## 数据库分类

数据库主要分成两种：

-关系型数据库（RDBMS）

·MySQL、Oracle、DB2、SQL Server.....…

·关系数据库中全都是表

-非关系型数据库（No SQ）

·MongoDB、Redis...…

·键值对数据库

·文档数据库MongoDB

## MongoDB简介

### MongoDB介绍

MongoDB是为快速开发互联网Web应用而设计的数据库系统。

MongoDB的设计目标是极简、灵活、作为Web应用栈的一部分。

MongoDB的数据模型是面向文档的，所谓文档是一种类似于JSON的结构，简单理解

MongoDB这个数据库中存的是各种各样的JSON。(BSON)

### BSON(二进制的json)

## 数据逻辑结构

### 概念

数据库（database）：数据库是一个仓库，在仓库中可以存放集合。

集合（collection)：集合类似于数组，在集合中可以存放文档。

文档（document):文档数据库中的最小单位，我们存储和操作的内容都是文档。

在MongopB中，数据库和集合都不需要手动创建，当我们创建文档时，如果文档所在的集合或数据库不存在会自动创建数据库和集合

### 层次关系



### MongoDB的sort和投影

## 文档之间的关系

### 一对一：(one to one)

在mongodb中，可以通过内嵌文档的形式体现出一对一的关系

|  |
| --- |
| db.wifeAndHusband.insert([  {  name:"黄蓉",  husband:{  name:"郭靖"  }  }, {  name:"潘金莲",  husband:{  name:"武大郎"  }  }  ]); |

### 一对多：one to many /多对一 many to one

|  |
| --- |
| //一对多 用户(users)和订单(orders)  db.users.insert([  {username:"swk"},{username:"zbj"}  ]);  db.users.find();  db.orders.insert({  list:["苹果","香蕉","大鸭梨"],  //通过id判断是swk的  user\_id:ObjectId("5cf727798c210b5a40995fcc")  })  //通过id判断是swk的  db.orders.insert({  list:["馒头","猪八戒","曼胡"],  user\_id:ObjectId("5cf727798c210b5a40995fcd")  })  db.orders.find();  //查找当用户sewk的订单  var user\_id = db.users.findOne({username:"swk"}).\_id;  db.orders.find({user\_id:user\_id}); |

### 多对多：many to many

|  |
| --- |
| //多对多 分类 商品  // 老师 学生  db.teachers.insert([  {name:"洪七公"},  {name:"黄药师"},  {name:"龟仙人"}  ]);  db.teachers.find();  db.stus.insert([  {  name:"郭靖",  tech\_ids:[  ObjectId("5cf72c7c8c210b5a4099fc13"),  ObjectId("5cf72c7c8c210b5a4099fc14") ,  ObjectId("5cf72c7c8c210b5a4099fc15")  ]  },  {  name:"孙悟空",  tech\_ids:[  ObjectId("5cf72c7c8c210b5a4099fc14") ,  ObjectId("5cf72c7c8c210b5a4099fc15")  ]  }  ])  db.stus.find(); |

## MongoDB的安装

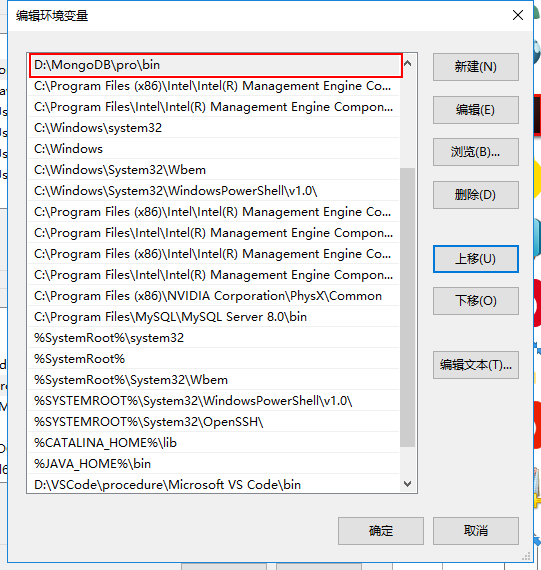
### MongoDB下载

<https://www.mongodb.com/dr/fastdl.mongodb.org/win32/mongodb-win32-x86_64-2008plus-ssl-4.0.9-signed.msi/download>

