# 一，MongoDB概述

## 1，什么是数据库

**问题1：淘宝网，聊天记录，游戏币....这些信息保存在什么地方？**

普通的记事本

**通常是保存在数据库中。**

**问题2：使用记事本保存数据有什么问题？**

1，存储的数据不是海量

2，程序操作数据不方便

3，不太安全

为了解决上面的不足，就出现了**数据库**的概念，它能更有效的管理数据。数据库的本质也是一个文件，但是这个文件有结构，更有得数据的管理，常见操作CRUD。 增删改查。

**什么是数据库？**

**定义1：**

数据库(Database)是按照数据结构来组织、存储和管理**数据**的**仓库。**

**定义2：**

严格来说，数据库是长期储存在计算机内、有组织的、可共享的数据集合。

**前面我们说过上网的本质：**



当在一台电脑上装了一个node的环境，那么这台电脑就是一个web服务器，它就对外提供web服务。如果一台电脑上装了一个mysql， mongodb，那么这台电脑就是一个数据库服务器，它就对外提供数据操作服务。

## 2，数据库的分类

**数据库有很多，分类如下：**

* 1，关系型数据库
* 2，非关系型数据库

**1，关系型数据库**

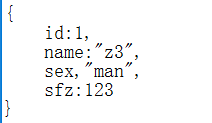
**如下图：**



表table，行row，列(字段)

**2，非关系型数据库**

**如下图：**



**3，关系型数据库有哪些？**

mysql, oracle，sqlServer, DB2....

**4，非关系型数据库有哪些？**

mongodb, redis, couchdb.....

## 3，什么是nosql

**什么是SQL?**

结构化查询语言(**S**tructured **Q**uery **L**anguage)

**什么是nosql?**

它泛指非关系型数据库，是Not Only SQL的简称，有些地方，non-relational。

**nosql的适应场景：**

NoSQL数据库在以下的这几种情况下比较适用：1、数据模型比较简单；2、需要灵活性更强的IT系统；3、对数据库性能要求较高；4、不需要高度的数据一致性；5、对于给定key，比较容易映射复杂值的环境。

**nosql的优点？**

1，灵活性强

2，高性能

3，易扩展

**有了nosql，是不是就意义着关系型的数据就淘汰了？**

nosql的出现是为了弥补关系型数据库的缺点，而不是为了代替关系型数据库，通常在一个系统中，会用到很多数据库，比如淘宝：oracle, mysql, mongodb....

## 4，什么是mongodb

mongodb是nosql中比较流行的一种非关系型数据库。2009年左右，mongodb出来了。

MongoDB是一个基于分布式文件存储的数据库。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。

**特点：**

1，易使用

2，易扩展

3，易部署

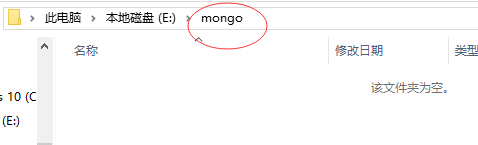
4，高性能

# 二，MongoDB入门

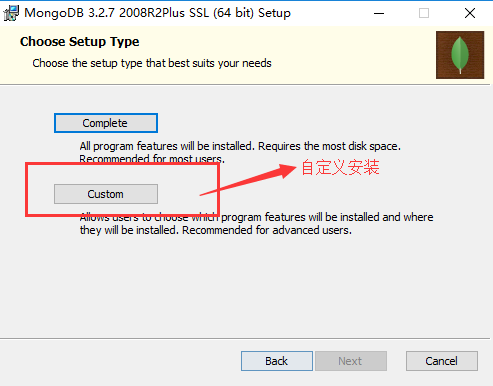
## 1，安装

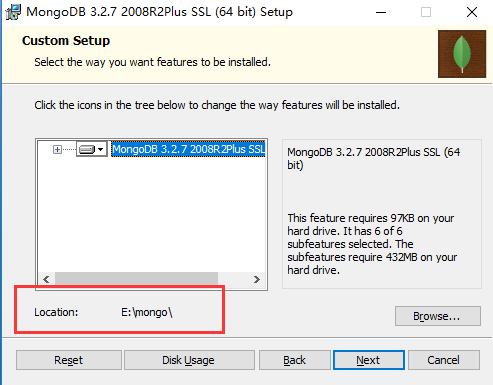
**第一步，下载mongodb,官网：https://www.mongodb.com**

**第二步，在电脑上，创建一个mongo的目录，用于安装mongodb**

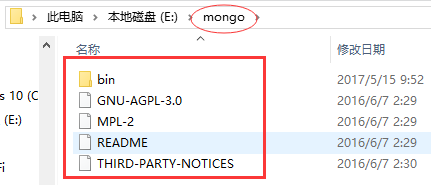


**第三步，安装mongo**

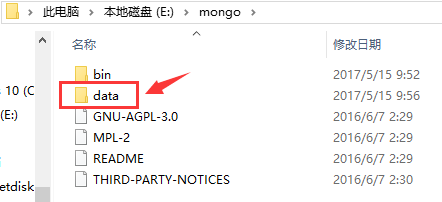




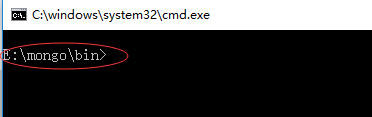
安装完毕后：



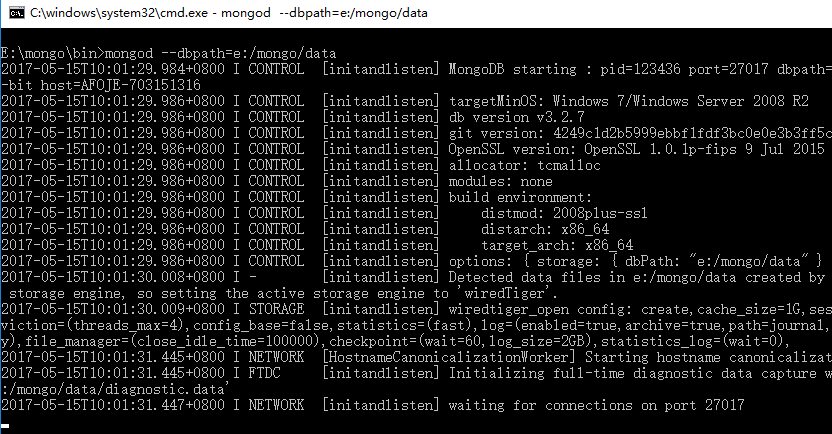
**第四步dada**



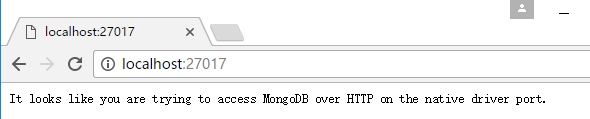
**第五步：在bin目录下面**



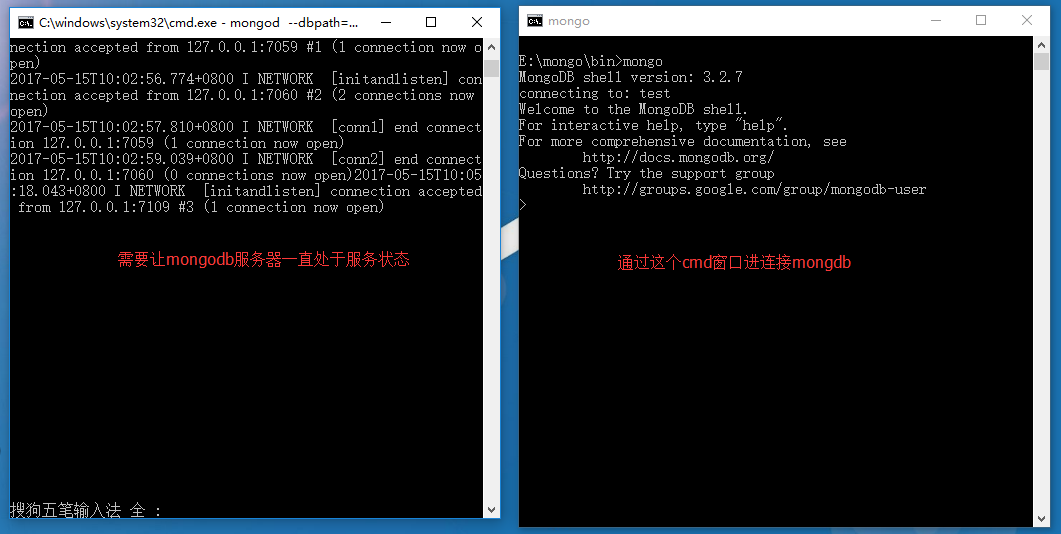
**第六步：启动服务 mongod --dbpath=e:/mongo/data**



