第5章

文档数据库MongoDB的原理与应用

5.1 MongoDB 简介

5.1.1 概述

MongoDB 是一个文档数据库，由 C++语言编写。它将数据存储为类似于 JSON 的文档，数据结构由键-值(key-value)对组成，可以存储比较复杂的数据类型，字段值除支持基础的数字、字符串、布尔值外，还支持数组、文档子对象等。

【名词概念解释】

- 键-值对：键(Key)通常又被称为关键字，每个关键字只出现一次；值(Value)则为该关键字对应的值(可以为单一数值，也可为数组、对象等)，这样就构成了一对数值关系，从而被称为键-值对。若读者仍不了解，可学习哈希表这一数据结构。
- JSON：可简单理解为由若干键-值对组成的字符串。
- 文档：类似于 JSON 对象，也可以存储若干键-值对。

5.1.2 特点

MongoDB 的特点为高性能、易部署、易使用、存储数据非常方便。其主要特点有以下 6 点。

(1) 数据类型丰富：MongoDB 将数据存储为 BSON(一种 JSON 的扩展)文档，与传统关系数据库中由行和列组成的表相比，BSON 支持更丰富、更灵活的数据结构。除支持基础的数字、字符串、布尔值等字段类型外，BSON 字段还可以是数组或嵌套子对象。

(2) 结构灵活：MongoDB 无须像关系数据库一样提前声明文档结构，字段类型可以灵活变化，同时也不存在模式，用户可以灵活地插入不同类型的文档等。

(3) 性能优越：MongoDB 支持完全索引，包括内部对象，即可以直接对文档中的任何对象进行索引检索。同时实验证明，对于大量数据，索引字段查询不会比关系数据库(如 MySQL)查询慢，非索引字段查询相对于关系数据库则全面胜出。

(4) 可扩展性强：文档被视为单个单元，这意味着它可以分布在多个服务器上，因此可以轻松地扩展，实现分布式的数据分布。

(5) 地理位置索引：MongoDB 除支持传统关系数据库中的索引外，还支持特有的地理位置索引。

（6）支持多种编程语言：支持 Java、C++、Python、C# 等多种编程语言。

5.1.3　发展历程

2007 年 10 月，Dwight Eliot 和 Kevin Ryan 一同创办了名为 10gen 的公司，在研发过程中，他们每次都在解决同样的数据水平扩展性问题。于是 MongoDB 在此背景下诞生，通过自动分片构建大型 MongoDB 集群，从而实现水平伸缩。2009 年 2 月，MongoDB 首次在数据库领域亮相，至此 MongoDB 1.0 发布。

2012 年 5 月，MongoDB 2.1 开发分支发布。该版本采用全新架构，包含诸多功能增强。

2012 年 6 月，MongoDB 2.0.6 发布，分布式文档数据库诞生。

2013 年 4 月，MongoDB 2.4.3 发布，此版本包括一些性能优化、功能增强以及 bug 修复。

2015 年 3 月，MongoDB 3.0 发布，包含新的存储引擎，大幅提升了 MongoDB 的写入性能。

2017 年 10 月，MongoDB 公司成立 10 周年之际，顺利上市，成为 26 年来第一家以数据库产品为主要业务的上市公司。

2018 年 6 月，MongoDB 4.0 发布，提供跨文档事务处理能力。

2019 年 8 月，MongoDB 4.2 发布，开始支持分布式事务。

至今，MongoDB 已经从一个在数据库领域籍籍无名的“小透明”，变成了话题度和热度都很高的“流量”数据库。MongoDB 数据库平台下载量超过 1.25 亿次，MongoDB 客户遍布全球 100 多个国家和地区，其中思科使用 MongoDB 取代 Oracle 重构电商平台，字节跳动也将部分 MySQL 应用迁移到 MongoDB 上来，如与地理相关的查询、游戏运营日志等业务。

【名词概念解释】

- 水平扩展性：连接多个软硬件特性，从而将多个服务器从逻辑上看成一个整体，进而提高性能。水平扩展性最大的优点是不需要单一机器具有极高的性能。
- 垂直扩展性：通过对单一物理实体增加资源而提高性能。

5.1.4　应用场景

基于 MongoDB 支持的数据类型丰富、结构灵活、性能优越和可扩展性强等特点，MongoDB 具体在以下 6 个领域得以应用。

（1）**游戏应用**：游戏应用需求灵活多变，MongoDB 结构灵活、可扩展性强等特点能够很好地解决游戏应用中的痛点。可以使用 MongoDB 存储游戏用户信息，用户的装备、积分等直接以内嵌文档的形式存储，方便查询、更新。

（2）**移动应用**：MongoDB 支持二维空间索引，可以高效地查询地理位置关系和检索用户地理位置数据，很好地支撑基于地理位置查询的移动类 App 的业务需求。同时，MongoDB 动态模式存储方式非常适合存储多重系统的异构数据，满足移动 App 应用的需求。

（3）**电商应用**：对于商品信息、订单信息，借助 MongoDB 支持的数据类型丰富等特点，通过嵌套的子文档，一次简单查询就能获取所需的信息。例如，上衣和裤子有相同的属性，如产地、价格、材质、颜色等；还有各自独有的属性，如上衣独有的肩宽、胸围、袖长等，裤子独有的臀围、脚口、长度等。独有属性可以直接以 BSON 子文档方式嵌套到商品这个文档中，一次查询直接获取全部内容，不需要进行多表 join 查询。

（4）**物流应用**：在电商配套的物流领域，使用 MongoDB 存储订单信息，订单状态在运送过程中会不断更新，以 MongoDB 内嵌数组的形式来存储，一次查询就能将订单所有的变更读

取出来。可以将一个快递的物流信息直接嵌套在以商品 Id 为唯一索引的文档中，一次查询就可以获取完整的快递流向信息。

（5）**视频直播**：视频直播行业会产生大量的礼物信息、用户聊天信息等，数据量较大。MongoDB 水平可扩展的优点能够很好地解决上述问题。

（6）**社交应用**：使用 MongoDB 存储用户信息，以及用户发表的朋友圈信息，通过 MongoDB 特有的地理位置索引特点，实现附近的人、地点等功能。

5.2 MongoDB 的相关基本概念

关系数据库中涉及数据库、表、行、列等基本概念，MongoDB 中同样涉及相关的概念，但是却存在相应的差异。为便于理解，下面将对比关系数据库的基本概念，对 MongoDB 中相关的基本概念进行介绍。MongoDB 数据库与关系数据库相关概念的对比如表 5-1 所示。

表 5-1 MongoDB 数据库与关系数据库相关概念的对比

关系数据库	MongoDB 数据库
数据库(Database)	数据库(Database)
表(Table)	集合(Collection)
行(Row)	文档(Document)
列(Column)	字段(Field)

5.2.1 命名规则

MongoDB 关于数据库名、文档名、字段键名等存在统一的命令规则，同时也存在一些不同。MongoDB 统一命名规则有以下 2 点：

（1）不能为空字符串。

（2）不得有空格、$ 、\和\0(空字符)(其中，空字符表示结尾除外)。

同时，还存在特殊的命名规则，具体为以下 4 点：

（1）数据库名和文档键中不得有点('.')，只能在特定环境中使用。

（2）数据库名中的字母必须全为小写字母。

（3）集合名不能以“system.”开头，因为这是为系统集合保留的前缀，同时用户创建的集合名中不能含有保留字符。

（4）以下画线“_”开头的键为系统保留的键。

5.2.2 数据库

MongoDB 中的库类似于传统关系数据库中库的概念，用不同的库来隔离不同的应用数据。MongoDB 中可以建立多个数据库，每个数据库都有自己的集合和权限，不同的数据库放置在不同的文件中。在操作过程中，若不指定数据库，则 MongoDB 默认操作的数据库为 test。

1. MongoDB 默认数据库

MongoDB 中默认包含 admin、local、config 三个数据库，可以直接访问。这三个数据库的含义如下。

（1）admin：从权限角度看，可理解为 root 数据库。若将一个用户添加至该数据库，则此

用户自动继承所有数据库的权限。一些特定的服务器端命令只能通过该数据库运行，例如列出所有的数据库或关闭服务器。

（2）local：该数据库永远不会被复制，可用来存储限于本地单服务器的任何集合。

（3）config：当 MongoDB 用于分片设置时，该数据库在内部使用，用来保存分片相关信息。

5.2.3　集合

集合就是 MongoDB 的文档组，类似于关系数据库管理系统中的表格。集合存在于数据库中，同时集合没有固定的结构，这意味着可以对集合插入不同格式和类型的数据，但通常情况下插入集合的数据都会存在一定的关联性。

使用 MongoDB 可以自定义集合，当第一个文档插入时，集合就会被创建。

5.2.4　文档

文档是若干键-值对（BSON）。MongoDB 的文档不需要设置相同的字段，并且相同的字段不需要相同的数据类型，这与传统关系数据库存在很大的区别，也是 MongoDB 非常突出的特点之一。

一个简单的文档例子为：{"name"："小明","age"：21,"major"："计算机科学与技术"}。

文档中还存在以下 5 点注意事项：

（1）文档中的键-值对绝对是有序的。

（2）文档中的值不仅可以是在双引号里面的字符串，还可以是其他几种数据类型（甚至可以是整个嵌入的文档）。

（3）区分数据类型和大小写。

（4）文档中不能出现重复的键。

（5）文档中的键是字符串。除少数例外情况外，键可以使用任意 UTF-8 字符。

5.2.5　MongoDB 的数据类型

MongoDB 的主要数据类型如表 5-2 所示。

表 5-2　MongoDB 的主要数据类型

数 据 类 型	描　　述
String	字符串。存储数据常用的数据类型。在 MongoDB 中，只有 UTF-8 编码的字符串才是合法的
Integer	整型数值。根据采用的服务器，可分为 32 位或 64 位
Boolean	布尔值。用于存储布尔值（真/假）
Double	双精度浮点数。MongoDB 中默认浮点数存储格式为双精度浮点数
Array	用于将数组、列表或多个值存储为一个键或值
Object	用于内嵌文档
Null	用于创建空值
Date	日期时间。可以指定日期时间：创建 Date 对象，传入年、月、日信息
ObjectId	对象 ID。用于创建文档的 ID

下面重点说明 ObjectId，ObjectId 类似于唯一主键，可以很快地生成和排序，包含 12 字节，分别表示的含义如下：

（1）前 4 字节表示创建时间戳，默认为格林尼治时间（UTC 时间）。

（2）接下来 3 字节是机器标识码。

（3）紧接着 2 字节表示 PID(由进程 ID 组成)。

（4）最后 3 字节是随机数。

ObjectId 各字节的具体含义如图 5-1 所示。

0	1	2	3	4	5	6	7	8	9	10	11
时间戳				机器			PID		计数器		

图 5-1　ObjectId 各字节的具体含义

MongoDB 中存储的文档必须有一个_id 键。这个键的值可以是任何类型，若不特殊指定，则默认为一个 ObjectId 对象，例如"_id":ObjectId("620e07e539f098ae9e114fd3")。