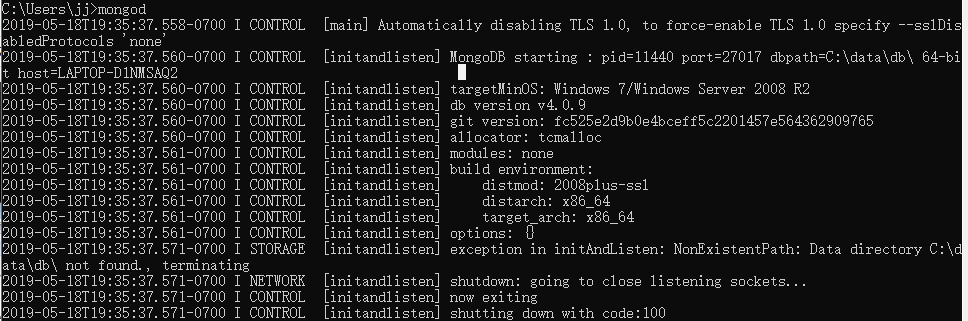
### 安装MongoDB和配置

配置MongoDB环境变量

MongoDB: D:\MongoDB\pro\bin



测试是否成功：cmd命令窗口下输入mongod 命令



### 创建数据库目录

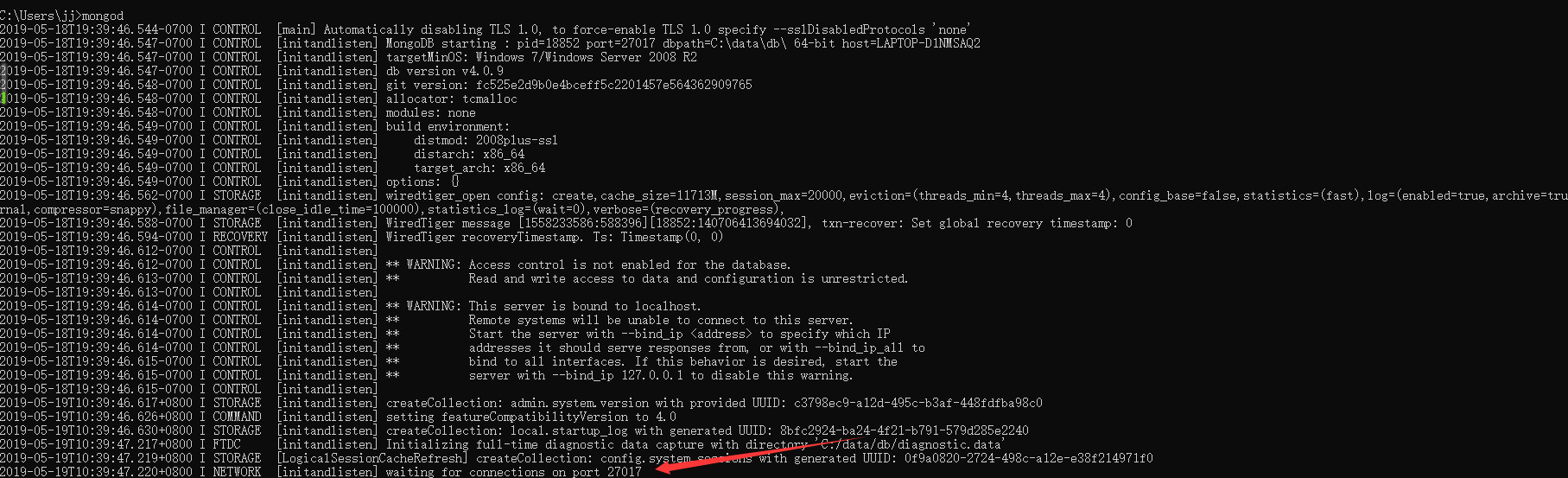
1、创建一个data文件夹，然后在data文件夹中创建一个db目录

C:\data\db

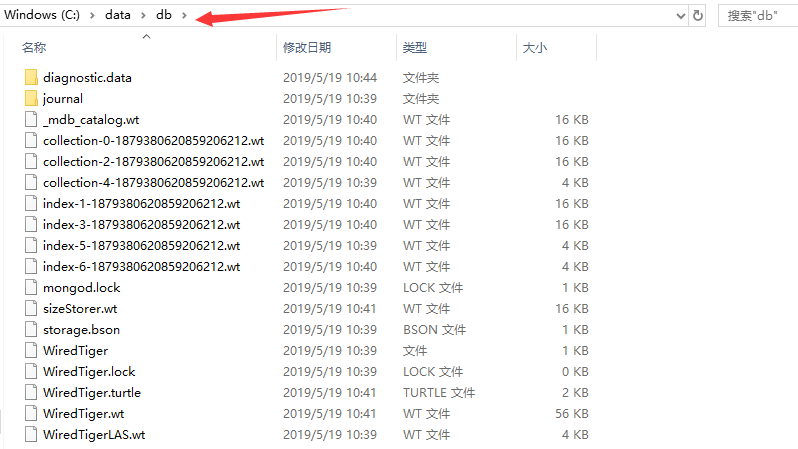
2、创建一个log文件夹，然后在data文件夹中创建一个log目录

C:\data\log 日志文件夹