可以在浏览器中输入localhost:27017进行访问：



**第七步，连接mongodb, 在打开一个cmd窗口，使用mongo命令连接mongodb**



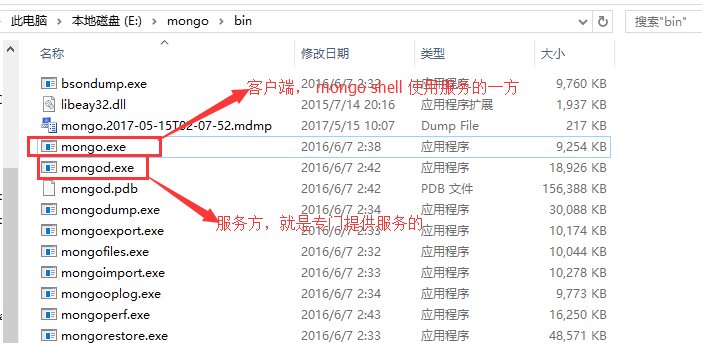
## 2，安装的注意事项

1，安装时，一定要注意到安装的位置

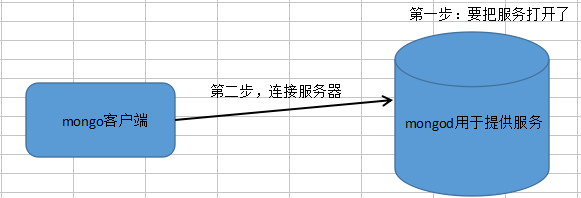
2，必须手动创建data目录

3，对于bin目录，它里面的放的是可执行文件，一般是.exe结尾。

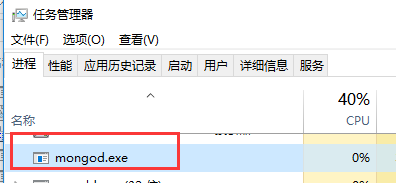
4，对于mongodb，有一个客户端，也有一个服务端。都在bin下面



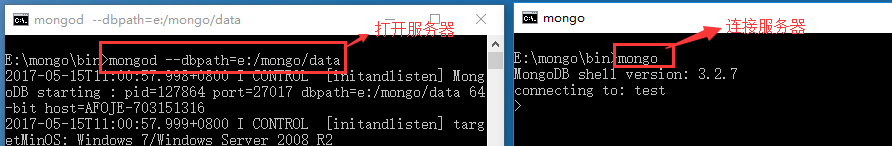
如下图：



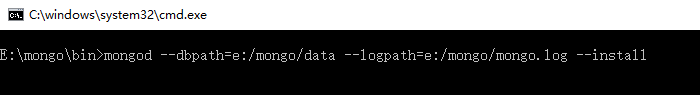
5，当打开服务器时，使用的命令是 mongod --dbpath=e:/mongo/data, 第一点，mongod和--之间有一个空格，第二点，在=两边不能出现空格。这时在任务管理器中就有了这个服务：



6，使用mongo命令去连接服务：



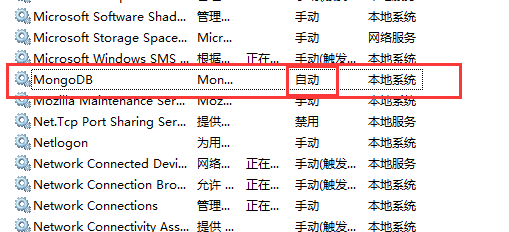
7，我们可以把mongo这个服务作为windows上的服务，命令：mongod --dbpath=e:/mongo/data --logpath=e:/mongo/mongo.log --install：



注意的问题，一定要使用管理员：



作为window的服务：



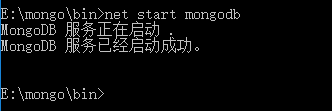
8，一旦作为window的服务，使用时，有两种方式：

开启服务有两种方式：

* 在window服务中，鼠标右键来设置启动这个服务:



* 还有一种方式，使用命令： net start mongodb

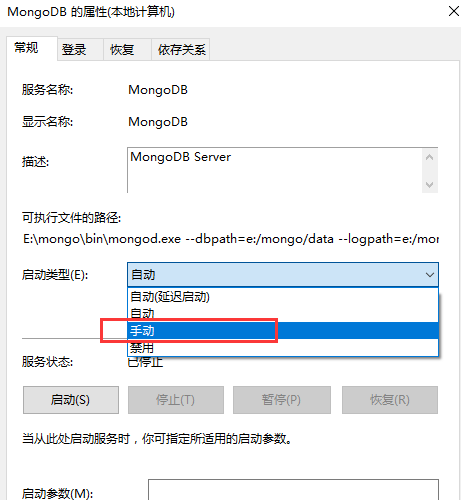


关闭服务有两种方式：

* 在window服务中，鼠标右键来设置启动这个服务:
* 还有一种方式，使用命令： net stop mongodb



9，以后使用mongodb时，要先开启服务，不使用时，关闭服务。把mongodb服务设置成手动启动：



## 3，mongodb中的基本概念

前面，关系型数据库，是以表为作结构，如：



数据库 + 表 + 行

mongodb不是表的结构，是对象的结构。

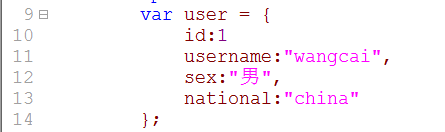
**数据库** + **集合** + **文档**

数据库 + 多个对象 + 对象

在mongodb中它的核心是文档，文档说的直观一点，就是对象，它是面向文档。

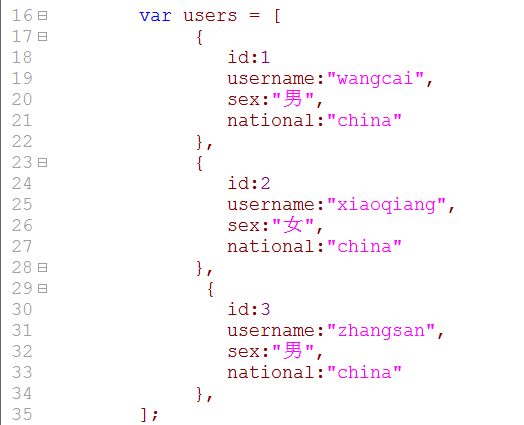
举例：

现在要描述一个用户的信息，包括ID，姓名，性别，国籍



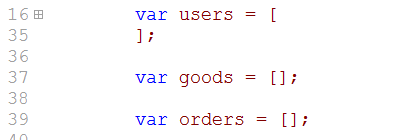
上面的这个对象，就是一个文档。

现在要描述多个用户，包括ID，姓名，性别，国籍



上面我们描述了多个用户的信息，这就是集合。

需求又变了，在一个系统中，需要多个集合：



上面我们描述了多个集合，这就是数据库。

**结论：在mongodb中，有三大核心概念：数据库 + 集合 + 文档，文档是核心，也是基础。**

## 4，一些入门的指令

mongodb **shell**

说的直观一点，就是上午我们所见到的cmd黑窗口。是指命令解析器。

**配置环境变量：**

上午，我们执行命令都是在bin目录下面，现在想在任何目录都要使用bin目录下面的命令，这个时候，我们可以把bin目录，配置到环境变量中。如何配置？

答：把bin目录的路径放在环境变量中的系统环境变量中的path中，这样，当你在任何一个目录输入相关的指令时，如果当前的目录中没有这个指令，它就会去你的path中指定的目录中去找这个命令，就可以找到这个命令，然后执行这个命令。

**使用mongodb的步骤：**

要使用数据库的指令，第一步，肯定要开启数据库服务，有两种开启方式，一个是在windows服务中开启，一个通过命令开启。第二步，就是去连接mongodb服务器。第三步，写指令。

**数据库相关的：**

* show dbs
* use dbname
* db.dropDatabase()