5.3　MongoDB 的安装和配置方法

为便于读者理解与掌握，本节将引入一个具体实例(情节均为虚构)，以实例驱动，讲解 MongoDB 的安装与具体的使用细节。

目前，已有大部分有志青年放弃大城市的高薪工作，开始服务于农村家乡，借助互联网平台，以亲身经历教会父老乡亲出售本地农产品特产，提高家乡人民的生活水平和生活质量。某高校计算机专业的学生小波作为一名资深的“剁手党”，经常在各大电商平台购物，也希望能够通过专业知识为农村发展贡献自己的微薄之力，他深知传统关系数据库对数据有着严格的规范，如建表前需要严格指定数据类型等，这对初创公司不太友好。

而 MongoDB 以文档结构存储数据，面向集合存储，易存储对象类型的数据。同时，MongoDB 支持快速查询以及动态查询，能够高效地获取数据信息，且可扩展性较强，方便后续加大数据规模，并且商品信息、订单信息的事务性要求也不是很高，使用 MongoDB 搭建农产品电商数据库(agr_products_ecommerce_db)当然是一个不错的选择。于是小波开始发挥他的专业特长，着手设计农产品电商数据库，希望未来能够有机会服务于农村、农民。

数据库设计的第一步是下载 MongoDB 数据库。MongoDB 支持主流操作系统，包括 Windows、macOS、Linux 等。用户只需存在相应的存储空间，便可下载并安装使用，对软硬件的要求极低。

MongoDB 官方网站(网址详见前言二维码)提供了很多版本，包括企业版、社区版等。为了便于理解与使用，本节所有操作均在 Windows 操作系统的社区版下进行(社区版下载网址详见前言二维码)。编写时，MongoDB 社区版已更新至 5.0.6 版本。前面的章节已经介绍了基于 Docker 方式安装的优点，本节除介绍传统方式安装配置外，还将介绍基于 Docker 方式的安装配置方法。

5.3.1　传统方式的安装与配置

下载相应版本的安装包，直接双击安装包，即可进行具体的安装操作。操作过程较简单，可以单击 Custom 按钮，设置安装目录。自定义设置安装目录，如图 5-2 所示。

选择安装路径，如图 5-3 所示。

随后会显示数据库 Data 目录和 Log 目录的位置，默认为安装目录下的 data 和 log 子文件夹。选择 Data 和 Log 目录安装路径，如图 5-4 所示。

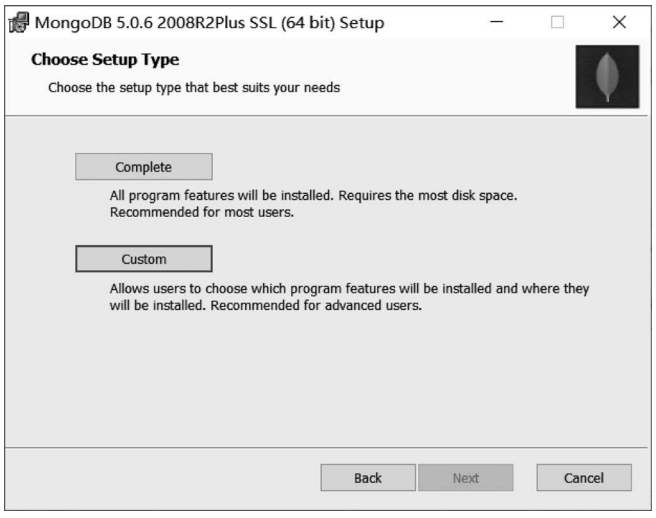

图 5-2　自定义设置安装目录

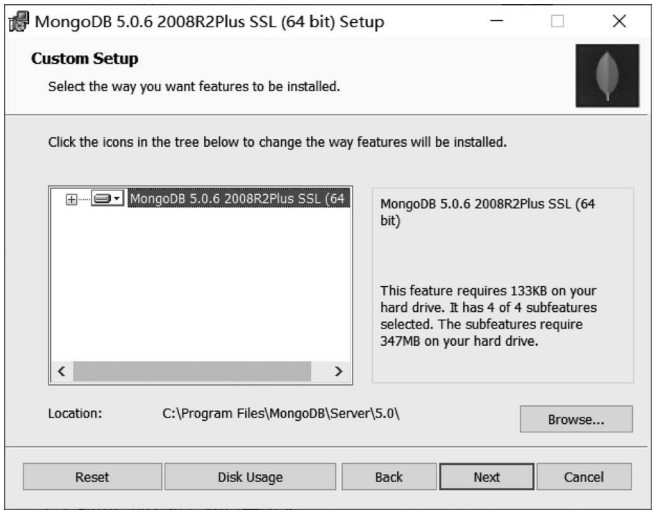

图 5-3　选择安装路径

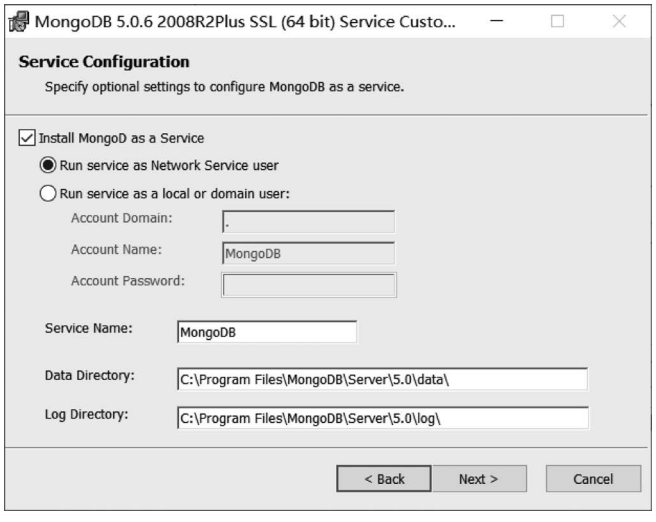

图 5-4　选择 Data 和 Log 目录安装路径

接下来会提示是否安装 MongoDB Compass 工具(安装过程可能较慢,后续也可在 MongoDB 官方网站自行下载,离线安装)。MongoDB Compass 是一个图形化界面管理工具,后续将具体介绍。安装 MongoDB Compass 工具,如图 5-5 所示。

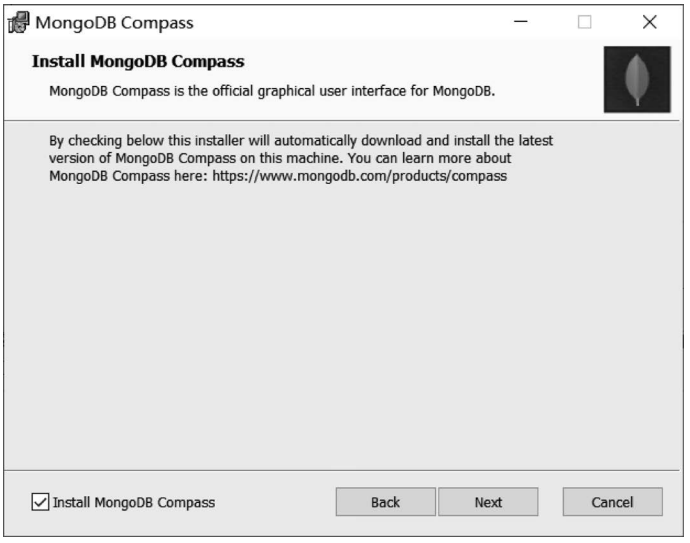

图 5-5　安装 MongoDB Compass 工具

安装成功后,根目录下存在 bin、data 和 log 三个子文件夹。

(1) bin 文件夹：存放 MongoDB 相关工具组件,如 mongod.exe(服务器应用程序)、mongo.exe(客户端应用程序)等。

(2) data 文件夹：存储 MongoDB 数据信息。

(3) log 文件夹：存储 MongoDB 日志信息。

在 bin 文件夹中存在 mongod.cfg 文件,用来存储 MongoDB 配置信息,默认有 storage(数据存储信息)、systemLog(系统日志信息)和 net(默认 IP 和端口信息),具体信息如下。

(1) storage：默认数据库存储位置为 data 文件夹。

(2) systemLog：系统日志信息默认存储至 log 文件夹的 mongod.log 文件中,且以追加的方式写入日志信息。

(3) net：默认 IP 为 127.0.0.1,默认端口为 27017。

读者可以根据自己的需要修改 mongod.cfg 文件的相应配置信息,也可以在命令行修改当前连接的配置信息,具体后续将进行介绍。

5.3.2　基于 Docker 方式的安装与配置

Windows Docker 的安装在第 3 章已详细介绍。本小节将介绍 Docker 方式的 MongoDB 的安装与配置。

首先拉取 MongoDB 镜像。笔者撰写本书时 MongoDB 社区版已更新至 5.0.6 版本,于是拉取 MongoDB 镜像的具体命令为：

```
docker pull mongo:5.0.6
```

安装完成后,输入 docker images 指令即可查看 Docker 中的所有镜像信息,并验证 MongoDB 是否拉取成功。查看 Docker 本地镜像信息,如图 5-6 所示。

```
PS C:\Program Files\MongoDB> docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
mongo        5.0.6     532c84506200   3 months ago   699MB
```

图 5-6　查看 Docker 本地镜像信息

5.3.3　MongoDB 的连接

MongoDB 分为服务端和客户端，只有启动 MongoDB 服务端后，客户端才可连接 MongoDB，进一步进行数据库、集合和文档的操作。

1. MongoDB 服务启动

基于传统方式安装的 MongoDB，启动服务主要有以下 3 种方法：

(1) 在 Windows 任务管理器中的服务栏，找到 MongoDB 对应的服务打开即可(默认为开机自动打开)。

(2) 在 bin 目录路径下打开 cmd(命令提示符)窗口，直接输入 mongod 命令也可打开 MongoDB 服务，当然也可输入其他配置信息，如 mongod --dbpath＝/data --port＝27017(自定义数据库目录和端口信息)。

(3) 输入 net start mongodb 命令，由于安装过程中 MongoDB 已加入 Windows 服务中，因此可直接在 cmd 窗口输入 net start mongodb 命令打开服务。

基于 Docker 方式安装的 MongoDB 启动服务较为简单，具体命令如下：

```
docker run -d --name mongo -p 27017:27017 mongo:5.0.6
```

【参数说明】

- -d：表示后台运行。
- --name：为容器分配一个名称。
- -p：向主机发布容器端口集合。

2. MongoDB 客户端连接

基于传统方式安装的 MongoDB，客户端连接方法主要有以下 2 种：

(1) 直接运行 mongo.exe。

(2) 在 bin 文件夹下打开 cmd 窗口输入 mongo 命令(也可指定连接的配置信息，如端口、日志信息等)。

基于 Docker 方式安装的 MongoDB，客户端连接的具体命令如下：

```
docker exec -it mongo mongo
```

【参数说明】

- 第一个 mongo 表示容器名，第二个 mongo 表示镜像名。

5.4　MongoDB 的基本操作

5.4.1　数据库的创建

通过 5.3 节的学习，小波已成功安装 MongoDB。接下来开始创建农产品电商数据库(agr_

products_ecommerce_db)。MongoDB 创建数据库的语法格式为：use database_name(数据库名)。其中，use 代表创建并使用，若数据库不存在，则自动创建数据库。

