## 看看mongodb的基本操作

### mongodb的基本使用包括几个方面

* 增、删、查、改是重点

CRUD

C:insert()

R:find()

U:update()

D:remove()

### 增加数据

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | [root@blog ~]# hostname  blog.unix178.com  [root@blog ~]# mongo  MongoDB shell version: 2.4.6  connecting to: test  > show dbs  admin0.203125GB  local0.078125GB  test0.203125GB  > use zhuima  switched to db zhuima  > info = {Name:"zhuima",Age:26,Gender:"F",Address:"Beijing China",Work:"Engineer",Other:"DevOps"}  {  "Name" : "zhuima",  "Age" : 26,  "Gender" : "F",  "Address" : "Beijing China",  "Work" : "Engineer",  "Other" : "DevOps"  }  > db.peroson.insert(info)  > db.person.find()  > db.peroson.find()  { "\_id" : ObjectId("53cbc52255ea1fd13e47ad9d"), "Name" : "zhuima", "Age" : 26, "Gender" : "F", "Address" : "Beijing China", "Work" : "Engineer", "Other" : "DevOps" }  > |

### 同样我们可以批量插入数据

### 删除数据

### 批量删除数据

### 指定某个条件查找数据

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | > db.peroson.findOne()  {  "\_id" : ObjectId("53cbc67855ea1fd13e47ad9e"),  "Name" : "zhuima1",  "Age" : 1,  "Address" : "1.Number Beijing China"  }  > db.peroson.find({Name:"zhuima888"})  { "\_id" : ObjectId("53cbc67855ea1fd13e47b115"), "Name" : "zhuima888", "Age" : 88, "Address" : "888.Number Beijing China" }  > |

### 指定多重条件查找

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | > db.peroson.find({Age:{$gt:33,$lt:44}})  { "\_id" : ObjectId("53cbc67855ea1fd13e47adbf"), "Name" : "zhuima34", "Age" : 34, "Address" : "34.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc0"), "Name" : "zhuima35", "Age" : 35, "Address" : "35.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc1"), "Name" : "zhuima36", "Age" : 36, "Address" : "36.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc2"), "Name" : "zhuima37", "Age" : 37, "Address" : "37.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc3"), "Name" : "zhuima38", "Age" : 38, "Address" : "38.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc4"), "Name" : "zhuima39", "Age" : 39, "Address" : "39.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc5"), "Name" : "zhuima40", "Age" : 40, "Address" : "40.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc6"), "Name" : "zhuima41", "Age" : 41, "Address" : "41.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc7"), "Name" : "zhuima42", "Age" : 42, "Address" : "42.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47adc8"), "Name" : "zhuima43", "Age" : 43, "Address" : "43.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae23"), "Name" : "zhuima134", "Age" : 34, "Address" : "134.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae24"), "Name" : "zhuima135", "Age" : 35, "Address" : "135.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae25"), "Name" : "zhuima136", "Age" : 36, "Address" : "136.Number Beijing China" }  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae26"), "Name" : "zhuima137", "Age" :  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae2c"), "Name" : "zhuima143", "Age" : 43, "Address" : "143.Number Beijing China" }  Type "it" for more  > |

### 更新数据

|  |  |
| --- | --- |
| 1  2  3  4  5 | > db.peroson.find({Name:"zhuima100"})  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae01"), "Name" : "zhuima100", "Age" : 0, "Address" : "100.Number Beijing China" }  > db.peroson.find({Name:"zhuima100"})  { "\_id" : ObjectId("53cbc67855ea1fd13e47ae01"), "Name" : "zhuima100", "Age" : 100, "Address" : "100.Number Beijing China" }  > |

### 批量更新数据

* 数组修改器
* $push  会向已有的数组末尾加入一个元素，要是没有就会创建一个新的数组

## 看了上面的一些例子，我们来点短小的命令

### 命令行接口

|  |  |
| --- | --- |
| 1  2 | mongo  > |

### 查看所有数据库

|  |  |
| --- | --- |
| 1 | > show dbs |

### 查看数据库中所有的表

* 后面的；可有可无，看个人喜好

|  |  |
| --- | --- |
| 1  2  3 | > show dbs;  > use zhuima;  > show collections; |

### 为数据库下某个表插入数据

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | > j = { name : "zhuima"}  > k = { age : 23}  > m = {sex : "fmale"}  > db.test.insert(j)  > db.test.insert(k)  > db.test.insert(m)  > |