2、打开cmd命令行窗口，输入mongoD启动mongodb服务器



会在该文件夹下自动创建下列文件

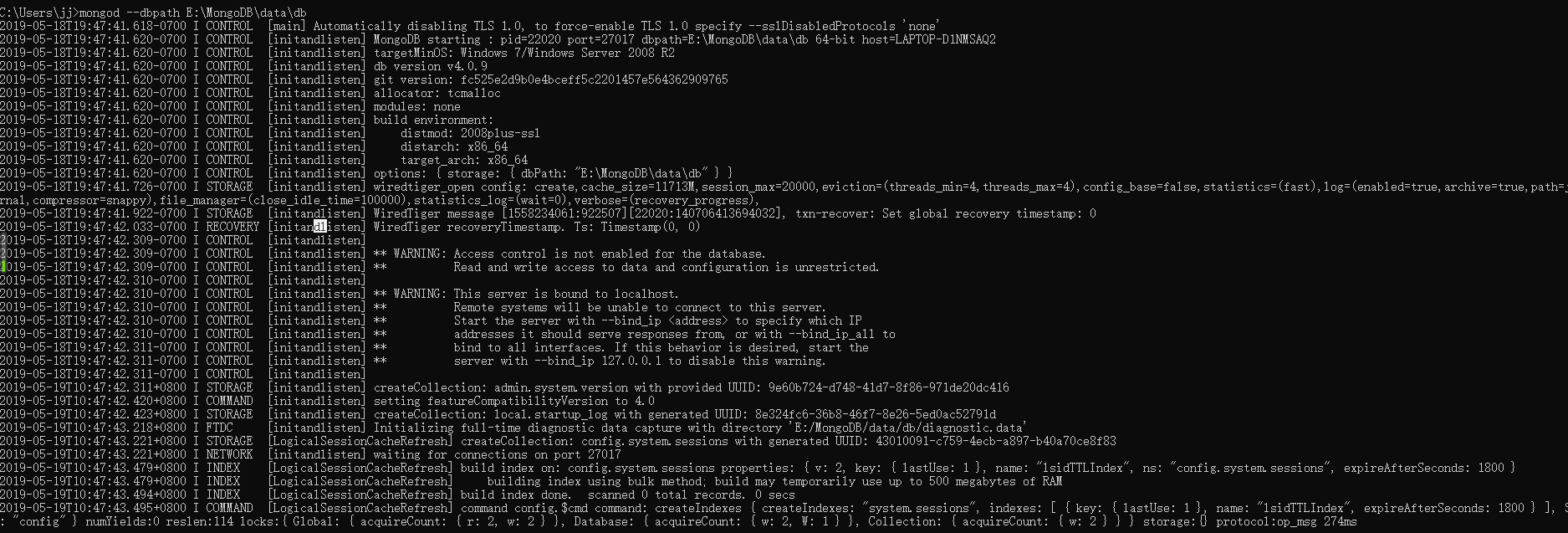


### 自定义数据库目录

通过命令制定数据库的位置,并指定数据库的端口

mongod –path 数据库命令 --prot 端口号

例如：mongod –path E:\MongoDB\data\db --prot 10086



### 启动mongoDB

数据库(database)

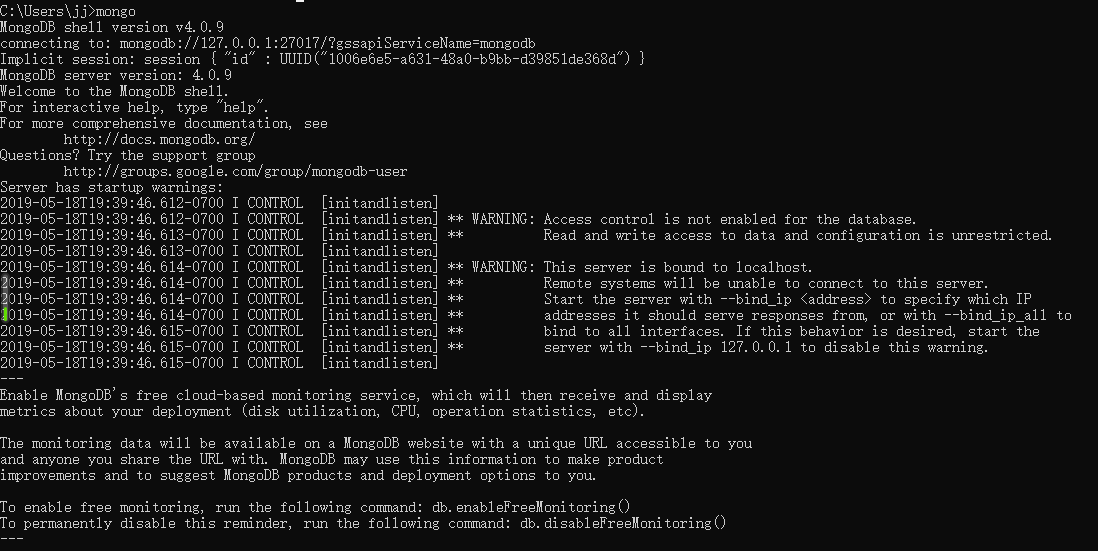
--数据库的服务器：服务器用来保存数据

启动服务器：mongod

--数据库的客户端：客户端用来操作服务器，对数据增删改查的操作、