小波已迫不及待开始操作。创建农产品电商数据库(agr_products_ecommerce_db)，代码如下：

```
> use agr_prducts_ecommerce_db
switched to db agr_prducts_ecommerce_db
```

5.4.2 数据库的查询与删除

1. 数据库的查询

数据库查询指令主要有 2 种，分别为 show dbs 和 db。

1) show dbs 指令

查询 MongoDB 中的所有数据库，但若数据库为空数据库，则不会被查询显示。

小波小试牛刀，想要证实刚刚创建的农产品电商数据库(agr_products_ecommerce_db)的确没有任何数据信息。查询所有数据库，如图 5-7 所示。因为刚刚创建的农产品电商数据库没有数据，是空数据库，所以并未显示。

2) db 指令

查询当前所在数据库，如图 5-8 所示。可以发现，此时所在数据库的确为小波刚刚创建的农产品电商数据库(agr_products_ecommerce_db)。

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
```

图 5-7 查看所有数据库

```
> db;
agr_products_ecommerce_db
```

图 5-8 查询当前所在数据库

2. 数据库的删除

数据库删除指令为 db.dropDatabase()，表示删除当前所在的数据库。例如，删除 agr_products_ecommerce_db 数据库，如例 5-1 所示。

【例 5-1】 删除 agr_products_ecommerce_db 数据库

```
> use agr_products_ecommerce_db
switched to db agr_products_ecommerce_db
> db.dropDatabase();
{"ok": 1}
```

5.4.3 集合的创建

相比于传统关系数据库中的库由若干表组成，MongoDB 数据库由若干集合组成。

集合创建指令为 db.createCollection(name,options)。具体参数说明如下。

- name：必选参数，创建集合的名称。
- options：可选参数，指定有关内存大小及索引。

options 参数表示如表 5-3 所示。

表 5-3　options 参数表示

字　段	类　型	描　述
capped	布尔	如果该值为 true，则创建固定集合。固定集合是指有着固定大小的集合，当达到最大值时，它会自动覆盖最早的文档。该值为 true 时，必须指定 size 参数
size	数值	为固定集合指定一个最大值，即字节数
max	数值	指定固定集合中包含文档的最大数量

小波开始设计农产品电商数据库(agr_products_ecommerce_db)所需的集合结构，其中不可或缺的便是商品集合(products)。创建包含文档最大数量为 100、集合空间最大值为 5000 字节的 products 集合，如例 5-2 所示。

【例 5-2】　创建包含文档最大数量为 100、集合空间最大值为 5000 字节的 products 集合

```
> db.createCollection("products",{max:100,capped:true,size:5000});
{"ok": 1}
```

当然，MongoDB 中也可以不需要创建集合。当你插入一些文档时，MongoDB 会自动创建集合，读者可自行操作。

5.4.4　集合的查询与删除

1. 集合的查询

集合查询指令为 show collections 或 show tables。查询当前数据库(agr_products_commerce_db)的所有集合，如例 5-3 所示。

【例 5-3】　查询当前数据库(agr_products_commerce_db)的所有集合

```
> show collections;
products
> show tables;
products
```

2. 集合的删除

集合删除指令为 db.collection.drop()。具体参数说明如下。

- collection 表示需要删除的集合名。若成功删除选定集合，则 drop()方法返回 true，否则返回 false。

例如，删除 products 集合，如例 5-4 所示。

【例 5-4】　删除 products 集合

```
> db.products.drop();
true
```

5.4.5　文档添加

MongoDB 为文档数据库，由一个个文档组成，所以文档的操作为 MongoDB 的核心，具体可分为文档插入指令、文档查询指令、文档更新指令、文档删除指令。本小节将重点介绍文档的添加操作。

1. MongoDB 的文档格式

添加文档之前，需要简单了解文档的格式。MongoDB 的文档为 BSON 格式（一种类似于 JSON 的格式），由若干键-值对组成。其中 key、value 均可为任何数据类型格式，包括文档子对象。MongoDB 文档示例如例 5-5 所示。

【例 5-5】 MongoDB 文档示例

```
{
        "_id": ObjectId("62d6c1443064879e0bad6d86"),
        "product_name": "西红柿",
        "variety_name": "普罗旺斯番茄",
        "producing_area": "山东省海阳市",
        "price": 3,
        "unit": "斤",
        "shipping_address": "山东省海阳市"
}
```

2. MongoDB 文档添加操作

简单导入农产品信息，通过命令行有两种插入数据形式，分别为 insert（插入单个文档）和 insertMany（插入多个文档）。插入单个文档如例 5-6 所示。

【例 5-6】 插入单个文档

```
> db.products.insert({
        product_name: "胡萝卜",
        variety_name: "秤杆红萝卜",
        producing_area: "陕西省渭南市大荔县",
        leaf_length: "15cm 以上",
        price: 0,
        unit: "斤",
        shipping_address: "陕西省渭南市大荔县"
})
WriteResult({ "nInserted" : 1 })
```

插入多个文档如例 5-7 所示。除使用 insertMany 指令插入多个文档外，也可通过文件形式导入多个文档内容，具体操作将在 5.7.2 节进行介绍。

【例 5-7】 插入多个文档

```
db.products.insertMany(
[{
    product_name: "菠菜",
    variety_name: "大叶菠菜",
    producing_area: "山东省聊城市",
    leaf_length: "25 - 30cm",
    price: 0.6,
    unit: "斤",
    shipping_address: "山东省聊城市东昌府区"
},
{
    product_name: "山药",
    variety_name: "河北铁棍山药",
    producing_area: "河北省保定市",
```

```
    specifications: {
    "家庭实惠装 - 长 30 - 40cm - 5 斤装 二级,5.00 斤/箱": 24,
    "优良装.长 45 - 50cm.5 斤装 二级,5.00 斤/箱": 26,
    "优选装 - 长 55 - 60cm - 5 斤装 一级,5.00 斤/箱": 28,
    "精选装 - 长 65 - 70cm - 5 斤装 一级,5.00 斤/箱": 30
    },
    unit: "斤",
    shipping_address: "河北省保定市蠡县"
}]);
{
        "acknowledged" : true,
        "insertedIds" : [
              ObjectId("62d6c4d22fca6e0caacdefca"),
              ObjectId("62d6c4d22fca6e0caacdefcb")
        ]
}
```

3. 文档添加指令详细介绍

文档添加指令为 db.collection.insert()或 db.collection.insertMany()。其中 collection 表示插入文档所在的集合名。

在 MongoDB 中,每个文档都会有一个_id 作为唯一标识(类似于传统关系数据库的主键),_id 默认会自动生成,由 12 字节组成,具体含义 5.2.4 小节已介绍。若手动指定_id 的值,则需要确保当前集合中没有出现过此_id,否则会报错。

5.4.6　文档查询

1. 文档查询概述

在数据库操作中,查询操作最为普遍,而 MongoDB 为文档数据库,对于文档的查询更加常见。具体指令为 db.collection.find(query,projection)。其中 collection 表示查询的集合名。

参数说明如下。

- query:可选,使用查询操作符指定查询条件。
- projection:可选,使用投影操作符指定返回的键。若想要查询所有的键值,则只需省略该参数即可(默认省略)。

如果需要以易读的方式读取数据,可以使用 pretty()方法,语法格式为 db.collection.find().pretty()。例如,查询农产品电商数据库中 products 集合的所有数据,如例 5-8 所示。

【例 5-8】 查询农产品电商数据库中 products 集合的所有数据

```
> db.products.find().pretty();
{
        "_id" : ObjectId("62d6c1443064879e0bad6d86"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
```

```
{
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "specifications" : {
              "家庭实惠装 - 长 30 - 40cm - 5 斤装 二级,5.00 斤/箱" : 24,
              "优良装.长 45 - 50cm.5 斤装 二级,5.00 斤/箱" : 26,
              "优选装 - 长 55 - 60cm - 5 斤装 一级,5.00 斤/箱" : 28,
              "精选装 - 长 65 - 70cm - 5 斤装 一级,5.00 斤/箱" : 30
        },
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县"
}
```

MongoDB 和传统关系数据库一样，支持的查询操作相当丰富，例如条件查询、索引、聚合、排序、AND、OR、模糊查询等。MongoDB 文档查询类型如表 5-4 所示。接下来将对各个查询类型分别进行介绍。

表 5-4　MongoDB 文档查询类型

文档查询类型	说　　明
条件查询	条件判断查询，类似于传统关系数据库的 where 语句
AND、OR 查询	与传统关系数据库中的 AND、OR 查询类似
$ type 查询	通过数据类型查询
排序查询	使用 sort()方法对查询结果指定排序顺序
分页查询	使用 limit()和 skip()方法实现分页查询
索引查询	建立索引，并通过索引查询
聚合查询	主要用于处理数据(如统计平均值、求和、最值等)
其他查询	如数组中的查询、模糊查询、去重、指定返回字段等

2. 条件查询

条件操作符即为条件判断，类似于关系数据库中的 where 语句。首先将 MongoDB 数据

库中的各种条件操作符与相同含义的 where 语句进行对比，便于进一步理解，如表 5-5 所示。

表 5-5　MongoDB 条件操作符与传统关系数据库 where 语句对比

操　作	格　式	示　例	关系数据库中类似语句
等于	{<key>:<value>}	db.products.find({"product_name":"菠菜"})	where product_name ="菠菜"
小于	{<key>:{$lt:<value>}}	db.products.find({"price":{$lt:0.6}})	where price < 0.6
小于或等于	{<key>:{$lte:<value>}}	db.products.find({"price":{$lte:0.6}})	where price <= 0.6
大于	{<key>:{$gt:<value>}}	db.products.find({"price":{$gt:0.6}})	where price > 0.6
大于或等于	{<key>:{$gte:<value>}}	db.products.find({"price":{$gte:0.6}})	where price >= 0.6
不等于	{<key>:{$ne:<value>}}	db.products.find({"price":{$ne:0.6}})	where price ! = 0.6

在此将对农产品电商数据库中的 products 集合进行查询，以便更好地理解与掌握条件操作符。查询 products 集合中 product_name 为“菠菜”的数据，如例 5-9 所示。

【例 5-9】 查询 products 集合中 product_name 为“菠菜”的数据

```
> db.products.find({"product_name": "菠菜"}).pretty();
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

查询 products 集合中 price 小于 0.6 的数据，如例 5-10 所示。

【例 5-10】 查询 products 集合中 price 小于 0.6 的数据

```
> db.products.find({"price":{$lt:0.6}}).pretty();
{
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
```

查询 products 集合中 price 小于或等于 0.6 的数据，如例 5-11 所示。

【例 5-11】 查询 products 集合中 price 小于或等于 0.6 的数据

```
> db.products.find({"price":{$lte:0.6}}).pretty();
{
```

```
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

查询 products 集合中 price 大于 0.6 的数据，如例 5-12 所示。

【例 5-12】 查询 products 集合中 price 大于 0.6 的数据

