**#安装mongodb**

-傻瓜式安装

-配置环境变量

-在c盘 根目录 下建立 data/db 文件夹 默认是这个路径 用来存储数据

（更改默认存储地址 mongod --dbpath 路径名 --port 端口号）

-在命令行输入

mongod 来启动服务器

#数据库

-数据库的服务端

用来存储数据

mongod 来启动服务器

-数据库的客户端

用来操作数据 crud

mongo 来启动客户端

#将mongodb设置为后台服务 开机自启动

1.在c盘创建data目录

data下创建db 和log 目录

2.在C:\Program Files\MongoDB\Server\3.6 目录下导入一个mongod.cfg

在文件中添加

systemLog: destination: file path: c:\data\log\mongod.log storage: dbPath: c:\data\db

3.以管理员的身份开启命令行

4.执行如下命令

sc.exe create MongoDB binPath= "\"C:\Program Files\MongoDB\Server\3.6\bin\mongod.exe\"

--service --config=\"C:\Program Files\MongoDB\Server\3.6\mongod.cfg\"" DisplayName= "MongoDB" start= auto

sc.exe create MongoDB binPath= "\"C:\mongodb的bin目录\mongod.exe\"

--service --config=\"C:\mongodb的安装目录mongod.cfg\"" DisplayName= "MongoDB" start= auto

5.启动任务管理器

在服务中找到MongoDB 手动启动服务器 成功

6.如启动失败

打开命令行窗口 输入sc.exe delete MongoDB

删除之前的配置 从头来过

#基本概念

database（数据库）

collection（集合）

document（文档）

**#基本指令**

1.查看数据库的个数

show dbs

show databases

2.进入到指定的数据库

use 数据库名

3.查看当前数据库

db 代表当前数据库 （相当于java中的this关键字）

4.查看当前数据库中的集合

show collections

#数据库的增删改查操作

1.向数据库中插入一个或多个文档

db.<collection>.insert(doc);

例：向col 集合中插入一个学生

{name:"张三",age:18}

db.col.insert({name:"张三",age:18});

插入多个

db.col.insert([

{name:"张三1",age:18},

{name:"张三2",age:18},

{name:"张三3",age:18}

]);

当插入时没有输入\_id 系统会自动生成一个 以作为每一个文档的唯一标识

\_id可以自己指定 就不会自动添加了 但也要确保其唯一性

-更加清晰

[db.collection.insertOne()](https://docs.mongodb.com/manual/reference/method/db.collection.insertOne/" \l "db.collection.insertOne) 必须传一个

[db.collection.insertMany()](https://docs.mongodb.com/manual/reference/method/db.collection.insertMany/" \l "db.collection.insertMany) 必须传数组

2 查询.db.<collection>.find();

db.col.find();

find() 查询所有符合条件的文档

find 可以接收一个对象 用来条件查询 （返回的是一个数组）

db.col.find({name:"张三"}) ：查询属性是固定值的字段

db.col.find({name:"张三"，age:28}) ：查询两个字段符合的 相当于and

db.col.findOne({name:"张三"，age:28}) ：查询符合条件的第一个文档 （返回的是一个文档对象）可以直接

db.col.findOne({name:"张三"，age:28}).name 返回文档的属性值

db.col.find({name:"张三"}) .count() 返回数量

3.修改db.<collection>.update(查询对象，新对象);

-这种方式有风险

db.col.update({name:"张三"}，{age:333}) 会替换以前的属性 4.0以后会报错

如果有查询到相同的姐股票 update只改第一个 （默认与[updateOne()](https://docs.mongodb.com/manual/reference/method/db.collection.updateOne/" \l "db.collection.updateOne) 一样）

-如果要修改指定的属性 需要使用修改操作符

$set 可以修改制定属性

db.col.update({name:"张三"},{$set:{age:333,gender:"男"}}) 也可以随便添加字段

$unset 可以删除一个属性

db.col.update({name:"张三"},{$unset:{age:333,gender:"男"}}) 也可以随便添加字段

[db.collection.updateOne()](https://docs.mongodb.com/manual/reference/method/db.collection.updateOne/" \l "db.collection.updateOne) *New in version 3.2* 修改一个

[db.collection.updateMany()](https://docs.mongodb.com/manual/reference/method/db.collection.updateMany/" \l "db.collection.updateMany) *New in version 3.2* 修改多个

[db.collection.replaceOne()](https://docs.mongodb.com/manual/reference/method/db.collection.replaceOne/" \l "db.collection.replaceOne) *New in version 3.2 替换一个*

~~db.col.updateOne({name:"张三",{$unset:{age:333,gender:"男"}}}~~

~~db.col.updateMany({name:"张三",{$unset:{age:333,gender:"男"}}}~~

4.删除文档

db.col.remove({name:"张三2"}); 传递条件的方式与find一样 删除全部（与[deleteMany()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteMany/" \l "db.collection.deleteMany) 一样）

如果remove 传递第二个参数true

db.col.remove({name:"张三"},true); 删一个

如果只传递一个空对象

db.col.remove({}); 删除集合中的所有文档

db.col.drop(); 删除这个集合

db.dropDatabase() 删数据库

[db.collection.deleteOne()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteOne/" \l "db.collection.deleteOne) *New in version 3.2*

[db.collection.deleteMany()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteMany/" \l "db.collection.deleteMany) *New in version 3.2*

#文档间的关系

-一对一

1.通过内嵌文档

-一对多（多对一）

也可以通过内嵌文档

-多对多