启动服务端：mongo

输入：mongo，进入MongoDB



### 设置mongoDB自动启动

将MongoDB设置为系统服务，可以自动在后台启动，不需要每次都手动启动

1、创建对应的文件夹，在data下创建db和log文件夹

2、创建配置文件

在目录 D:\MongoDB\pro 创建配置文件 mongod.cfg

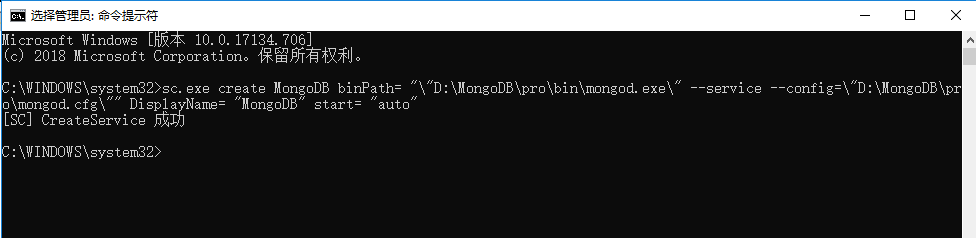
写入以下内容

|  |
| --- |
| systemLog:  destination:file  path: E:\MongoDB\data\log\mongod.log  storage:  dbPath: E:\MongoDB\data\db |