```
> db.products.find({"price":{$gt:0.6}}).pretty();
{
        "_id" : ObjectId("62d6c1443064879e0bad6d86"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
```

查询 products 集合中 price 大于或等于 0.6 的数据，如例 5-13 所示。

【例 5-13】 查询 products 集合中 price 大于或等于 0.6 的数据

```
> db.products.find({"price":{$gte:0.6}}).pretty();
{
        "_id" : ObjectId("62d6c1443064879e0bad6d86"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
```

```
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

查询 products 集合中 price 不等于 0.6 的数据，如例 5-14 所示。

【例 5-14】 查询 products 集合中 price 不等于 0.6 的数据

```
> db.products.find({"price":{ $ne:0.6}}).pretty();
{
        "_id" : ObjectId("62d6c1443064879e0bad6d86"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
{
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "specifications" : {
                "家庭实惠装 - 长 30 - 40cm - 5 斤装 二级,5.00 斤/箱" : 24,
                "优良装.长 45 - 50cm.5 斤装 二级,5.00 斤/箱" : 26,
                "优选装 - 长 55 - 60cm - 5 斤装 一级,5.00 斤/箱" : 28,
                "精选装 - 长 65 - 70cm - 5 斤装 一级,5.00 斤/箱" : 30
        },
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县"
}
```

3. AND、OR 查询

MongoDB 中的 find()方法支持传入多个键，每个键以逗号隔开，即为关系数据库中的 AND 条件。语法格式为 db.collection.find({key1:value1,key2:value2})。

同样以农产品电商数据库中的 products 集合操作为例，查询 products 集合中 price 大于或等于 0.6 且 producing_area(产地)为“山东省聊城市”的数据，如例 5-15 所示。

【例 5-15】 查询 products 集合中 price 大于或等于 0.6 且 producing_area(产地)为“山东省聊城市”的数据

```
> db.products.find({"price": { $gte: 0.6},"producing_area":"山东聊城"}).pretty();
{
```

```
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

但是注意，如果 key 相同，则后面的条件会把前面的条件覆盖。若想要实现传统关系数据库中类似于 OR 的查询，则需要使用 OR 操作符，具体语法格式为 db.collection.find({$or:[{key1:value1},{key2:value2}]})。查询 products 集合中 product_name 为“菠菜”或 price 等于 0.5 的数据，如例 5-16 所示。

【例 5-16】 查询 products 集合中 product_name 为“菠菜”或 price 等于 0.5 的数据

```
> db.products.find({$or:[{"product_name":"菠菜"},{"price":0.5}]}).pretty();
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

同时也可实现 AND 和 OR 的联合查询，读者可自行选择数据进行实验。

3. $type 查询

$type 操作符是基于 BSON 类型来检索集合中匹配的数据类型，并返回结果。$type 查询可使用的类型如表 5-6 所示。

表 5-6 $type 查询可使用的类型

类　型	数　字	备　注
Double	1	
String	2	
Object	3	
Array	4	
Binary data	5	
Undefined	6	已废弃
Object id	7	
Boolean	8	
Date	9	
Null	10	
Regular Expression	11	
JavaScript	13	
Symbol	14	

续表

类　　型	数　　字	备　　注
JavaScript(with scope)	15	
32-bit Integer	16	
Timestamp	17	
64-bit Integer	18	
Min key	255	Query with -1
Max key	127	

查询 products 集合中 product_name 为 String 的数据，如例 5-17 所示。

【例 5-17】 查询 products 集合中 product_name 为 String 的数据

```
> db.products.find({"product_name":{ $type:'string'}}).pretty();
{
        "_id" : ObjectId("62d6c1443064879e0bad6d86"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
{
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25－30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "specifications" : {
              "家庭实惠装－长 30－40cm－5 斤装 二级,5.00 斤/箱" : 24,
              "优良装.长 45－50cm.5 斤装 二级,5.00 斤/箱" : 26,
              "优选装－长 55－60cm－5 斤装 一级,5.00 斤/箱" : 28,
              "精选装－长 65－70cm－5 斤装 一级,5.00 斤/箱" : 30
        },
        "unit" : "斤",
```

```
        "shipping_address" : "河北省保定市蠡县"
}
```

4. 排序查询

MongoDB 中可使用 sort()方法对数据进行排序，sort()方法可以通过参数指定排序的字段，并使用 1 和－1 指定排序的方式，其中 1 为升序排列，而－1 为降序排列。sort()方法的基本语法为 db.collection.find().sort({key:1})。products 集合中数据按字段 price 升序排序，如例 5-18 所示。

【例 5-18】 products 集合中数据按字段 price 升序排序

```
> db.products.find().sort({"price":1}).pretty();
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "specifications" : {
                "家庭实惠装－长 30－40cm－5 斤装 二级,5.00 斤/箱" : 24,
                "优良装.长 45－50cm.5 斤装 二级,5.00 斤/箱" : 26,
                "优选装－长 55－60cm－5 斤装 一级,5.00 斤/箱" : 28,
                "精选装－长 65－70cm－5 斤装 一级,5.00 斤/箱" : 30
        },
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县"
}
{
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25－30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
{
        "_id" : ObjectId("62d6c1443064879e0bad6d86"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
```

5. 分页查询

若需要读取指定数量的数据记录，MongoDB 提供了 limit()方法，limit()方法的基本语法为 db.collection.find().limit(number)。

参数说明如下：

- number：指定从 MongoDB 中读取的记录条数。

MongoDB 除提供了 limit()方法读取指定数量的数据外，还支持 skip()方法来跳过指定数量的数据，skip()方法的基本语法为 db.colletion.find().limit(number1).skip(number2)。

参数说明如下。

- number1：limit()方法中的参数。
- number2：表示跳过的记录条数。

结合 limit()和 skip()方法即可实现传统关系数据库中的分页查询。例如，只显示 products 集合中的第 3 条数据，如例 5-19 所示。

【例 5-19】 只显示 products 集合中的第 3 条数据

```
> db.products.find().limit(1).skip(2).pretty();
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

6. 索引查询

索引通常能够极大地提高查询的效率，如果没有索引，MongoDB 在读取数据时必须扫描集合中的每个文件并选取那些符合查询条件的记录。这种扫描全集合的查询效率是非常低的，特别在处理大量的数据时，查询需要花费几十秒甚至几分钟，这对网站的性能是非常致命的。

索引是特殊的数据结构，索引存储在一个易于遍历读取的数据集合中，索引是对数据库表中一列或多列的值进行排序的一种结构。从根本上说，MongoDB 中的索引与其他数据库系统中的索引类似。MongoDB 在集合层面上定义了索引，并支持对 MongoDB 集合中的任何字段或文档的子字段进行索引。MongoDB 提供了 createIndex()方法来创建索引。其基本语法格式为 db.collection.createIndex(key,options)。

参数说明如下。

- key：表示需要创建的索引字段(1 为指定按升序创建索引，－1 为指定按降序创建索引)。
- options：可选参数。

options 可选参数列表，如表 5-7 所示。

表 5-7 options 可选参数列表

参　数	类　型	说　明
background	Boolean	建索引过程会阻塞其他数据库操作，background 可指定以后台方式创建索引，即增加 "background" 可选参数。"background" 默认值为 false
unique	Boolean	建立的索引是否唯一。指定为 true 可创建唯一索引。默认值为 false
name	String	索引的名称。如果未指定，MongoDB 通过连接索引的字段名和排序顺序生成一个索引名称
dropDups	Boolean	3.0+版本已废弃。在建立唯一索引时是否删除重复记录，指定为 true 可创建唯一索引。默认值为 false
sparse	Boolean	对文档中不存在的字段数据不启用索引。这个参数需要特别注意，如果设置为 true 的话，在索引字段中不会查询出不包含对应字段的文档。默认值为 false
expireAfterSeconds	Integer	指定一个以秒为单位的数值，完成 TTL(Time To Live，生存时间)设定，设定集合的生存时间
v	Index Version	索引的版本号。默认的索引版本取决于 MongoDB 创建索引时运行的版本
weights	Document	索引权重值，数值的取值范围为 1～99 999，表示该索引相对于其他索引字段的得分权重
default_language	String	对于文本索引，该参数决定了停用词及词干和词器的规则的列表。默认为英语
language_override	String	对于文本索引，该参数指定了包含在文档中的字段名，语言覆盖默认的 language，默认值为 language

相关操作有如下 4 点。

(1) 查看集合所有索引：db.collection.getIndexes()。

(2) 查看集合索引大小：db.collection.totalIndexSize()。

(3) 删除集合所有索引：db.collection.dropIndexes()。

(4) 删除集合指定索引：db.collection.dropIndex("索引名称")。

索引还存在更加复杂的操作，如复合索引等。若读者感兴趣，可自行查阅相关资料。

7. 聚合查询

MongoDB 中的聚合(Aggregate)主要用于处理数据(如统计平均值、求和等)，并返回计算后的数据结果，类似于传统关系数据库中的 count(*)语句。常见的聚合表达式如表 5-8 所示。

表 5-8 常见的聚合表达式

表　达　式	说　明
$ sum	计算总和
$ avg	计算平均值
$ min	获取集合中所有文档对应值的最小值
$ max	获取集合中所有文档对应值的最大值
$ push	将值加入一个数组中，不会判断是否有重复的值
$ addToSet	将值加入一个数组中，会判断是否有重复的值，若相同的值在数组中已经存在，则不加入
$ first	根据资源文档的排序获取第一个文档的数据
$ last	根据资源文档的排序获取最后一个文档的数据

8. 其他特殊查询

除上述查询外，MongoDB还支持其他特殊查询，主要有数组中查询、模糊查询、去重、指定返回字段等。在MongoDB中可以使用正则表达式实现近似模糊查询功能。例如，查询products集合中products_name包含“菜”的数据，如例5-20所示。

【例5-20】 查询products集合中products_name包含“菜”的数据

```
> db.products.find({"product_name":/菜/}).pretty();
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

若想要在查询输出时只显示部分字段，MongoDB同样支持。基本语法格式为db.collection.find(query,{字段1:1,字段2:1})。

参数说明如下。

- query表示查询条件。
- 参数2中的1表示该字段返回，0表示不返回(若不指定_id字段是否显示，则默认为_id字段显示)。注意：1和0不能同时使用。

例如，查询products集合所有数据中的product_name字段，如例5-21所示。

【例5-21】 查询products集合所有数据中的product_name字段

```
> db.products.find({},{"product_name":1});
{ "_id" : ObjectId("62d6c1443064879e0bad6d86"), "product_name" : "西红柿" }
{ "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"), "product_name" : "胡萝卜" }
{ "_id" : ObjectId("62d6c4d22fca6e0caacdefca"), "product_name" : "菠菜" }
{ "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"), "product_name" : "山药" }
```

其他特殊操作，读者可自行查阅相关资料了解并实验。

5.4.7　文档更新

文档更新语法格式如下：

```
db.collection.update(query,update,{upset:<boolean>,multi:<boolean>,writeConcern:<document>})
```