**集合相关的：**

* show collections
* db.createCollections(name)
* db.集合名.drop()

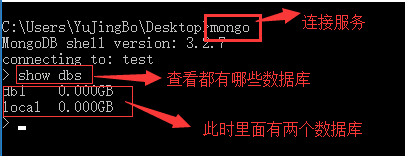
**文档相关的：**

* db.集合名.find()
* db.集合名.insert()
* db.集合名.remove()
* db.集合名.update()

### （1）数据库相关的：

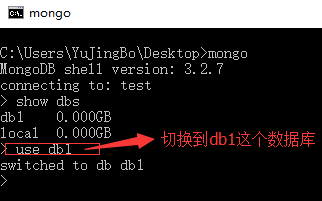
**show dbs 查看都有哪些数据库**

show 是显示的意思 db 是database的意思 s是复数

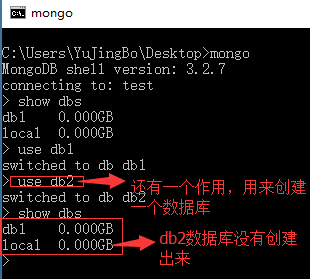


**use dbname（数据库的名字） 切换到哪一个数据库**

user是使用的意思 dbname是一个数据库的名字



**use 还有一个作用，用来创建一个数据库**

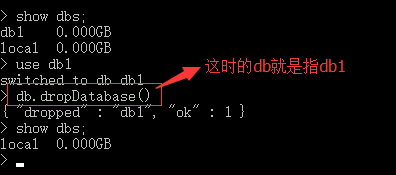


**为什么同有创建出来这个数据库呢？**

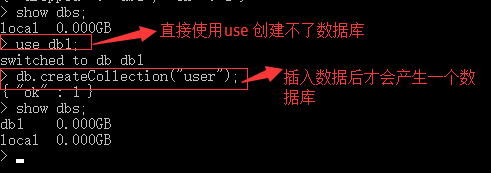
原因：当使用use去创建一个数据库，因为此时它里面没有插入真正的数据，当离开时，这个数据库就会被废掉。

**db.dropDatabase() 删除数据库**

drop是删除的意思



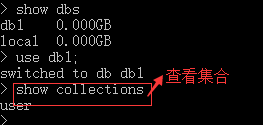
**练习时：**



### （2）集合相关的：

**show collections 查看集合的**

collections 是集合的意思



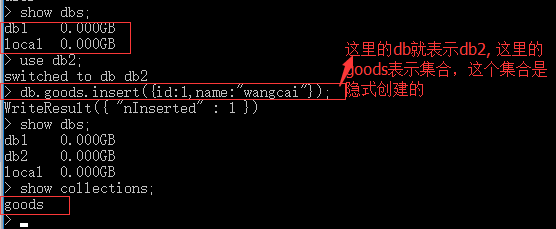
**db.createCollections(name) 创建集合**

create是创建的意思

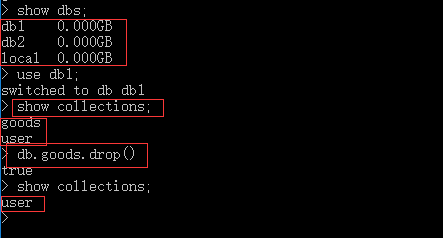
**显式创建集合**



**隐式创建集合**



**db.集合名.drop() 删除一个集合**



说明：

1，db表示当前的数据库

2，从数据库到集合，再到文档，它们之间是通过对象的语法来访问，如：db1.goods.iphone 表示db1这个数据库中的goods集合中的iphone文档。

3，数据库就是一个对象，这个对象中有很多属性，这些属性就是集合，集合也是一个对象，里面也有很多属性，这些属性就是文档，文档也是对象，也有很多发属性。

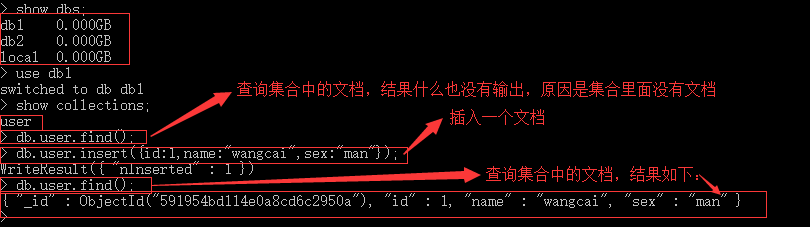
### 文档相关的：

文档就是一个对象，对于对象的操作，也就是CRUD操作：

* **增加 create**
* **查询 read**
* **修改 update**
* **删除 delete**

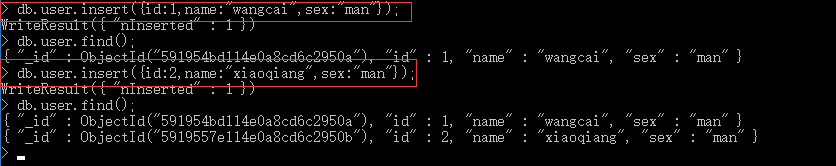
**db.集合名.find()**

find是查询的意思



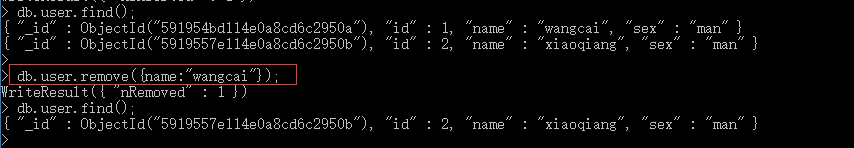
**db.集合名.insert()**

insert表示插入



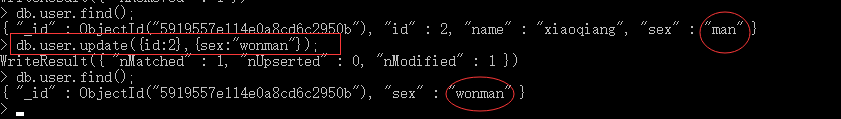
**db.集合名.remove()**

remove是移除的意思



**db.集合名.update()**

update是更新的意思，在这里就是修改的意思



## 5，MongoDB的可视化操作

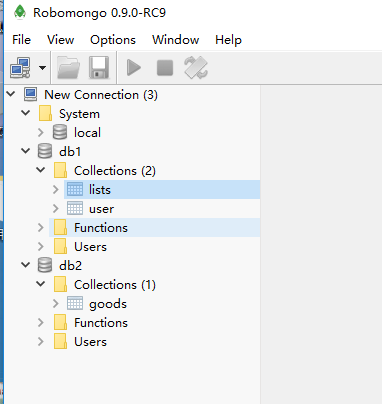
有些人，专门开发了一个工具，用来可视化管理mongodb，不需要命令，直接是可视化操作。在window平台上也很多，自已可以上网找。

下载了一个：



对于这些工具的安装，类似于QQ，下一步，下一步.......

界面如下，自己摸索摸索就可以了。



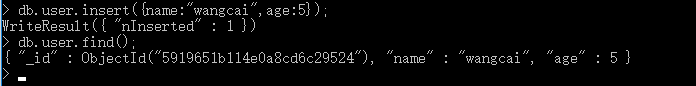
学习是以命令为主。

# 三，MongoDB的增删改查

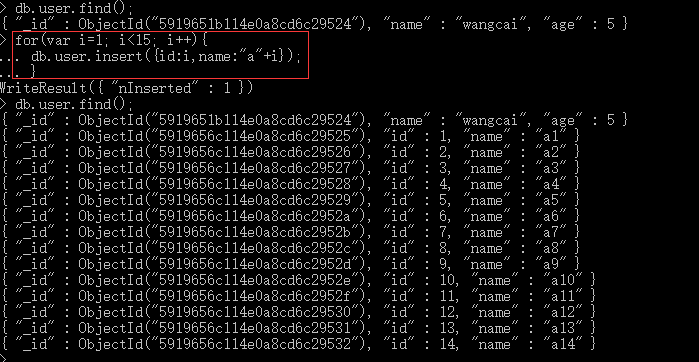
## 1，添加

插入一个文档使用insert(), 也可以批量插入

单条插入：



批量插入：



## 2，查询

## 简单查询

find

格式：db.集合名.find({条件},{返回的键});

