
| 13.4.3 | 位数组的实现 | 305 |
| 第三部分 Redis 实战篇 | | |
| 第 14 章 | Java 操作 Redis | 310 |
| 14.1 | Java 客户端 Jedis | 310 |

| | | |
|---------------|------------------------------------|------------|
| 14.1.1 | Jedis 的获取 | 310 |
| 14.1.2 | Jedis 的使用 | 311 |
| 14.1.3 | Jedis 常用 API | 311 |
| 14.1.4 | Jedis 事务 | 313 |
| 14.1.5 | Jedis 主从复制 | 316 |
| 14.1.6 | Jedis 的连接池 | 318 |
| 14.2 | Java 操作 Redis 数据类型 | 321 |
| 14.2.1 | Java 操作 Redis 字符串类型 | 322 |
| 14.2.2 | Java 操作 Redis 列表类型 | 323 |
| 14.2.3 | Java 操作 Redis 集合类型 | 325 |
| 14.2.4 | Java 操作 Redis 哈希表类型 | 326 |
| 14.2.5 | Java 操作 Redis 有序集合类型 | 328 |
| 14.3 | Java 操作 Redis 实现排行榜 | 329 |
| 14.4 | Java 操作 Redis 实现秒杀功能 | 332 |
| 14.5 | Java 操作 Redis 实现消息队列 | 335 |
| 14.6 | Java 操作 Redis 实现故障转移 | 338 |
| 第 15 章 | SpringBoot 操作 Redis | 343 |
| 15.1 | 在 SpringBoot 中应用 Redis | 343 |
| 15.1.1 | Redis 依赖配置 | 343 |
| 15.1.2 | Redis 配置文件 | 344 |
| 15.2 | SpringBoot 连接 Redis | 345 |
| 15.3 | SpringBoot 整合 Redis 实现缓存 | 352 |
| 第 16 章 | Python 操作 Redis | 364 |
| 16.1 | 在 Python 中应用 Redis | 364 |
| 16.1.1 | 在 PyCharm 中配置 Redis | 364 |
| 16.1.2 | Python 连接 Redis | 365 |
| 16.2 | Python 操作 Redis 数据类型 | 367 |
| 16.2.1 | Python 操作 Redis String 类型 | 367 |
| 16.2.2 | Python 操作 Redis List 类型 | 370 |
| 16.2.3 | Python 操作 Redis Set 类型 | 372 |
| 16.2.4 | Python 操作 Redis Hash 类型 | 374 |
| 16.2.5 | Python 操作 Redis SortedSet 类型 | 376 |
| 16.2.6 | Python 操作 Redis 的其他 key | 378 |
| 16.3 | Python 操作 Redis 实现消息订阅发布 | 380 |

第一部分 Redis 初始篇

- ▶ 第 1 章 初识 NoSQL
- ▶ 第 2 章 认识 Redis
- ▶ 第 3 章 Redis 数据类型
- ▶ 第 4 章 Redis 必备命令
- ▶ 第 5 章 Redis 数据库

第 1 章

初识 NoSQL

相信各位读者都知道数据库的概念，即用来存储数据的仓库。本书主要讲解数据库中的 NoSQL。与 NoSQL 接触，大家是不是很高兴呢？不用着急，本章将为大家讲述什么是 NoSQL、NoSQL 的使用场景、数据模型及 NoSQL 的分类。通过对本章的学习，读者将会更加清晰地了解 NoSQL，为以后的工作、学习奠定基础。话不多说，就让我们与 NoSQL 亲密接触吧！

1.1 什么是 NoSQL

NoSQL 不仅仅是 SQL，它是 Not Only SQL 的缩写，也是众多非关系型数据库的统称。NoSQL 和关系型数据库一样，也是用来存储数据的仓库。

为什么需要使用 NoSQL？

随着互联网的高速发展，数据量、访问量呈爆发式增长，人们对网络的需求逐渐多样化。比如，通过 QQ、微信、微博等进行聊天互动，刷朋友圈，点赞，互评；又如，通过各大视频网站、音乐网站看视频、看直播、听音乐等，这么多数据都是需要存储的。然而，传统的关系型数据库面对这些海量数据的存储，以及实现高访问量、高并发读/写，就会显得力不从心，尤其是当面对超大规模、高并发、高吞吐量的大型动态网站的时候，就会暴露出很多难以克服的问题，影响用户体验。为了满足对海量数据的高速存储需求，实现高并发、高吞吐量，NoSQL 应运而生。NoSQL 的出现可以解决传统关系型数据库所不能解决的问题。