参数说明如下。

- query：必选，查询条件。
- update：必选，待更新的对象和一些更新的操作符(如$)等。
- upset：可选，表示如果不存在update的记录，是否插入新的对象，true表示插入，false表示不插入，默认为false。
- multi：可选，MongoDB默认为false，表示只更新找到的第一条数据，如果为true，则表示对查询到的所有数据进行更新。

- writeConcern：可选，表示抛出异常的级别。

例如，修改 products 集合中所有 product_name 为“西红柿”的价格为 2.5，如例 5-22 所示。

【例 5-22】 修改 products 集合中所有 product_name 为“西红柿”的价格为 2.5

```
> db.products.update({"product_name":"西红柿"},{$set:{"price":2.5}},{multi:true});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

其他更新操作此处并不演示，读者可自行实验。

5.4.8 文档删除

文档删除语法格式如下：

```
db.collection.remove(query,{justOne:<boolean>,writeConcern:<document>})
```

参数说明如下。

- query：可选，查询条件。
- justOne：可选，如果设为 true 或 1，则只删除一个文档，如果不设置此参数或使用默认值 false，则删除所有匹配条件的文档。
- writeConcern：可选，抛出异常的级别。

例如，删除 products 集合中 product_name 为“西红柿”的文档数据，如例 5-23 所示。

【例 5-23】 删除 products 集合中 product_name 为“西红柿”的文档数据

```
> db.products.remove({"product_name":"西红柿"});
WriteResult({ "nRemoved" : 1 })
```

5.4.9 文档结构修改

MongoDB 作为文档数据库，除能像传统关系数据库一样实现基础的增、删、改、查外，还能轻松地实现修改文档字段、删除文档字段、添加文档字段等操作。

1. 重命名文档字段

小波发现之前设计的数据库字段名存在一些问题，只要存在长度特性的农产品，均使用的是 leaf_length 字段，这显然对于根茎类农产品是不对的。于是小波思考能否将字段名 leaf_length 修改为 length。MongoDB 显然能够轻松实现。例如，将 products 集合中的所有 leaf_length 字段修改为 length，如例 5-24 所示。

【例 5-24】 将 products 集合中的所有 leaf_length 字段修改为 length

```
> db.products.update({},{$rename:{"leaf_length":"length"}},{multi:true})
WriteResult({ "nMatched" : 4, "nUpserted" : 0, "nModified" : 3 })
> db.products.find().pretty();
{
        "_id" : ObjectId("62d6c4ba2fca6e0caacdefc9"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "price" : 0,
        "unit" : "斤",
```

```
        "shipping_address" : "陕西省渭南市大荔县",
        "length" : "15cm 以上"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefca"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区",
        "length" : "25 - 30cm"
}
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "specifications" : {
                "家庭实惠装 - 长 30 - 40cm - 5 斤装 二级,5.00 斤/箱" : 24,
                "优良装 - 长 45 - 50cm - 5 斤装 二级,5.00 斤/箱" : 26,
                "优选装 - 长 55 - 60cm - 5 斤装 一级,5.00 斤/箱" : 28,
                "精选装 - 长 65 - 70cm - 5 斤装 一级,5.00 斤/箱" : 30
        },
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县",
}
{
        "_id" : ObjectId("626fdb4e430ed4347b1cd182"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
```

2. 删除文档字段

近年受到天气影响，小波家乡山药的长度相比往年有所减短，只有 30～40cm 这一种规模可以出售，于是不得不删除 product_name 为“山药”的文档中原有的 specifications 字段，MongoDB 同样能够轻松实现。例如，删除 products 集合中 product_name 为山药的文档的 specifications 字段，如例 5-25 所示。

【例 5-25】 删除 products 集合中 product_name 为“山药”的文档的 specifications 字段

```
> db.products.update({"product_name":"山药"},{"$unset":{"specifications":1}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.products.find({"product_name":"山药"}).pretty();
{
        "_id" : ObjectId("62d6c4d22fca6e0caacdefcb"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
```

```
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县",
}
```

3. 新增文档字段

删除 specifications 字段后，山药没有了价格和长度信息，于是需要新增 price 和 length 字段。对于文档数据库的 MongoDB 实现起来仍然十分轻松。例如，在 products 集合中 product_name 为山药的文档中新增 price 和 length 字段，如例 5-26 所示。

【例 5-26】 在 products 集合中 product_name 为“山药”的文档中新增 price 和 length 字段