返回的键的格式为

键：0或者1

键：0 (代表不显示)

键：1（代表显示）

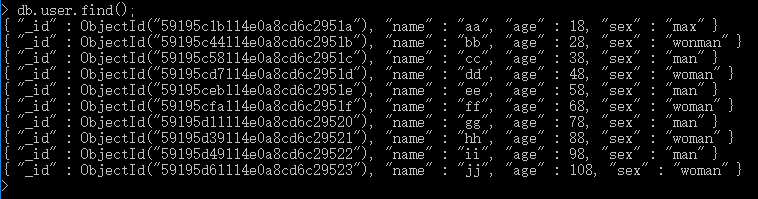
注意：

1，如果说没有指定条件，则返回所有的文档

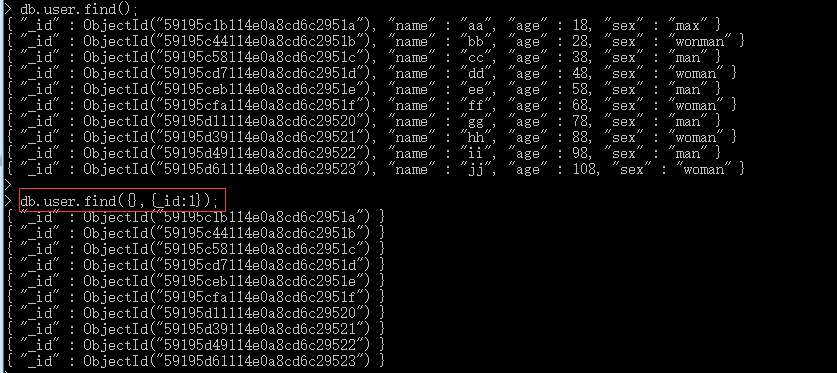
2，如果说没有指定返回的键，就返回该文档中所有的键

代码举例：

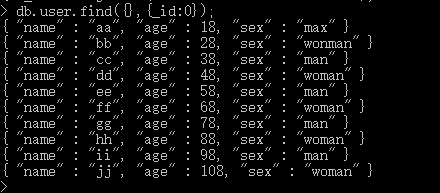
在find里面没有条件和返回的键：



如果find({},{\_id:1})，表示返回的结果中要包含这个键：



如果find({},{\_id:0})，表示返回的结果中要过滤掉这个键：



**为什么这么设计？**

当有海量数据时，查询是有时间开销的，为了提高性能，我们就返回我们需要的信息就可以了，不需要返回所有的信息。

**现在要获取name=”aa”的键值对？ 如下：**

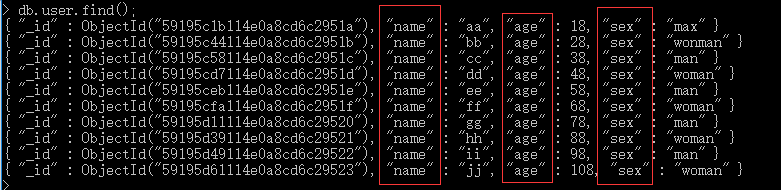


## 比较查询

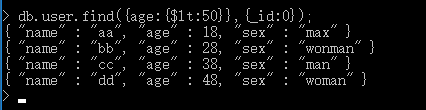
在mongodb中，提供了一些比较相关的操作符：

* $lt < less than 小于
* $lte <= less than and equal 小于等于
* $gt > greater than 大于
* $gte >= greater than and equal 大于等于
* $ne != not equal 不等于

准备的数据如下：



案例1： 将年龄小于50岁用户查询出来



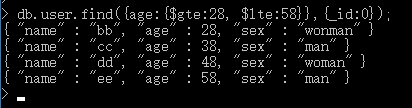
注意：find的格式：db.集合名.find({条件},{返回的键}); 在条件中，键名是外面，比较操作符是写在里面，以对象的形式。如：db.user.find({age:{$lt:50}},{\_id:0});

## 逻辑查询

* or
* and
* not
* in

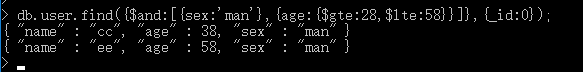
and查询表示同时满足多个条件

**案例2：查询年龄在28到58之间**



注意：在操作符这个对象中，多个键值对是以逗号隔开的，表示同时满足多个条件。

**案例3：查询年龄在28到58之间，并且是男人**



注意：这里面我们使用了$and, $and需要写在最外面，它对应的值是一个数组，在数组中就是简单查询或比较查询了。

**案例4：查询age为18岁或为28的用户**



注意：$or也是写在外面，多个条件时，也是使用数组，数组里面按照正常的写法就行。

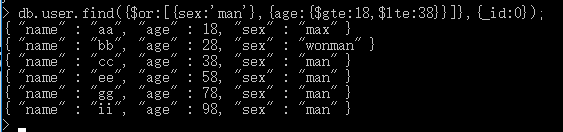
对于上面的写法，还有一种方式，就是使用$in



**问题：$or $in是否能够完全互换？**

如果条件是不同的键，只能用$or, 如果就一个键，可以互换。

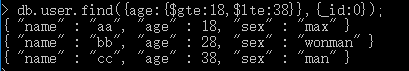
**案例5：当查询sex为man 或 age 在18~38之间这个时候，这个时候，只能用$or**



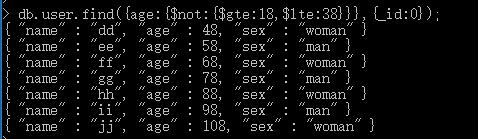
**案例6：查询age在18~38之外的用户**

使用$not

先查询在18~38之内的用户：



然后在上面的基础上，使用$not:



**复习：**

**1，什么是数据库？**

数据库是用来存储数据的，可以从两个方面理解：

a，有组织的存放数据

b，对外提供一些操作规则，用来操作数据库，包括增删改查

**2，关系型数据库和非关系型数据库的区别？**

关系型数据库的核心是二维表， 如：mysql sqlserver oracle...