3、以管理源身份打开 命令行窗口

输入命令：sc.exe create MongoDB binPath= "\"D:\MongoDB\pro\bin\mongod.exe\" --service --config=\"D:\MongoDB\pro\mongod.cfg\"" DisplayName= "MongoDB" start= "auto"

显示：



## MongoDB图形化工具

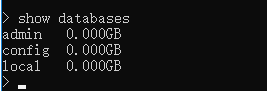
## mongoDB命令

### 数据库命令：

显示数据库：

show databases

show dbs



切换数据库：

use 数据库名 进入指定的数据库

显示数据中所有的集合

show clollections

在MongopB中，数据库和集合都不需要手动创建，当我们创建文档时，如果文档所在的集合或数据库不存在会自动创建数据库和集合

### 向数据库中插入文档：

1、 db.<collection>.insert(doc) :向集合中插入一个文档

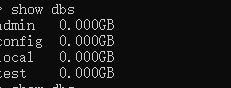
例如：向test数据库中的，stus 集合中插入一个新的学生对象

{name:’孙悟空’,age:18,gender:’男’}

db.stus.insert({name:’孙悟空’,age:18,gender:’男’})



插入数据后会自动的创建数据库



2、向集合中插入多个文档

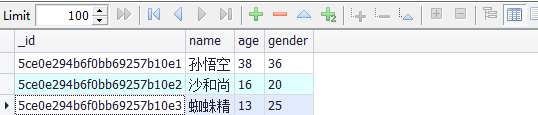
插入一个数据

|  |
| --- |
| db.stus.insert([  {name:"孙悟空",age:38,gender:"36"},  {name:"沙和尚",age:16,gender:"20"},  {name:"蜘蛛精",age:13,gender:"25"}  ])  db.stus.find() |

3、数据库\_id属性

当我们向集合中插入文档时，如果没有给文档指定id属性，则数据库会自动为文档添加id该属性用来作为文档的唯一标识。

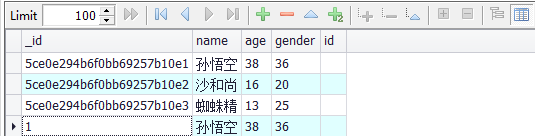
id属性的作用：保证数据的唯一性。



4、自己指定id

\_id战们可以自己指定，如果我们指定了或据库就不会在满加了，如果自己指定\_id 也必须确保它的唯一

db.stus.insert({\_id:1,name:"孙悟空",age:38,gender:"36"});



5、插入一个和插入多个

db.collection.insertOne();

db.collection.insertMany();

### 查询集合

1、db.<collection>.find() 查询当前集合中的所有文档



2、find():返回一个数组

1、find()用来查询集合中所有符合条件的文档

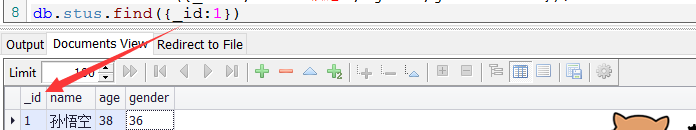
2、find()可以接受一个对象作为条件参数

3、如果find()中不写任何东西，表示查询所有的文档

{ } 查询集合中所有的文档

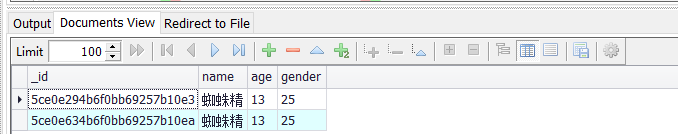
{属性:值} 查询属性是指定的文档

例如：db.stus.find({});



4、多个条件的查询

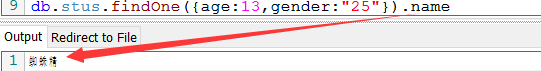
db.stus.find({age:13,gender:"25"})