```
> db.products.update({"product_name":"山药"},{ $ set:{"price":4.0,"length":"30 - 40cm"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.products.find({"product_name":"山药"}).pretty();
{
        "_id" : ObjectId("62d7ae7c2fca6e0caacdefcd"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县",
        "length" : "30 - 40cm",
        "price" : 4
}
```

此处仅简单介绍了 MongoDB 作为文档数据库的灵活之处，具体操作读者可自行查阅资料了解并学习，进一步感受 MongoDB 的便捷与灵活。

5.4.10 小结

5.4 节为本章的重难点内容，读者需要认真理解与掌握，可以通过本章所给的示例进行实验。本节仅介绍了 MongoDB 中数据库、集合和文档的简单操作，更深入的操作读者可自行查阅资料学习了解。


视频讲解

5.5 MongoDB 基于图形化管理工具的图数据库查询方法

在 5.3.1 节 MongoDB 数据库的安装中，已简单提及 MongoDB Compass 图形化界面管理工具。MongoDB Compass 工具为 MongoDB 官方自带的图形化管理工具，相比于命令行界面更加便捷易用。

5.5.1 MongoDB Compass 的简单使用

MongoDB Compass 登录主页面如图 5-9 所示。

使用 MongoDB Compass 新建连接，如图 5-10 所示。

此处默认使用本机 27017 端口连接 MongoDB 数据库，同时可以通过展开 Advanced Connection Options（高级连接选项）设置更多的信息。

MongoDB Compass 连接后主页面如图 5-11 所示。

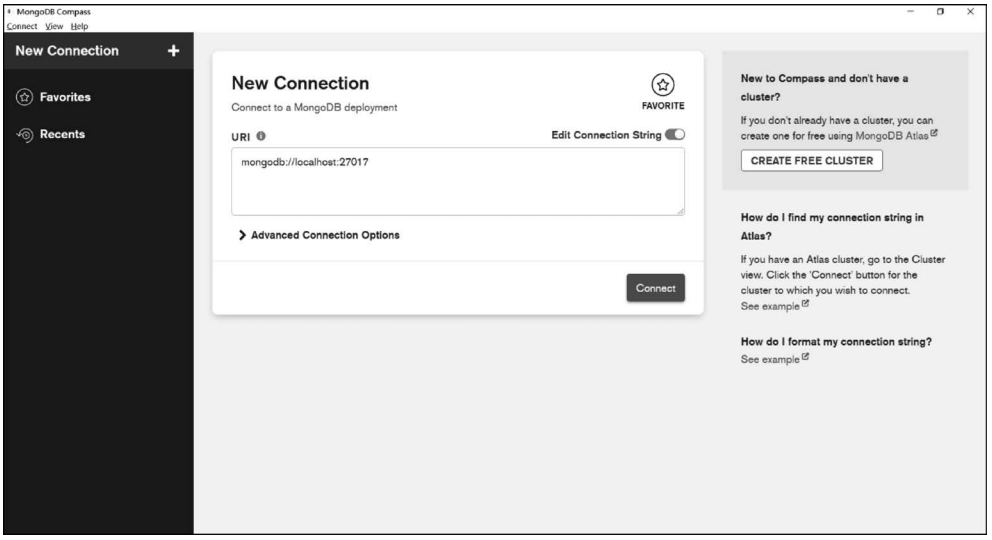

图 5-9　MongoDB Compass 登录主页面

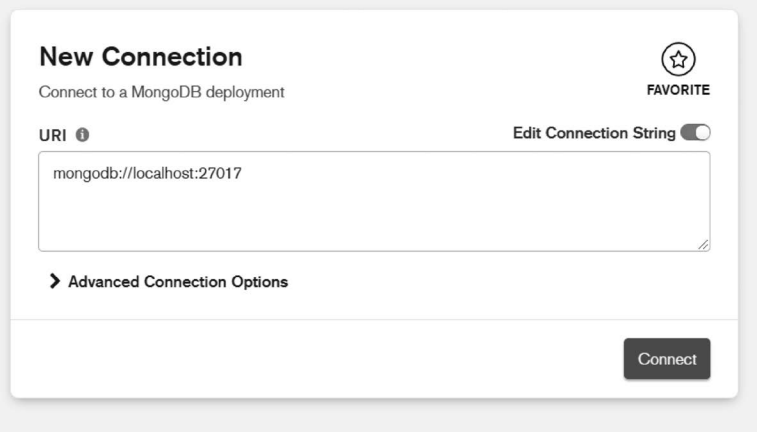

图 5-10　MongoDB Compass 新建连接

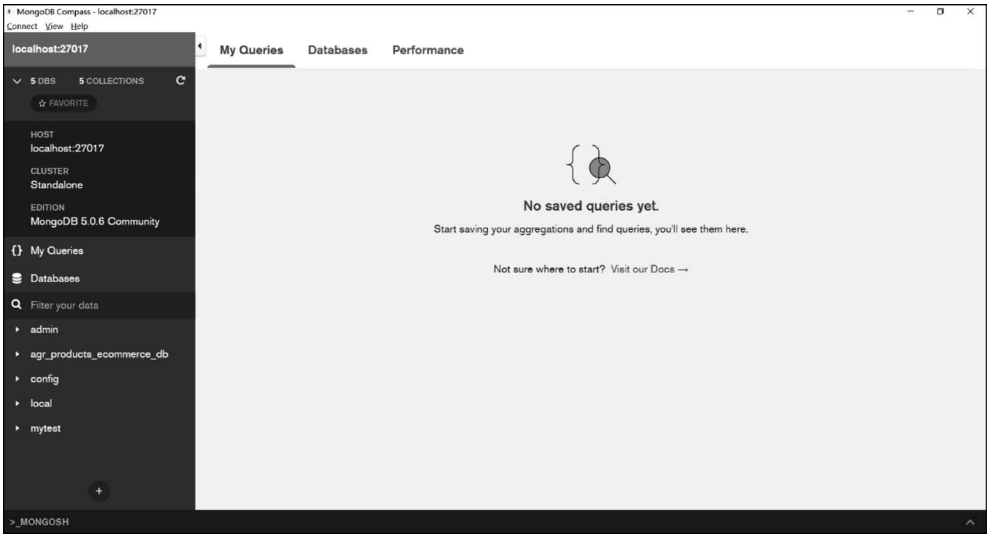

图 5-11　MongoDB Compass 连接后主页面

左侧导航栏显示了 MongoDB 数据库的版本信息，以及存在的数据库信息等。上方导航栏分别为 My Queries(搜索)、Databases(数据库)、Performance(可视化展示)功能。需要注意的是，在页面下方存在_MONGOSH，即 5.5 节重点介绍的命令行模式下操作 MongoDB，将其内嵌至 MongoDB Compass 中，便于切换使用。

小波的农产品电商数据库设计又更进了一步，打开农产品电商数据库(agr_products_ecommerce_db)，在 5.5 节中已经创建了 products 集合，打开 products 集合，可看到相关数据。products 集合部分数据可视化如图 5-12 所示。

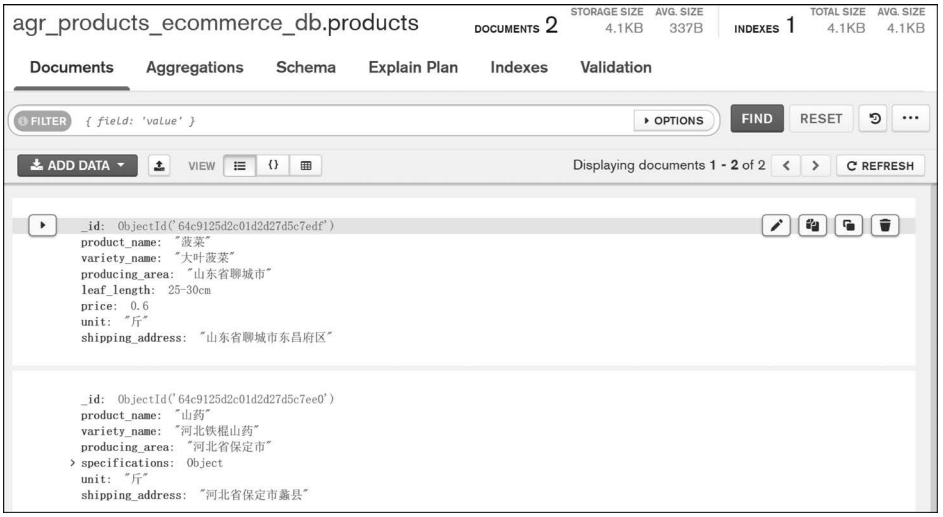

图 5-12　products 集合部分数据可视化

5.5.2　MongoDB Compass 的数据库操作

MongoDB Compass 支持 5.4 节基于命令行的所有操作指令，且更加简单易用，此处首先介绍数据库操作。

单击左侧导航栏的 Databases 按钮即可跳转至数据库操作界面。数据库操作界面如图 5-13 所示。

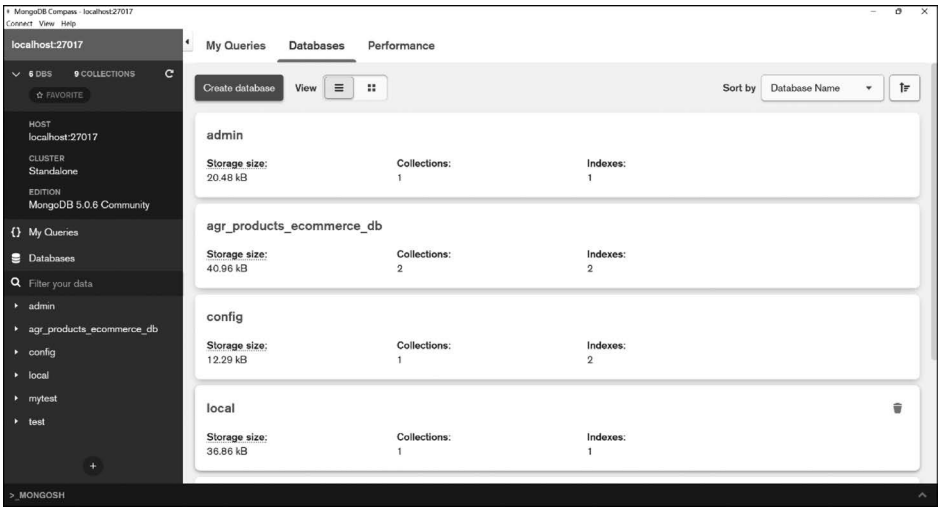

图 5-13　数据库操作界面

1. 创建数据库操作

单击 Create Database 按钮，即可创建数据库。MongoDB Compass 创建数据库时，除需要指定创建的数据库名外，还需要指定其中的一个集合名。使用 MongoDB Compass 创建数据库，如图 5-14 所示。读者还可展开 Advanced Collection Options 选项卡，对创建的集合进行高级设置。

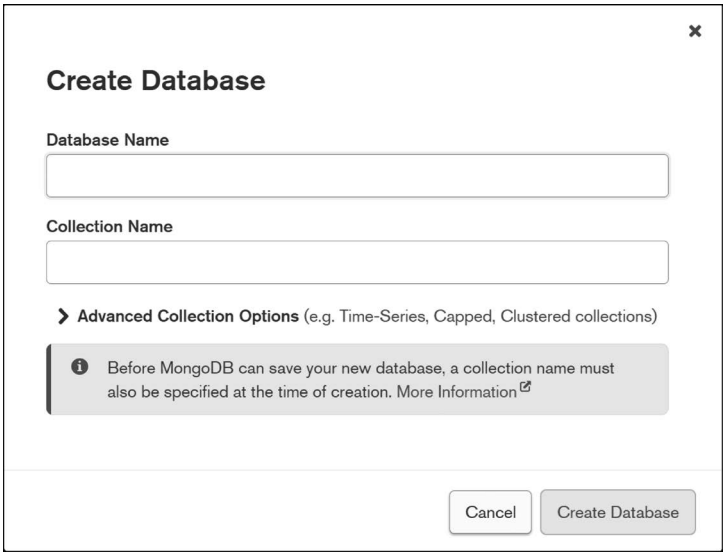

图 5-14　使用 MongoDB Compass 创建数据库

2. 删除数据库操作

在 MongoDB Compass 中，只需单击某个数据库右上角的删除图标，即可实现删除数据库操作。为了防止误操作，MongoDB Compass 还需要输入删除的数据库名进行二重验证。

5.5.3　MongoDB Compass 的集合操作

与数据库操作类似，单击 Create collection 按钮即可创建集合，单击某个集合右上角的删除图标即可实现删除集合操作，同样需要二重验证。单击集合即可查看当前集合中的所有文档，具体文档操作将在 5.5.4 节介绍。

5.5.4　MongoDB Compass 的文档操作

1. 文档添加操作

单击 ADD DATA 按钮，可选择 Import File(导入文件)和 Insert Document(插入文档)选项。此处以 Insert Document(插入文档)选项为例。MongoDB Compass 添加文档操作如图 5-15 所示。

此处_id 会自动生成，可自行修改，但需要确保唯一性。

2. 文档更新操作

鼠标光标浮于文档上方，右侧会显示 4 个操作按钮，其中第一个按钮为 Edit document(更

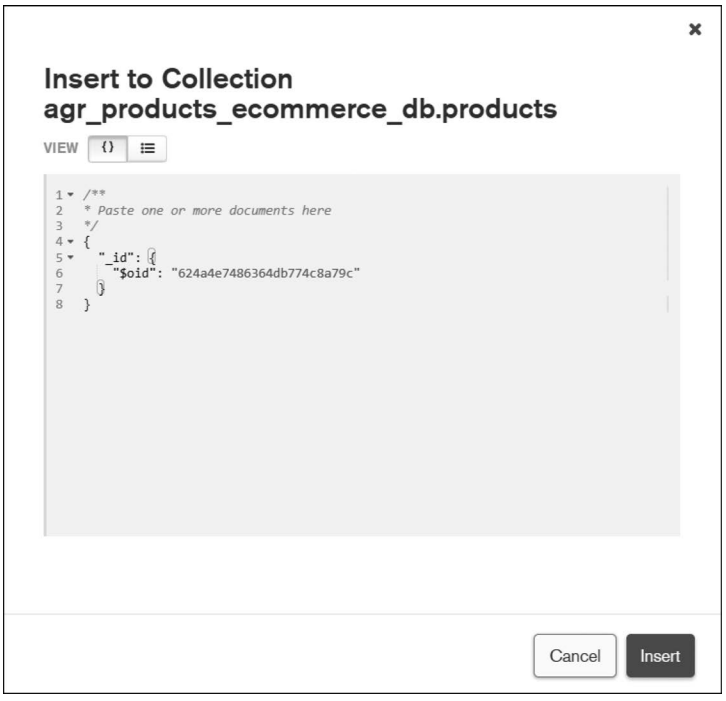

图 5-15　MongoDB Compass 添加文档操作

新文档）。单击 Edit document 按钮便可直接在其上方进行更新操作。

3. 文档删除操作

与文档更新操作类似，文档删除（Remove document）按钮在右侧 4 个操作按钮中的第 4 个。单击 Remove document 按钮便可直接删除当前文档。

4. 文档查询操作

文档可视化上面有一个搜索框，即为文档查询功能，直接输入查询条件即可。例如，查询 products 集合中 product_name 为“菠菜”的数据，如图 5-16 所示。

图 5-16　查询 products 集合中 product_name 为“菠菜”的数据

展开右侧的 OPTIONS 按钮，可输入更多的查询条件，实现 5.5.4 节中的各种查询功能。MongoDB Compass 高级查询选项如图 5-17 所示。

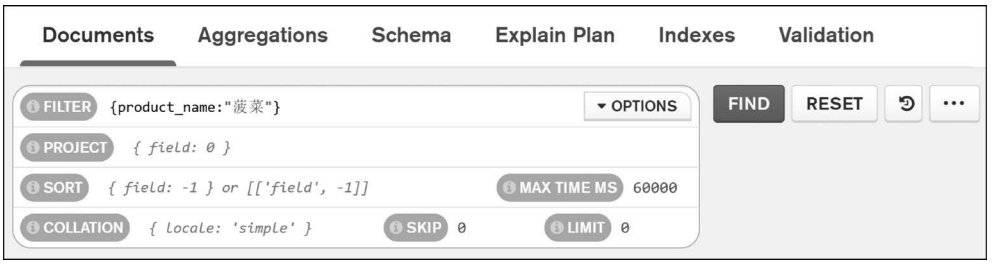

图 5-17　MongoDB Compass 高级查询选项

灵活使用 OPTIONS 中的各种查询条件，可实现基于命令行界面的所有查询操作。例如，查询 products 集合中 price 大于或等于 0.6 的文档的 product_name 和 price 信息，且以 price 升序显示，如图 5-18 所示。

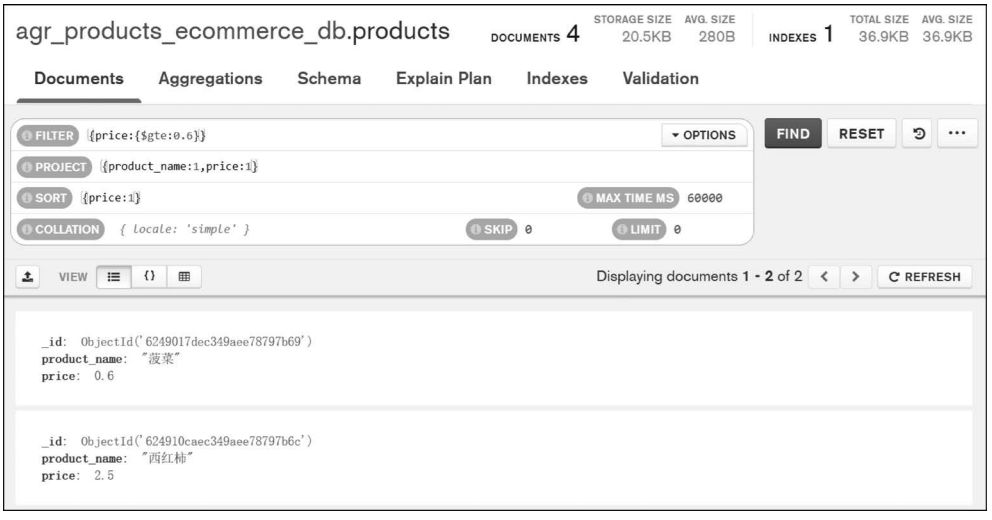

图 5-18　查询 products 集合中 price 大于或等于 0.6 的文档的 product_name 和 price 信息，且以 price 升序显示

5.6　MongoDB 基于 Python 的数据库连接和查询

5.1 节 MongoDB 数据库的简介中已提及 MongoDB 支持多种语言，如 Python、Java 等。随着近年来人工智能的火热，Python 的热度也推上顶峰。Python 凭借其简单易用、高可读性等特点，近年来被人们广泛使用。Python 的设计具有很高的可读性，同时又是一种交互式语言，对于初学者极其友好。又因其也是面向对象语言，能够很好地对代码和对象进行封装。因此，本节将重点介绍 Python 如何连接 MongoDB 数据库，并进行相应的操作。

注意：本节的所有实验均基于 Python 3.9 进行，Python 2.x 版本可能存在相关操作无法使用的情况。使用的编译器为 VS Code，读者也可使用其他编译器在自己的实验环境下运行相应的代码。

5.6.1　PyMongo

PyMongo 是 Python 提供的操作 MongoDB 数据库的安装包，基于 PyMongo 能够非常方

便地操作 MongoDB 数据库。

首先安装 PyMongo，可直接通过命令 pip3 install pymongo 进行安装。若通过 Anaconda 管理 Python，则可以通过 conda/pip install pymongo 命令安装。

5.6.2 Python 连接 MongoDB

PyMongo 通过 MongoClient()方法对 MongoDB 数据库进行连接。所有连接均为本地连接，且端口为 27107(默认端口)。