非关系型数据库的核心是文档，文档就是一个json对象，如：mongodb

**3，mongodb中的三大概念？**

数据库 + 集合 + 文档

**4，入门命令？**

**数据库相关的指令**

show dbs; // 查看都有哪些数据库

use dbname; // 切换到哪个数据库， 创建一个数据库

db.dropDatabase(); // 删除数据库

**集合相关的指令**

show collections; // 查看都有哪些集合

db.createCollection(name); // 创建一个集合

db.集合名.drop(); // 删除一个集合

**文档相关的指令**

db.集合名.insert(); // 插入一个文档

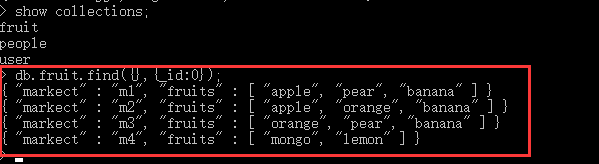
db.集合名.remove(); // 删除一个文档

db.集合名.update(); // 修改一个文档

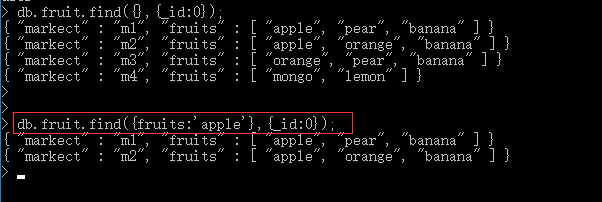
db.集合名.find(); // 查看都有哪些文档

## 数组查询

有如下的数据：



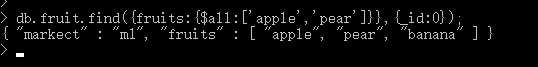
**案例7：查询出售apple的水果店**



可以把{markect:”m1”,”fruits”:[“apple”,”pear”,”banana”]}理解成：

{markect:”m1”, ”fruits”:“apple”, ”fruits”:”pear”, ”fruits”:”banana”]}

**案例8：查询既卖apple又卖pear的水果店**



$all是一个操作符，它的值是一个数组，整体需要构成一个对象，作为fruits的值。

**案例9：查询第二种水果为ornge的水果店 key.index**



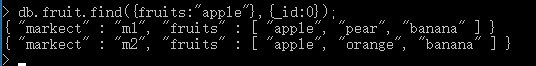
注意：fruits.1也需要加上双引号。

**案例10：查询只买两种水果的水果店 $size**

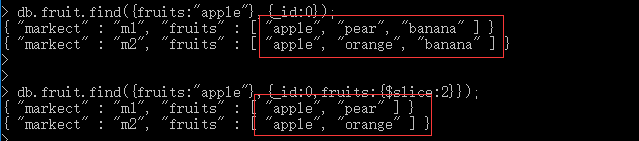


**案例11：查询出售apple的水果店，但结果只取前两种水果**

第一步，查出出售apple的水果店：



第二步，在第一步的结果之上只取前两种水果： $slice



**小结：**

$all

$size

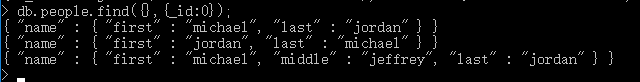
$slice

key.index

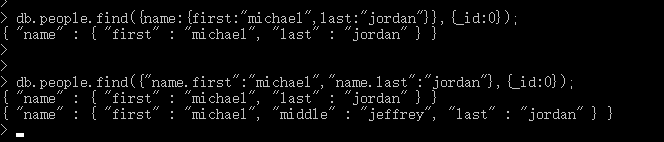
## 内嵌文档查询

文档就是一个json对象 ，内嵌文档就是键值对中的值是一个文档。

准备数据如下：



**案例12：查询姓名为michael jordan的人**

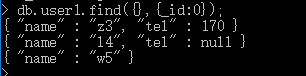


注意：当使用了.时，键也需要加上引号。

## null查询

null也是比较特殊的，可以表示当前这个键为null, 也可以表示没有这个键

准备数据：



**案例13：查询tel为null的文档：**



对于上面的结果是有问题，怎么解决？

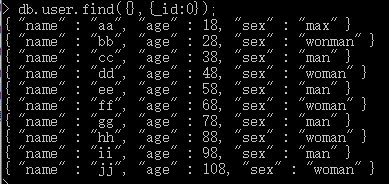
需要通过$exists来判断： $exists需要配合$in来使用



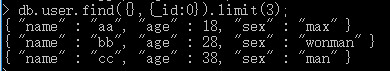
## 限定查询结果

limit skip

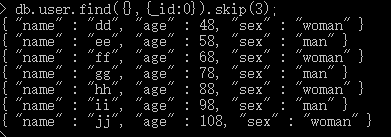
准备数据：



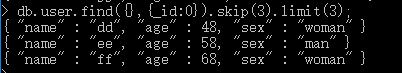
**案例14：查询出前三个文档**



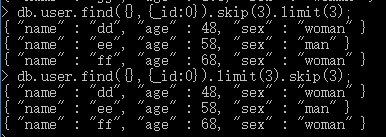
**案例15：查询出除了前三个文档后面的所有文档**



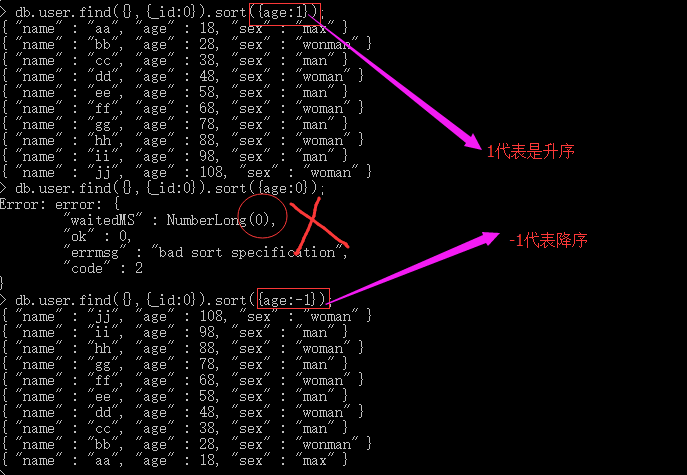
**案例16：查询出前三个后面紧挨着的三个文档**



考虑：skip和limit有没有先后顺序？



**案例17：按照年龄进行排序**



如果按照姓名或性别进行排序，那它是按照字母的ASCLL码进行排序。

注意：当按照年龄进行排序时，如果年龄一样，我们可以这样写：

.sort({age:1,sex:1});

## 3，更新

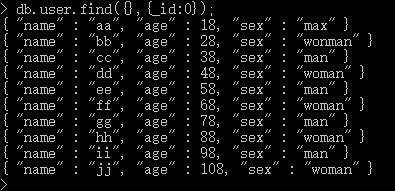
### update

格式: db.集合名.update(参数1，参数2，参数3，参数2);

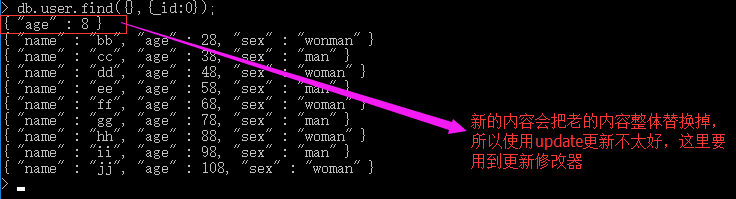
* 参数1：更新条件，同查询条件
* 参数2：具体更新有内容
* 参数3：是一个布尔值，表示如果不存在update的文档，是否插入这个文档，默认是false，不插入
* 参数4：是更新查询所有的文档，true表示更新所有的，false表示只更新第一条记录，默认是false.

重点关注参数1和参数2。

**准备数据：**



**案例18：把name为aa的age换成8**



注意的问题：

1，默认情况下只更新满足条件的第一个文档

2，在使用update更新时，默认是替换操作，会把老的内容全部替换成新的内容。

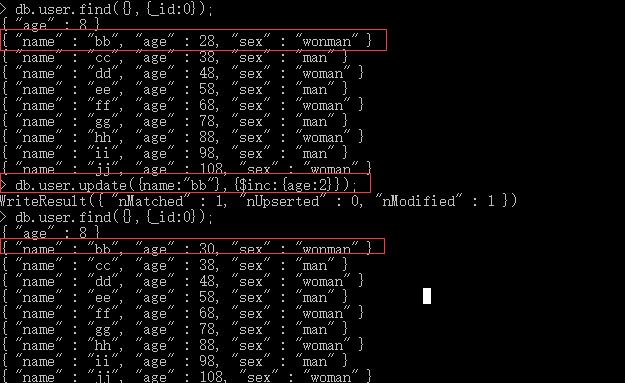
### 更新修改器

对于上面的update操作，有两个问题，有时候我们只对一个文档中的某个键作更新，这个时候使用update不合适，此时就需要使用更新修改器。

更新修改器，只更新一个文档中的一部分。

* $inc increment **增加**
* $set set 是集合，**设置**
* $unset 删除
* $push 向数组中增加一个
* $pushAll 向数组中增加多个（建议不要使用，因为在3.6版本中会删除这个指令）
* $pop 删除数组最后一个或第一个
* $pull 从数组中删除指定的一个
* $pullAll 从数组中删除多个
* $addToSet 向数组中添加一个内容