1. NoSQL 的出现解决了高并发读/写问题

Web 2.0 动态网站需要根据用户的个性化信息来实时生成动态页面和提供动态信息，而无法使用动态页面的静态化技术，因此数据库的并发负载就会非常高。比如，微博、朋友圈的实时更新，就会出现每秒上万次的读/写需求。关系型数据库在面对每秒上万次的 SQL 查询操作时还能应对自如，但是在面对每秒上万次的 SQL 写操作时就难以胜任了。普通的 BBS 系统网站也存在高并发读/写的需求，比如，实时统计在线人数、记录热门帖子的浏览次数等，当面对这些需求时，传统的关系型数据库就会出现大量问题。

2. NoSQL 的出现解决了海量数据的高效率存储和访问问题

面对实时产生的大数据量的存储与查询，关系型数据库是难以应付的，会显得效率非常低；而利用 NoSQL 的高效存储与查询能力，就能解决这个问题。

3. NoSQL 的出现实现了高可用性及高可扩展性

在基于 Web 的架构中，关系型数据库难以进行横向扩展。当一个网站系统的用户量和访问量与日俱增的时候，数据库没有办法像 Web 服务器或应用服务器那样通过添加更多的硬件来搭建负载均衡的服务器。对于很多提供 24 小时不间断服务的网站来说，对数据库系统的维护升级和扩展是非常折磨人的一件事，往往需要停机维护和数据迁移。

NoSQL 的出现解决了大规模数据库集中和数据种类不同所带来的各种问题，尤其是大数据实现的困难。

常见的 NoSQL 如图 1.1 所示。



图 1.1 常见的 NoSQL

NoSQL 具有如下特点：

- 容易扩展，方便使用，数据之间没有关系。
- 数据模型非常灵活，无须提前为要存储的数据建立字段类型，随时可以存储自定义的数据格式。
- 适合大数据量、高性能的存储。
- 具有高并发读/写、高可用性。

1.2 NoSQL 与传统关系型数据库的比较

相信大家传统关系型数据库都不陌生，我们常常使用的关系型数据库有 MySQL、Oracle、SQL Server、SQLite、DB2、Teradata、Infomix、Sybase、PostgreSQL、Access、FoxPro 等。

我们将通过以下几个方面来比较 NoSQL 与传统关系型数据库。

1. 使用成本

NoSQL: NoSQL 使用简单，易搭建，大部分是开源软件，比较廉价，任何人都可以使用。

关系型数据库：相对于 NoSQL，关系型数据库通常需要安装部署，开源的比较少，使用成本比较昂贵。尤其是 Oracle 数据库，需要花费大量资金购买，使用成本比较高。

2. 存储形式

NoSQL：NoSQL 具有丰富的存储形式，如 key-value（键值对）形式、图结构形式、文档形式、列簇形式等，因此，它可以存储各种类型的数据。

关系型数据库：关系型数据库是采用关系型数据模型来组织的，它是行列表结构，通过行与列的二元形式表示出来，数据之间有很强的关联性。它采用二维表结构的形式对数据进行持久存储。

3. 查询速度

NoSQL：NoSQL 将数据存储于系统的缓存中，不需要经过 SQL 层的解析，因此查询效率很高。

关系型数据库：关系型数据库将数据存储于系统的硬盘中，在查询的时候需要经过 SQL 层的解析，然后读入内存，实现查询，因此查询效率较低。

4. 扩展性

NoSQL：NoSQL 去掉了传统关系型数据库表与字段之间的关系，实现了真正意义上的扩展。它采用键值对的形式存储数据，消除了数据之间的耦合性，因此易扩展。

关系型数据库：由于关系型数据库采用关系型数据模型来存储数据，数据与数据之间的关联性较强，存在耦合性，因此不易扩展。尤其是存在多表连接（join）查询机制的限制，使得扩展很难实现。

5. 是否支持 ACID 特性

ACID 特性是指数据库事务的执行要素，包括原子性、一致性、隔离性、持久性。

NoSQL：NoSQL 一般不支持 ACID 特性，它实现最终一致性。

关系型数据库：关系型数据库支持 ACID 特性，具有严格的数据一致性。

6. 是否支持 SQL 语句

NoSQL：SQL 语句在 NoSQL 中是不被支持的，NoSQL 没有声明性查询语言，且没有预定义的模式。

关系型数据库：关系型数据库支持 SQL 语句，也支持复杂查询。SQL 是结构化查询语言、数据操纵语言、数据定义语言。

NoSQL 与传统关系型数据库是互补的关系，对方的劣势就是自己的优势，反之亦然。

1.3 在什么应用场景下使用 NoSQL

NoSQL 的应用场景比较广泛，下面简单说一下比较适合使用 NoSQL 的几个场景。