```
#无密码连接
import pymongo
client = pymongo.MongoClient(host = '127.0.0.1', port = 27017)
#有密码连接
import pymongo
client = pymongo.MongoClient(host = '127.0.0.1', port = 27017, username = '用户名', password =
'密码')
```

使用 print(client.server_info())判断是否连接成功。

成功连接 MongoDB 数据库后，可直接通过此前的 client 对象获取 Database(数据库)和 Collection(集合)。PyMongo 提供了两种获取 Database 和 Collection 的方式，若 MongoDB 中没有输入同名的 Database 和 Collection，则会自动创建。

第一种方式：

```
mongo_db = mongo_client['数据库名']
mongo_collection = mongo_db['集合名']
```

第二种方式：

```
mongo_db = mongo_client.数据库名
mongo_collection = mongo_db.集合名
```

【例 5-27】 连接 MongoDB 并获取农产品电商数据库(agr_products_ecommerce_db)的信息

```
import pymongo
client = pymongo.MongoClient('127.0.0.1',27017)
agr_products_ecommerce_db = client['agr_products_ecommerce_db']
print(agr_products_ecommerce_db)
```

【例 5-28】 连接 MongoDB 并获取 products 集合信息

```
import pymongo
client = pymongo.MongoClient('127.0.0.1',27017)
agr_products_ecommerce_db = client['agr_products_ecommerce_db']
products_col = agr_products_ecommerce_db['products']
print(products_col)
```

5.6.3 Python 对文档的 CURD 操作

MongoDB 数据库的核心为文档，文档为 MongoDB 数据库的最小组成部分。因此，本小节将基于 Python 实现对文档的 CURD 操作，即 Create(创建)、Update(更新)、Retrieve(读取)和 Delete(删除)。

1. 基于 Python 的文档创建

与命令行界面相同，PyMongo 同样支持单条数据的插入和多条数据的插入，分别由 insert_one()函数和 insert_many()函数实现。

基本语法为：col.insert_one(document)和 col.insert_many(documents)。

- col：待插入文档所在的集合。
- document：待插入的单条文档。
- documents：待插入的多条文档。

【例 5-29】 在 products 中插入单条数据。

```
import pymongo
client = pymongo.MongoClient('127.0.0.1',27017)
agr_products_ecommerce_db = client['agr_products_ecommerce_db']
products_col = agr_products_ecommerce_db['products']
doc = {
    "product_name": "菠菜",
    "variety_name": "大叶菠菜",
    "producing_area": "山东省聊城市",
    "leaf_length": "25 - 30cm",
    "price": 0.6,
    "unit": "斤",
    "shipping_address": "山东省聊城市东昌府区"
}
res = products_col.insert_one(doc) # 返回插入的对象
print(res)
```

2. 基于 Python 的文档查询

文档插入同样支持查询多条数据和一条数据，查询条件可随意设置，通过有效的查询条件可实现 5.5.4 节中所有的文档查询指令。PyMongo 主要提供了两种方法：find_one()和 find()。

- find_one()：匹配第一条满足查询条件的数据，结果以 Dict(字典)形式返回。若没有查询到结果，则返回 None。
- find()：返回所有满足查询条件的数据。

基本语法为：find_one(query,options)和 find(query,options)。

参数说明如下。

- query：查询条件。
- options：可选参数，如投影、排序等。

【例 5-30】 查询 products 集合中 product_name 为“西红柿”的数据

```
import pymongo
client = pymongo.MongoClient('127.0.0.1',27017)
agr_products_ecommerce_db = client['agr_products_ecommerce_db']
products_col = agr_products_ecommerce_db['products']
query = {'product_name':'西红柿'}
res = products_col.find(query) # 查询到的所有数据
for x in res:
    print(x)
```

文档查询操作相当丰富，读者可查阅相关资料进行更加复杂的查询。

3. 基于 Python 的文档更新

对于更新操作，PyMongo 提供了 update_one()和 update_many()函数，通过传入相关参数，可以实现非常丰富的操作，例如更新时，若没有满足条件，则插入数据等。

- update_one()：只会更新满足条件的第一条数据。
- update_many()：更新所有满足条件的数据。

基本语法为：update_one(query,values)和 update_many(query,values)。

参数说明如下。

- query：查询条件。
- values：修改后的数值。

【例 5-31】 更新 products 集合中 product_name 为“西红柿”的第一条文档的 price 字段值为 2.6

```
import pymongo
client = pymongo.MongoClient('127.0.0.1',27017)
agr_products_ecommerce_db = client['agr_products_ecommerce_db']
products_col = agr_products_ecommerce_db['products']
query = {'product_name':'西红柿'}
new_data = {'$set':{'price':2.6}}
res = products_col.update_one(query,new_data)
print(res.modified_count)     # 返回修改的条数
```

4. 基于 Python 的文档删除

关于文档删除操作，PyMongo 同样提供了两种方式，分别为 delete_one()和 delete_many()。

- delete_one()：删除满足条件的第一条数据。
- delete_many()：删除满足条件的所有数据。

基本语法为：delete_one(query)和 delete_many(query)。

参数说明如下。

- query：查询条件。

【例 5-32】 删除 products 集合中 product_name 为“菠菜”的所有文档

```
import pymongo
client = pymongo.MongoClient('127.0.0.1',27017)
agr_products_ecommerce_db = client['agr_products_ecommerce_db']
products_col = agr_products_ecommerce_db['products']
query = {'product_name':'菠菜'}
res = products_col.delete_many(query)
print(res.deleted_count)     # 成功删除的文档条数
```


视频讲解

5.7 MongoDB 的维护

5.7.1 MongoDB Database Tools 的安装与使用

1. MongoDB Database Tools 的安装

MongoDB 新版安装包并不包含相关工具包，如 mongoimport、mongoexport、mongodump

等，但是 MongoDB 官方提供了 MongoDB Database Tools，支持对 MongoDB 进行各种维护操作。

MongoDB Database Tools 的下载地址详见前言二维码。MongoDB 提供了两种下载方式，分别为.msi 文件和.zip 文件。本节针对.msi 文件进行安装介绍。下载并安装后，直接用鼠标双击安装包即可开始安装，其间支持自定义安装路径。MongoDB Database Tools 自定义安装路径如图 5-19 所示。此处为默认安装路径。

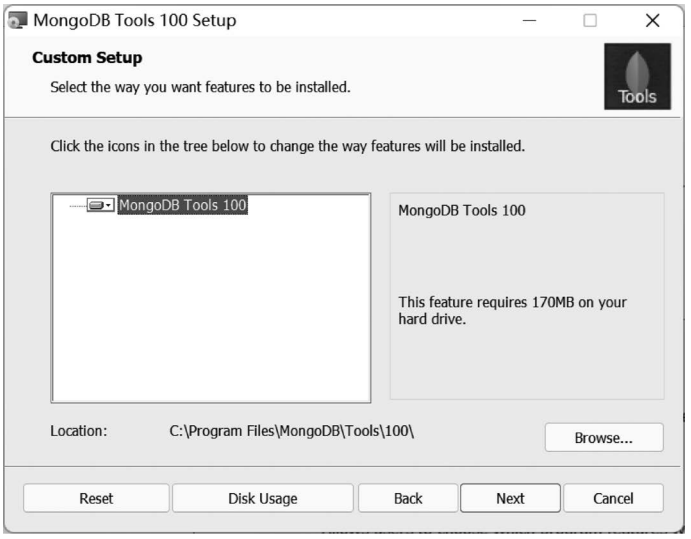

图 5-19　MongoDB Database Tools 自定义安装路径

MongoDB Database Tools 包含的工具如图 5-20 所示。MongoDB 的数据导入和导出、数据备份、数据恢复等都将借助其中的工具。

bsondump.exe	2022/2/1 13:49	应用程序	18,710 KB
libsasl.dll	2022/2/1 13:50	应用程序扩展	112 KB
mongodump.exe	2022/2/1 13:49	应用程序	22,378 KB
mongoexport.exe	2022/2/1 13:50	应用程序	21,945 KB
mongofiles.exe	2022/2/1 13:50	应用程序	23,126 KB
mongoimport.exe	2022/2/1 13:50	应用程序	22,288 KB
mongorestore.exe	2022/2/1 13:50	应用程序	22,835 KB
mongostat.exe	2022/2/1 13:50	应用程序	21,607 KB
mongotop.exe	2022/2/1 13:50	应用程序	21,219 KB

图 5-20　MongoDB Database Tools 包含的工具

2. MongoDB Database Tools 的使用

MongoDB Database Tools 的使用方式有如下两种。

(1) 用命令提示符(CMD)窗口打开 MongoDB Database Tools 安装路径下的 bin 文件，输入相关命令。

(2) 将安装路径下的 bin 文件添加至 path 环境变量，之后可在任意位置打开命令提示符(CMD)窗口输入相关命令。

本节后续操作将针对第一种使用方式进行演示与讲解。

5.7.2 MongoDB 数据导入和导出

1. MongoDB 数据导入

MongoDB 借助 mongoimport 命令实现对数据的批量导入。具体操作指令如下：