**案例19：把name为bb的age换成30，但是不能替换，意思就是name:”bb”,仍然存在**

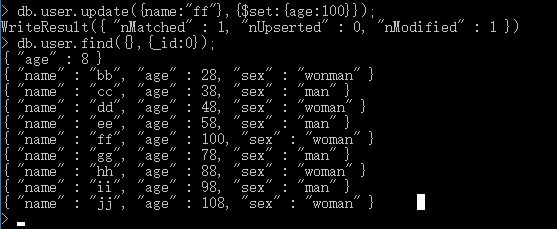


注意：

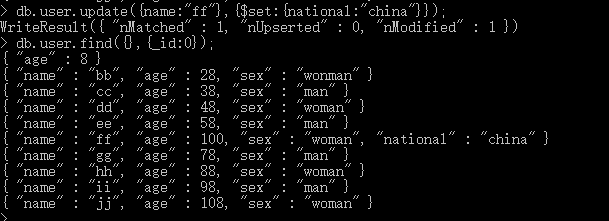
1，使用更新修改器后，没有修改的键还是存在的

2，增加时，使用$inc, 如果减少，可以使用负值

**案例20：设置name为ff的用户，age设置为100**



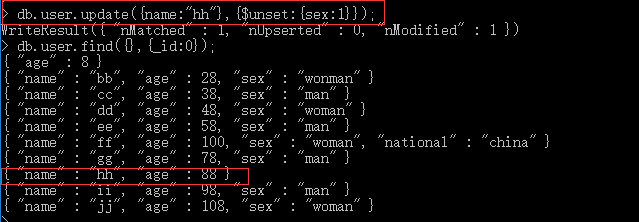
设置ff的国籍，原数据中是没有的，这个时候去设置，就以添加处理：



注意的问题：

1，$set有两个作用，一个是设置，如果找不到设置的键，就添加处理

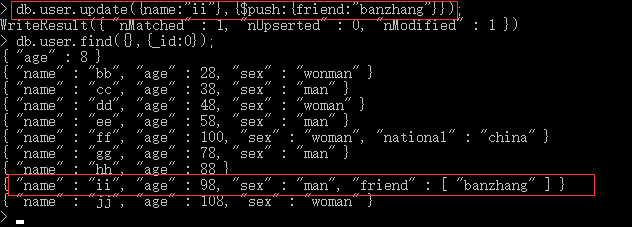
**案例21：删除name为hh的sex**



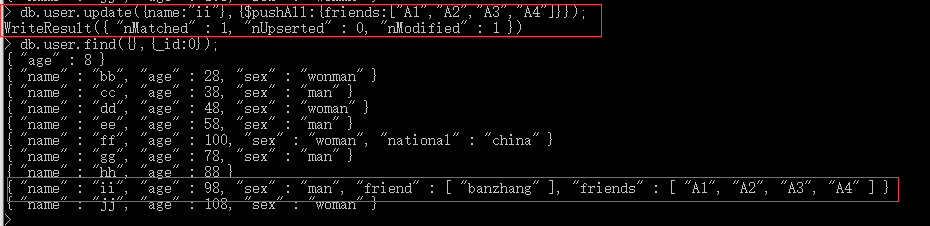
注意的问题：

1，在使用$unset时，只需要指定键名，但是语法上一定要以键值对形式存在，所以要给它一个值，设为1就会。

**案例22：给name为ii的这个用户添加一个朋友，朋友的name叫banzhang**



现在要添加多个朋友，如下：

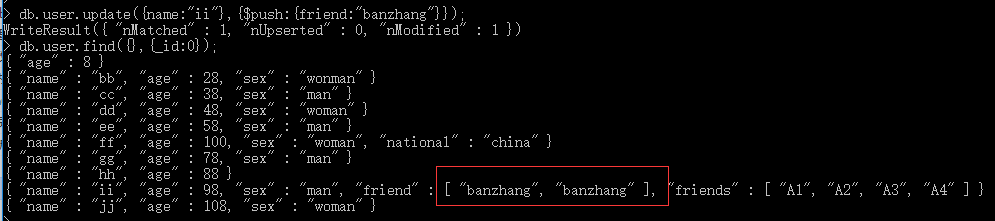


注意的问题：

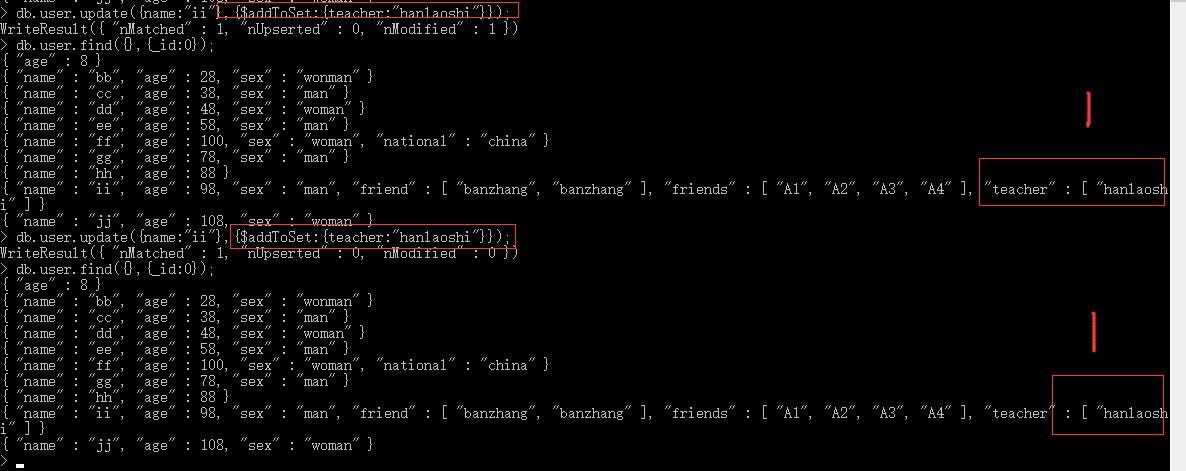
1，push只能添加一个

2，添加的数据最终是以数组的形式

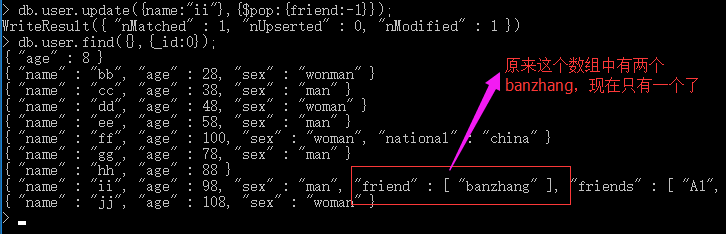
3，使用push时，可以插入重复的内容，如：我们再插入一个banzhang:



为了解决上面可以重复插入这个问题，我们可以使用$addToSet:



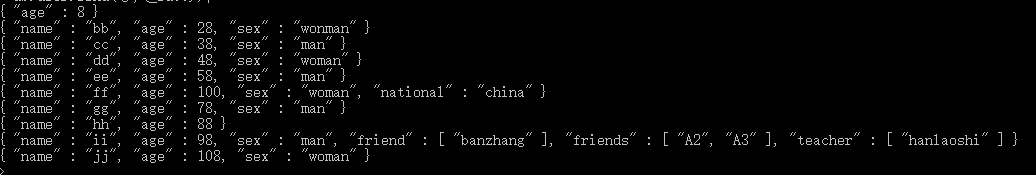
**案例23：给name为ii的这个用户的friend中最后一个banzhang删除掉**



注意的问题：

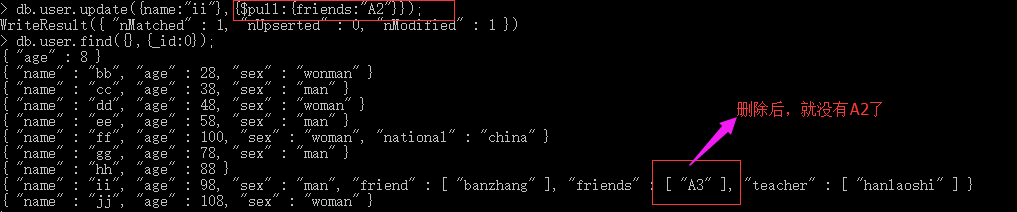
1，对于$pop来说，可以删除第一个，也可以删除最后一个，1表示删除最后一个，-1表示删除第一个

现在的数据如下：



在name为ii这个用户中有一个friends这个键，它对应了A2, A3

**案例24：给name为ii的这个用户的friends中A2 $pull**



假如想删除多个： {$pullAll:{friends:[‘A2’,’A3’]}}

### save

在update语句是，后面的四个参数:

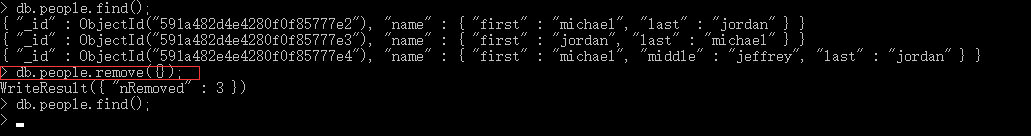
* **参数1：**更新条件，同查询条件
* **参数2：**具体更新有内容
* 参数3：是一个布尔值，表示如果不存在update的文档，是否插入这个文档，默认是false，不插入
* 参数4：是更新查询所有的文档，true表示更新所有的，false表示只更新第一条记录，默认是false.