3、findOne() ：返回一个对象

db.collection.findOne()：用来查询集合中符合条件的第一个文档

例如：db.stus.findOne({age:13,gender:"25"}).name



4、lenth()查询结果的长度

db.collection.find().count()：查询结果的数量。

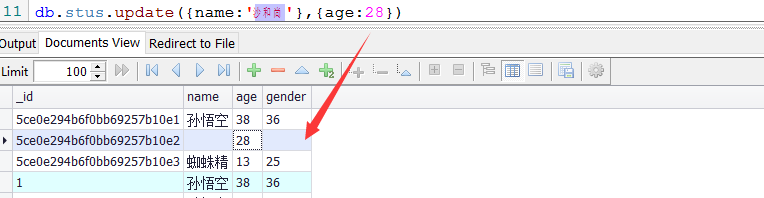
### 修改命令

$set可以用来修改文档的指定属性

$unset 可以用来删除文档的指定属性。

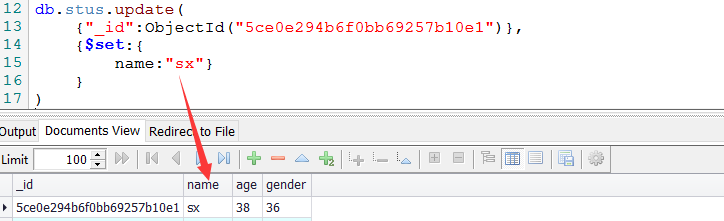
1、 db.collection.update(查询调条件,新对象) 此命令会将新对象替换为查询出来的对象。

db.stus.update({name:”沙和尚”},{age:28}) ：



2、修改操作符

set 可以用来修改文档中的指定属性。



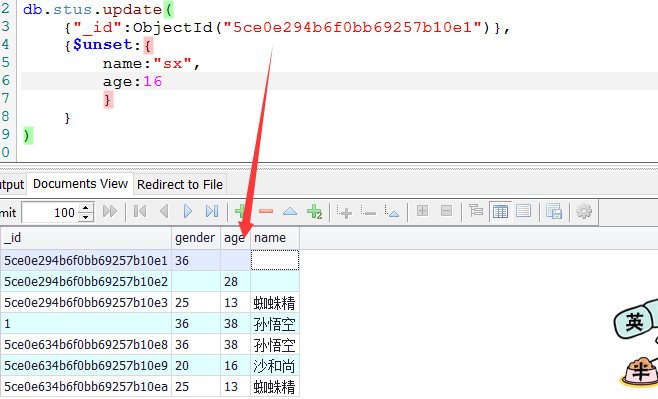
3、设置多个值

|  |
| --- |
| db.stus.update(  {"\_id":ObjectId("5ce0e294b6f0bb69257b10e1")},  {$set:{  name:"sx",  age:16  }  }  ) |

4、删除属性

删除对应id的name和age属性值

|  |
| --- |
| db.stus.update(  {"\_id":ObjectId("5ce0e294b6f0bb69257b10e1")},  {$unset:{  name:"sx",  age:16  }  }  ) |



4、修改多个

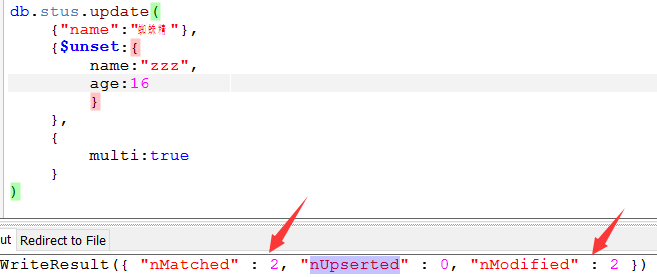
db.stus.update():默认情况下只会修改一个。

db.stus.updateMany():会修改多个。

db.stus.updateOne();修改一个

5、update()修改多个

|  |
| --- |
| db.stus.update(  {"name":"蜘蛛精"},  {$unset:{  name:"zzz",  age:16  }  },  {  multi:true //使用该属性修改多个  }  ) |



