show dbs:**显示所有**数据库列表

db: **显示当前**数据库对象或集合。

Use db1:**连接**到名为db1的数据库（若没有则**创建数据库**）

在 MongoDB 中默认数据库是：test。 如果您还没有创建过任何数据库，则集合/文档将存储在test数据库中。

db.dropDatabase()：**删除**现有的数据库

db.createCollection(name, options)：**创建集合**，name 是要创建的集合的名称。 options是一个文档，用于指定集合的配置。

show collections：显示创建的集合

在 MongoDB 中，不需要创建集合。当插入一些文档时，MongoDB 会**自动创建集合：**

>db.**newcollection**.insert({"name" : "yiibaitutorials"})

db.collection.drop() 用于从数据库中**删除集合**。

**db.COLLECTION\_NAME.insert(document)：**要插入多个文档只需要在大括号后面加逗号再写大括号

>db.mycol.insert ({

\_id: 100,

title: 'MongoDB Overview',

description: 'MongoDB is no sql database',

by: 'yiibai tutorials',

url: 'http://www.yiibai.com',

tags: ['mongodb', 'database', 'NoSQL'],

likes: 100,

})

不在文档中指定\_id，系统会自动分配主键ID值

**查询：**

**find()：db.collection.find() :查询所有文档（即在mysql中的一条记录）**

> db.mycol.find().pretty()：格式化方式显示查询结果

findone()：查询一个文档

| **操作** | **语法** | **示例** | **RDBMS等效语句** |
| --- | --- | --- | --- |
| 相等 | {<key>:<value>} | db.mycol.find({"by":"yiibai"}).pretty() | where by = 'yiibai' |
| 小于 | {<key>:{$lt:<value>}} | db.mycol.find({"likes":{$lt:50}}).pretty() | where likes < 50 |
| 小于等于 | {<key>:{$lte:<value>}} | db.mycol.find({"likes":{$lte:50}}).pretty() | where likes <= 50 |
| 大于 | {<key>:{$gt:<value>}} | db.mycol.find({"likes":{$gt:50}}).pretty() | where likes > 50 |
| 大于等于 | {<key>:{$gte:<value>}} | db.mycol.find({"likes":{$gte:50}}).pretty() | where likes >= 50 |
| 不等于 | {<key>:{$ne:<value>}} | db.mycol.find({"likes":{$ne:50}}).pretty() | where likes != 50 |

>db.mycol.find(

{

$and: [

{“key1”: value1}, {“key2”:value2}

]

}

).pretty()：

等价于 slect \* from mycol where key1 = value1 and key2= value2 ，用逗号隔开等同于and

Or 语句同上面and一样的写法

And 和or一起：>db.mycol.find({"likes": {$gt:10}, $or: [{"by": "yiibai tutorials"},

{"title": "MongoDB Overview"}]}).pretty()

要在**嵌入/嵌套文档**中的字段上指定查询条件，请使用点符号(“field.nestedField”)。

例如：查询size字段中的uom字段为“in”的：db.inventory.find( { "size.uom": "in" } )

其中size查出来为：{ h: 8.5, w: 11, uom: "in" }

db.UserInfo.find({userName :/A/})：查询username包含A的，类**似于like**

**选择字段，限制记录条数**

以下实例为显示查询文档中的两条记录：

> db.col.find({},{"title":1,\_id:0})**.limit(2)**

{ "title" : "PHP 教程" }

{ "title" : "Java 教程" }

第二个大括号是结果集只显示设置为1的这个字段，-id为0表示不显示\_id

更新数据，默认只更新一条

> db.mycol.**update(**{'title':'MongoDB Overview'},{$set:{'title':'New Update MongoDB Overview'}})

更新所有符合条件的数据：

>db.mycol.update({'title':'MongoDB Overview'},

{$set:{'title':'New Update MongoDB Overview'}},**{multi:true})**

**save()：**

db.mycol.save(

{

"\_id" : 100, "title":"Update By Save()Method.", "by":"yiibai.com"

}

)

将\_id为100的数据完全替换（只能根据id）

**删除**

> db.mycol.remove({'\_id':100})

只删除一条文档记录

>db.COLLECTION\_NAME.remove(DELETION\_CRITERIA,1)

删除所有：> db.mycol.remove()

**排序**

根据title降序，（升序为1）

> db.mycol.find({},{"title":1,\_id:0}).sort({"title":-1})

给title创建索引

>db.mycol.ensureIndex({"title":1})

**聚合**

> db.article.aggregate([{$group : {\_id : "$by\_user", num\_tutorial : {$sum : 1}}}])

等价于：select by\_user, count(\*) as num\_tutorial from `article` group by by\_user;

多表查询

第一种：

> author=db.authors.findOne({name:"chenzhou"})

db.posts.find({"author\_name":author.name});

第二种：$lookup

https://blog.csdn.net/harleylau/article/details/77899223

但是这两种都是mongodb里面

而不能在java中实现

在java实现要通过在实体类中增加字段去拼凑，这个字段可以是object，也可以是要增加的字段，根据需求来，建议选择object以显示可维护性