db.集合名.save() 相当于update语句中参数3为true,参数4为false

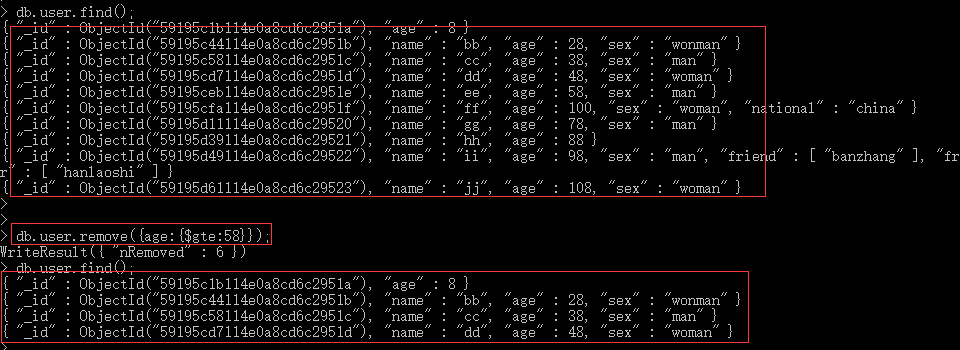
## 4，删除

格式：db.集合名.remove({条件});

删除所有：



删除指定条件的：

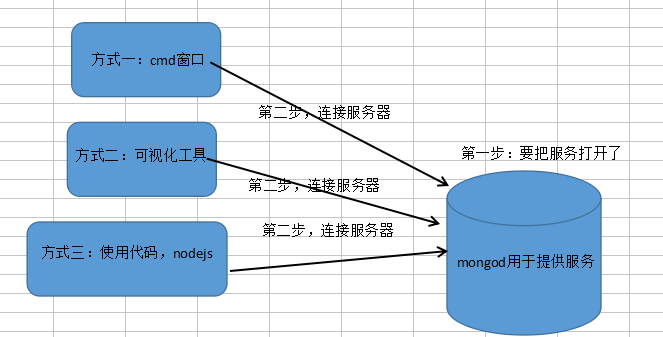


# 四，node.js操作MongoDB

## 1，提供服务方和使用服务方

在bin目录下面有一个mongod.exe是提供服务的，在bin目录下面还有一个mongo.exe它是使用服务的。

使用服务时，有多种方式：



使用服务是没有权限是停止服务的，只能去连接或断开连接服务。

## 2，安装驱动

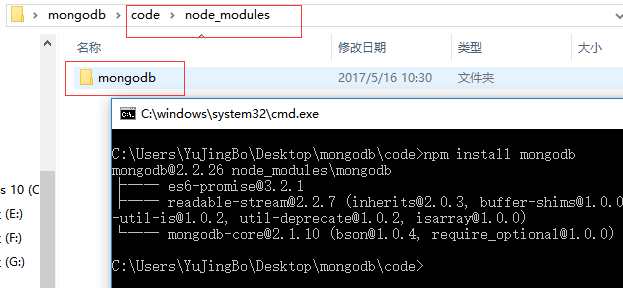
我们使用nodejs去使用服务，使用服务，我们需要去安装一个驱动。

驱动就是一个模块，想要通过nodejs使用mongodb，那么就要安装一个驱动模块：

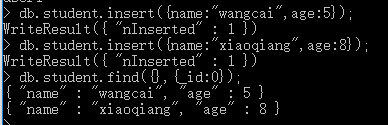


使用步骤：

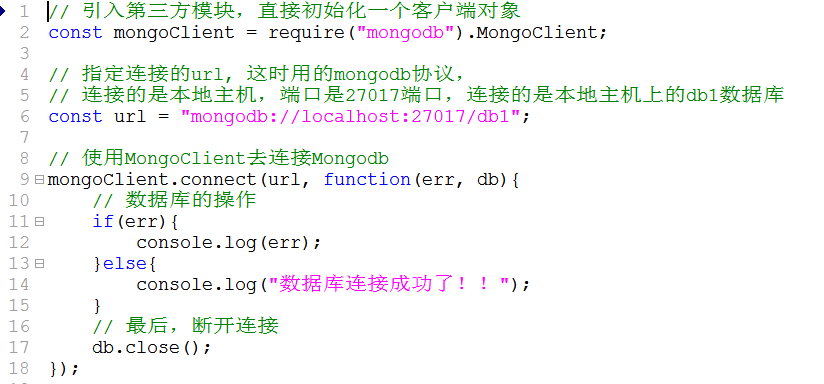
**第一步，安装驱动：**



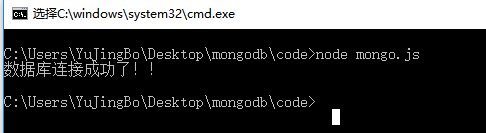
**第二步，准备点数据：**



**第三步，写代码：**

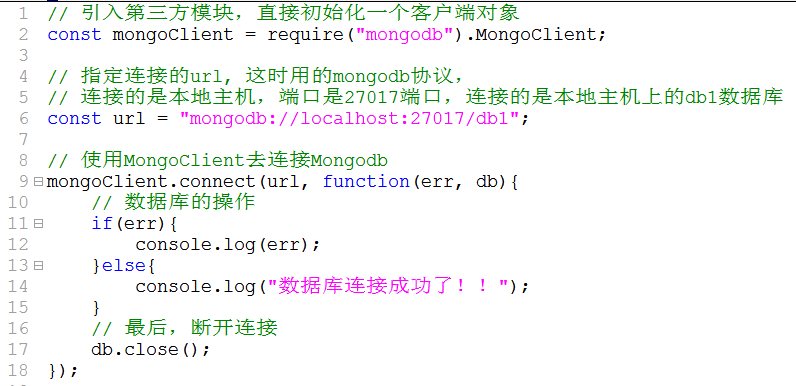


**第四步，执行代码：**



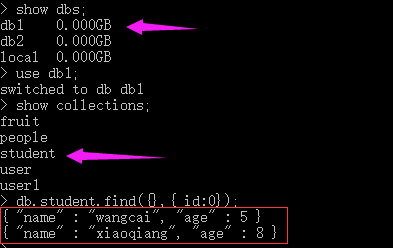
## 3，常见操作

### 连接和关闭



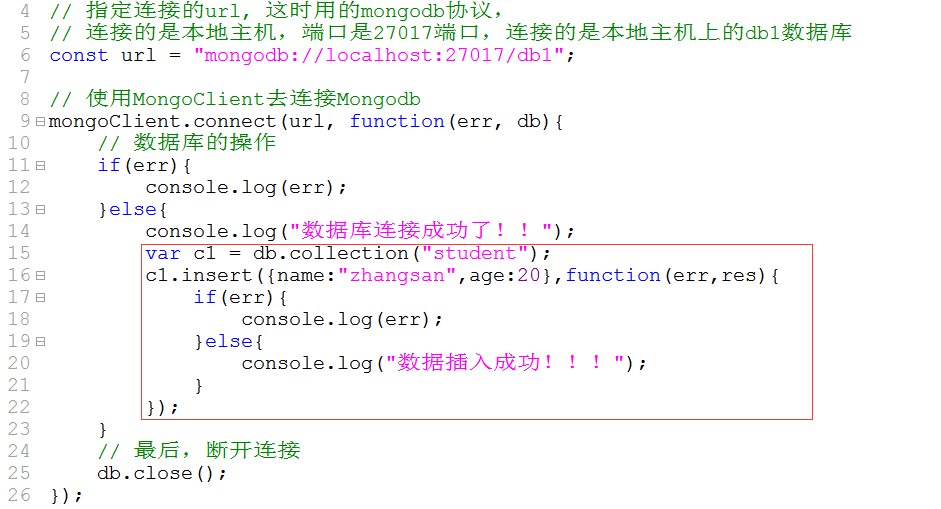
### 增加

首先看一下，我们的数据库中的数据，如下：

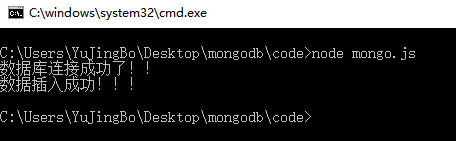


目前在我们的数据库中有两个文档。

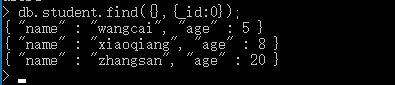
使用代码往数据库添加一个文档：



执行代码：



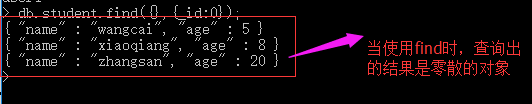
看一下数据库有没有这个文档：



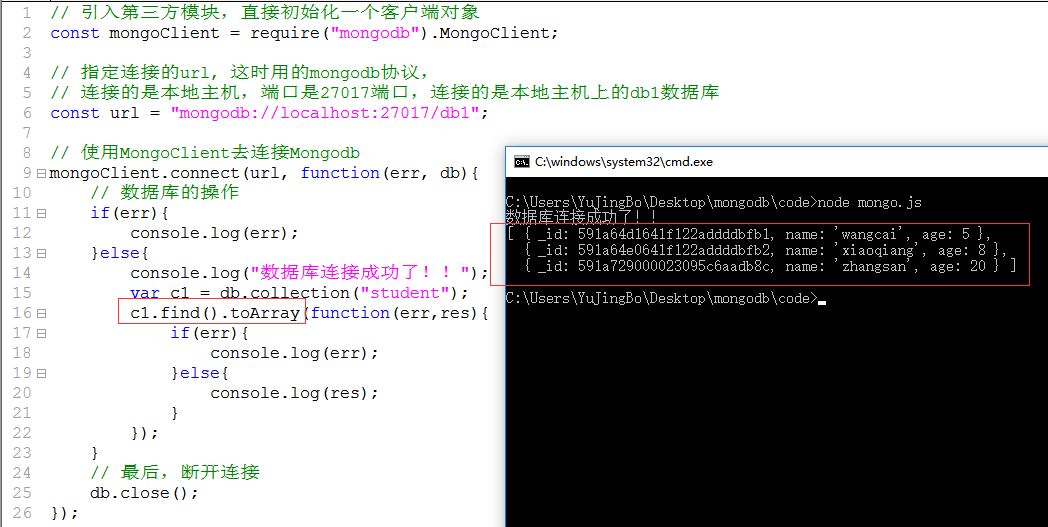
我们也可以使用insertMany([{},{},{}],function(){}); 插入多个文档。

### 查询

当使用find查询时，结果是零散的对象，使用不方便：



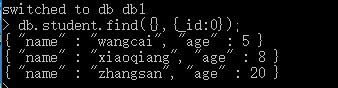
可以使用toArray把结果放到数组中，可能通过索引来使用里面的对象了：



### 更新

更新的意思就是修改.