```
mongoimport -d 数据库名 -c 集合名 --type csv|json --file 文件路径
```

参数说明如下。

- -d：需要导入数据的数据库，如果数据库不存在，则会自动创建。
- -c：需要导入数据的集合，如果不存在于数据库中，则会自动创建。
- --type：导入数据的文件格式，支持 CSV 和 JSON 格式。
- --file：存储数据的文件路径。

【例 5-33】 向 agr_products_ecommerce_db 数据库的 products 集合中导入 E:\MongoDB\input\products_input.json 文件存储的数据

```
C:\Program Files\MongoDB\Tools\100\bin>mongoimport -d agr_products_ecommerce_db -c products
--type json --file E:\MongoDB\input\products_input.json
2022-07-20T09:35:04.969+0800 connected to: mongodb://localhost/
2022-07-20T09:35:04.986+0800 2 document(s) imported successfully. 0 document(s) failed to
import.
```

2. MongoDB 数据的导出

MongoDB 借助 **mongoexport** 命令实现对数据的批量导出。具体操作指令如下：

```
mongoexport -d 数据库名 -c 集合名 -f field1,field2... -q 查询条件 -o 导出的文件路径
```

参数说明如下。

- -d：需要导出数据的数据库。
- -c：需要导出数据的集合。
- -f：需要导出数据的字段。
- -q：导出数据的查询条件。
- -o：导出数据存储的文件路径。

【例 5-34】 将 agr_products_ecommerce_db 数据库中 products 集合所有数据导出并存储至 E:\MongoDB\output\products_output.json 中

```
C:\Program Files\MongoDB\Tools\100\bin>mongoexport -d agr_products_ecommerce_db -c products
-o E:\MongoDB\output\products.json
2022-07-20T09:39:34.106+0800 connected to: mongodb://localhost/
2022-07-20T09:39:34.130+0800 exported 5 records
```

5.7.3 MongoDB 数据备份

MongoDB 借助 **mongodump** 命令实现对 MongoDB 数据的备份。该命令可以导出所有数据到指令目录中。同时，mongodump 命令可以指定需要转存的服务器等。具体操作指令如下：

```
mongodump -h dbhost -d dbname -o dbdirectory
```

参数说明如下。

- -h：需要备份的服务器 IP 及端口(若不指定,则默认为本地服务器的 27017 端口)。
- -d：需要备份的数据库名。
- -o：备份数据存储的文件夹路径。

【例 5-35】 将本地服务器中的 agr_products_ecommerce_db 数据库备份至 E:\MongoDB\mongodump 文件夹中

```
C:\Program Files\MongoDB\Tools\100\bin>mongodump -h 127.0.0.1 -d agr_products_ecommerce_db
-o E:\MongoDB\mongodump
2022-07-20T09:48:14.407+0800 writing agr_products_ecommerce_db.products to E:\MongoDB\
mongodump\agr_products_ecommerce_db\products.bson
2022-07-20T09:48:14.418+0800 done dumping agr_products_ecommerce_db.products (5 documents)
```

除上述指令外,mongodump 命令还支持其他参数。mongodump 命令可选参数列表如表 5-9 所示。

表 5-9　mongodump 可选参数列表

语　法	描　述	示　例
mongodump --host host_name --port port_number	备份服务器名为 host_name 的所有 MongoDB 数据	mongodump --host 127.0.0.1 --port 27017
mongodump --dbpath db_path --out backup_dir	将 db_path 路径下所有 MongoDB 数据备份至 backup_dir 中	mongodump --dbpath/data/db/ --out /data/backup/
mongodump --collection col_name --db db_name	备份 db_name 数据中的 col_name 集合	mongodump --collection products --db agr_products_ecommerce_db

5.7.4　MongoDB 数据恢复

MongoDB 使用 mongorestore 命令恢复备份的数据。具体操作指令如下：

```
mongorestore -h <hostname><:port> -d dbname <path>
```

参数说明如下。

- --host <:port>,-h <:port>：MongoDB 所在服务器地址,默认为 localhost:27017。
- --db,-d：需要恢复的数据库名(这个名词可以和备份时的名词不相同)。
- --drop：恢复的时候,先删除当前数据,然后恢复备份的数据。
- <path>：指定备份数据所在目录,例如 E:\MongoDB\mongodump\agr_products_ecommerce_db。注意：不能同时指定<path>和--dir 选项,--dir 也可以设置备份目录。
- --dir：指定备份数据的目录。

其中,<>表示的选项为可选选项。

【例 5-36】 恢复 agr_products_ecommerce_db 数据库

```
C:\Program Files\MongoDB\Tools\100\bin>mongorestore -h localhost:27017 -d agr_products_
ecommerce_db E:\MongoDB\mongodump\agr_products_ecommerce_db
2022-07-20T10:04:08.239+0800 The --db and --collection flags are deprecated for this use-
case; please use --nsInclude instead, i.e. with --nsInclude=${DATABASE}.${COLLECTION}
```

```
2022-07-20T10:04:08.249+0800 building a list of collections to restore from E:\MongoDB\
mongodump\agr_products_ecommerce_db dir
2022-07-20T10:04:08.251+0800 reading metadata for agr_products_ecommerce_db.products from
E:\MongoDB\mongodump\agr_products_ecommerce_db\products.metadata.json
2022-07-20T10:04:08.268+0800 restoring agr_products_ecommerce_db.products from E:\MongoDB\
mongodump\agr_products_ecommerce_db\products.bson
2022-07-20T10:04:08.292+0800 finished restoring agr_products_ecommerce_db.products (5
documents, 0 failures)
2022-07-20T10:04:08.293+0800 no indexes to restore for collection agr_products_ecommerce_
db.products
2022-07-20T10:04:08.296+0800 5 document(s) restored successfully. 0 document(s) failed to
restore.
```

5.8 MongoDB 的拓展知识

5.8.1 MongoDB 的注意事项

MongoDB 作为文档数据库，具有简单易用、可扩展性强等特点。本小节以**农产品电商数据库**(agr_products_ecommerce_db)的设计与实现展开，在添加农产品信息时不难发现，不同类型的农产品具有不同的字段，MongoDB 无须像传统关系数据库一样，提前声明各个字段名称以及数据类型，直接以文档形式添加即可。同时，也可轻松实现添加字段、字段重命名、修改字段类型、删除字段等操作。

但 MongoDB 具有灵活便捷等优点的同时，也存在相关不足，具体为如下 3 点。

(1) 事务隔离等级不足：目前版本支持副本集事务和分布式事务。MongoDB 的事务隔离等级对于一些 NoSQL 的应用是够的，但对一些关键业务场景(例如银行)是不够的。

(2) 占用内存过大：每次空间不足时，会申请一大块硬盘空间。同时，删除记录并不释放空间，而只是标记为“已删除”。

(3) 没有像 MySQL 一样有成熟的维护工具。

5.8.2 其他类似数据库

MongoDB 作为文档数据库的代表，其文档完善、社区活跃，受到广大开发者的学习与探索。但除此之外，还存在一些其他的文档数据库也被应用，具体有以下 4 种。

1. Amazon DynamoDB

DynamoDB 由亚马逊团队开发，是一个完全托管的 NoSQL 数据库服务。在开发的易用角度，DynamoDB 没有 MongoDB 强大，但是从运维的角度来看，DynamoDB 省去了开发者部署、监控、维护数据库环境，节约了大量时间，同时强大的扩展能力又减轻了后续运维的压力。

2. Microsoft Azure Cosmos DB

Azure Cosmos DB 由微软开发，是第一个全球分布式的多模型数据库服务，用于构建全球范围规模的应用。Azure Cosmos DB 无须复杂的多数据库中心配置，就可以构建全球分布式应用程序，但是只支持在云端 Azure 使用，且目前不支持关系数据库模型和 SQL。

3. Couchbase

Couchbase 是一个用于交互式 Web 应用的 NoSQL 数据库。它是一个易于扩展的数据

库，具有高度灵活的数据模型，可提供高性能。

4. Firebase Realtime Database

Firebase 由 Google 在 2012 年开发。它是一个用于实时存储和同步数据的数据库。它是个由云托管的实时文档存储，可以灵活地从任何设备访问数据。

5.9　本章习题

1. MongoDB 数据库中的(　　)类比于关系数据库中的行或记录。
   A. 表格　　B. 集合　　C. 文档　　D. 字段
2. (多选)下面哪几种类型是 MongoDB 支持的类型(　　)。
   A. 字符串　　B. 日期　　C. 数组　　D. 正则表达式　　E. 对象
3. (多选)与数据库备份直接相关的指令为(　　)。
   A. mongodump　　B. mongorestore
   C. mongoimport　　D. mongoexport
4. MongoDB 的默认数据库有哪些，分别代表什么含义？

5. 请简述 MongoDB 的特点和应用场景。

6. 要将服务器名为 test 的所有数据备份到/data 目录(其中开放端口为 27017)，需要执行的命令是什么？

7. 假设目前存在 products 集合信息如下：

```
{
        "_id" : ObjectId("6352605a1d56b3803df4cd9d"),
        "product_name" : "胡萝卜",
        "variety_name" : "秤杆红萝卜",
        "producing_area" : "陕西省渭南市大荔县",
        "leaf_length" : "15cm 以上",
        "price" : 0.5,
        "unit" : "斤",
        "shipping_address" : "陕西省渭南市大荔县"
}
{
        "_id" : ObjectId("6352605a1d56b3803df4cd9e"),
        "product_name" : "菠菜",
        "variety_name" : "大叶菠菜",
        "producing_area" : "山东省聊城市",
        "leaf_length" : "25 - 30cm",
        "price" : 0.6,
        "unit" : "斤",
        "shipping_address" : "山东省聊城市东昌府区"
}
```

```
{
        "_id" : ObjectId("6352605a1d56b3803df4cd9f"),
        "product_name" : "山药",
        "variety_name" : "河北铁棍山药",
        "producing_area" : "河北省保定市",
        "unit" : "斤",
        "shipping_address" : "河北省保定市蠡县",
        "length" : "30 - 40cm",
        "price" : 4
}
{
        "_id" : ObjectId("6354f5d3fd31334cec82b877"),
        "product_name" : "西红柿",
        "variety_name" : "普罗旺斯番茄",
        "producing_area" : "山东省海阳市",
        "leaf_length" : "15cm 以上",
        "price" : 3,
        "unit" : "斤",
        "shipping_address" : "山东省海阳市"
}
```

参考 5.4.6 节的文档查询操作，实现查询 products 集合中所有商品价格的总和、平均值、最大值和最小值的语句。

8. 假设目前 products 集合如题 7 所示，参考 5.4.9 节的文档结构修改操作，实现将 price 字段修改为 unit_price，集合所有文档新增 others 字段（值为 null）。

9. MongoDB 中的分片是什么意思？

10. MongoDB 有哪些索引类型？