### 删除

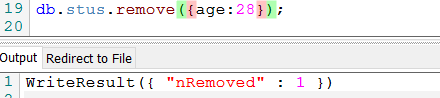
db.collection.remve(); remove 必须传参

db.collection.deleteOne();

db.collection. deleteMony();

db.collection.remve() :删除符合条件的所有文档，想删几个删几个。默认情况下删除多个。

删除一个：db.stus.remove({age:16},true); 后边传一个true



清空集合：

db.stus.remove({});如果只传递一个空对象作为参数，将删除集合中的所有文档。

此方式性能较差。remove是先匹配后删除。一个一个删。

删除集合：

db.stus.drop(); 直接删除集合。

删除数据库：

db.dropDatabase;删除数据库

## 练习

|  |
| --- |
| //1、进入my\_test数据库  use my\_test  //2、向数据库的user集合插入一个文档  db.users.insert({  username:"sunwukong"  });  //3、查询user集合中的文档  db.users.find();  //4、查询数据库  show dbs  //5、向数据库的user集合中插入一个文档  db.users.insert({  username:"zhubajie"  });  //6、统计数据库user集合中的数量  db.users.find().count();  //7、查询数据库user集合中username为sunwukong的文档  db.users.find({username:"sunwukong"});  //8、向数据库user集合的username为sunwukong的文档，添加一个address属性为huaguoshan  db.users.update({username:"sunwukong"},{$set:{address:"huaguoshan"}});  //9、使用{username:"tangseng"}替换username为zhubajie的文档  db.users.replaceOne({username:"zhubajie"},{username:"tangseng"});  //10、删除username为sunwukong的文档的address属性,注：删除属性用update  db.users.update({username:"sunwukong"},{$unset:{address:1}})  //11、向username为孙悟空的文档中，田家园一个hobby(对象)  //mongodb的文档的属性值也是一个文档，当一个文档的属性值是一个文档时，我们称这个文档为内嵌文档  db.users.update({username:"sunwukong"},{$set:{hobby:{cities:["beijing","shanghai","shenzheng"],movie:["sanguo","hero"]}}});  //12、给username为tangseng的文档中，添加一hobby:{movie:["chines","king"]}  db.users.update({username:"tangseng"},{$set:{hobby:{movie:["chines","king"]}}});  //13、查询查询细化你电影hero的文档  //电影属于内嵌文档，属性的属性。  //mongodb支持通过内嵌文档的水泥杆进行查询，如果要查询内嵌文档则可以通过.的形式来匹配。  //如果要通过内嵌文档来对文档进行查询，此时属性名必须使用引号.表示hobby.movies是一个表达式。而不是一个属性  db.users.find({'hobby.movie':"hero"});  //14、向tangseng中添加一个新的电影Interstellar $set是设置  //$push 用于向数组中添加一个新的元素  //$addToSet 向数组中添加一个元素，如果数组中已近存在了该元素，则不会添加。  db.users.update({username:"tangseng"},{$push:{"hobby.movie":"Insterstellar"}});  db.users.find();  //15、删除喜欢北京的用  db.users.remove({"hobby.cities":"beijing"});  //16、删除user集合  db.users.remove({})  db.users.drop()  show dbs;  //17向numbers中插入20000条数据  for(var i = 1;i<20000;i++){  db.numbers.insert({num:1})  }  db.numbers.find()  db.numbers.remove({});  //进阶版  var arr = [];  for(var i = 1; i<20000; i++){  arr.push({num:i});  }  db.numbers.insert(arr)  db.numbers.find() |

## MongoDB的sort和投影

### sort

//排序

//1、为1：升序 为-1时，降序排列

//2、可以指定多个排序字段。

//3、limit skip sort 可以以任意的顺序调用

db.emp.find({}).sort({sal:1,empno:-1})

### 投影

在查询是，可以在第二个参数的位置来设置查询结果的投影 db.emp.find({},{ename:1,\_id:0,sal:1});

1：表示要显示的字段。

0：表示不显示的字段，\_id是默认显示的。

## Monggoose