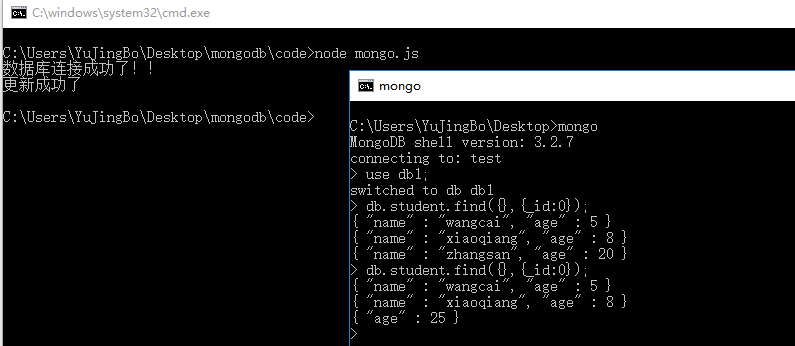
再看一个数据库的数据：



现在，需要把zhangsan年龄增加5岁：



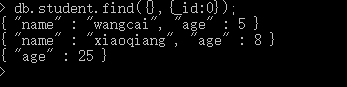
执行代码：



注意的问题，更新操作，新的内容会把原来的所有内容都替换了。

### 删除

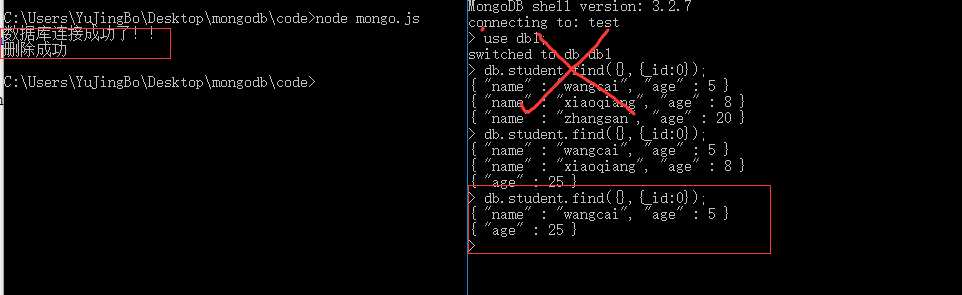
目前数据库中的数据，如下：



把xiaoqiang这个文档删除：



**执行代码：**



# 五，MongoDB中的数据类型

在MongoDB中，文档就是JSON对象，对象由键值对构成，这个键的数据类型永远是字符串，它的值可能是不同的数据类型。

说白了，MongoDB中的数据类型和JS中的基本一致。

## 1，基本数据类型

**1，基本数据类型**

字符串

数值

布尔

undefined

null

## 2，引用数据类型

**2，引用数据类型**

**对象**

**对象id**

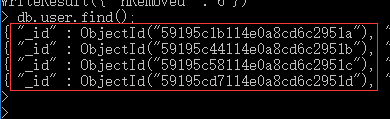
日期

function

正则

**数组**

**对象id: 作用：用来标识文档的惟一性。**



# 六、使用mongoose连接数据库

第一行代码：引入mongoose

第二行代码： 连接数据库

**var** mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/test');

var db = mongoose.connection;

db.on('error', console.error.bind(console, 'connection error:'));

db.once('open', **function**() {

*// we're connected!*

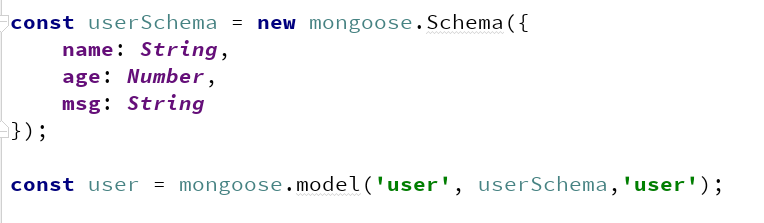
});

第一行代码：连接数据库的实例；

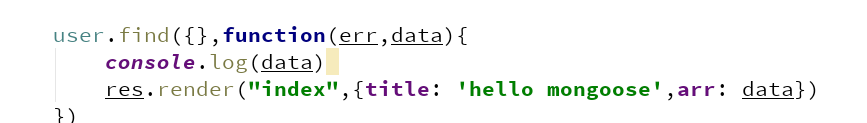
第二行：错误处理；

第三行：连接成功；

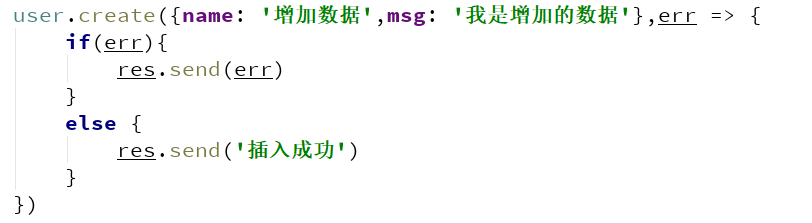
连接成功之后，我们要对我们需要操作的每个集合，显性的声明一个schema 然后利用这个schema注册一个模型。我们的所有操作，都在这模型上进行。



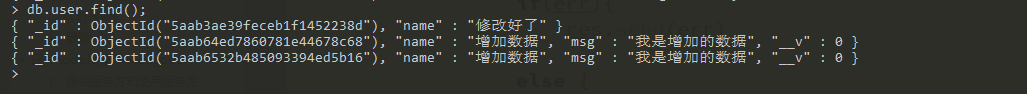
## 查询操作



## 插入操作



操作完成后，我们查看我们的数据库



发现插入操作，为我们多增加了一个字段\_\_v: 0。这个字段，是mongoose用于版本锁定的。如果不想要这个数据，那么，就在声明schema的时候，加上一个versionKey = false;