### 查看某个数据库中的数据

|  |  |
| --- | --- |
| 1  2 | > db.test.find() # 默认显示前20行 可以再>后接着输入it显示下一页的20行数据~  > |

### 查看数据库中的某个表的数据第一行

|  |  |
| --- | --- |
| 1  2 | > db.test.findOne()  > |

### 查看数据库中的某个表的前几个

* limit限制输出信息量

|  |  |
| --- | --- |
| 1  2 | > db.test.find().limit(3) 只列出前几行  > |

### 如何显示数据库中所有信息(突破只能显示20行的限制)

|  |  |
| --- | --- |
| 1  2  3 | > var c = db.test.find()  > while (c.hasNext()) printjson(c.next())  > |

### 对数据进行筛选

|  |  |
| --- | --- |
| 1  2 | > db.text.find({x:3}) #筛选出x对应值为3的匹配的行  > |

### 批量插入数据

|  |  |
| --- | --- |
| 1  2 | > for (i =1; i<=25;i++) db.test.insert({x:i})  > |

### 只列出从第几行开始的行（有规律的）

* sort进行排序操作
* $gt是mongodb内置的比较运算的内置变量

|  |  |
| --- | --- |
| 1  2 | > db.test.find({x:{$gt:5}}).sort({x:3})  #列出从第6行开始的行并进行排序  > |

### 只列出某个范围内的行

|  |  |
| --- | --- |
| 1  2 | > db.test.find({x:{$gt:5}}).limit(5) # 只列出第6行到第10行  > |

### 过滤包含某些关键字的筛选

* $in也是mongodb内置的比较运算的内置变量

|  |  |
| --- | --- |
| 1  2  3  4 | > db.inventory.find( { type: { $in: [ 'food', 'snacks' ] } } )  > db.zhuima.find({name:{$in:["zhuima"]}})  { "\_id" : ObjectId("53b272e9c74fabe96af3a054"), "name" : "zhuima" }  > |

### 数组定义

|  |  |
| --- | --- |
| 1 | > i={"date":new Date()} |

### 可以嵌套数组的

|  |  |
| --- | --- |
| 1  2  3 | > name={"name":{"gender":"male"}}  { "name" : { "gender" : "male" } }  > |

### 定义一个名为追马的人的相关信息

|  |  |
| --- | --- |
| 1  2  3 | > info={"name":"zhuima","other":["male","single"],"age":22}  { "name" : "zhuima", "other" : [ "male", "single" ], "age" : 22 }  > |

# 三.创建、更新及删除文档

1创建

>var joe = {"name":"joe","friends":32,"enemies":2}

>db.user.insert(joe)

WriteResult({ "nInserted" : 1 })

2更新

> var joe\_bak = db.user.findOne({"name":"joe"})

> joe\_bak.relationships = {"friends":joe.friends,"enemies":joe.enemies}

{ "friends" : 32, "enemies" : 2 }

> joe.username = joe.name

Joe

> delete joe.friends

true

> delete joe.enemies

true

> delete joe.name

True

> db.user.update({"username":"joe"},joe\_bak)

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

> db.user.findOne()

{

"\_id" : ObjectId("56d2a103d66803eeed7a475b"),

"name" : "joe",

"friends" : 32,

"enemies" : 2,

"relationships" : {

"friends" : 32,

"enemies" : 2

}

}

## 2.1使用更新修改器

### set/unset

> db.page.update({"\_id" : ObjectId("56d2a7b0d66803eeed7a475c")},

... {"$set":{"favorite book":"war and peace"}})

2.1.1用$set修改键的数据类型

> db.page.update({"\_id" : ObjectId("56d2a7b0d66803eeed7a475c")},

... {"$set":{"favorite book":["cat`s cradle","foundation trilogy","ender`s game"

]}})

> db.page.findOne()

{

"\_id" : ObjectId("56d2a7b0d66803eeed7a475c"),

"url" : "www.example.com",

"pageview" : 54,

"favorite book" : [

"cat`s cradle",

"foundation trilogy",

"ender`s game"

]

$unset可以将键完全删除

> db.page.update({"\_id" : ObjectId("56d2a7b0d66803eeed7a475c")},

... {"$unset":{""favorite book":1}})

结果

> db.page.findOne()

{

"\_id" : ObjectId("56d2a7b0d66803eeed7a475c"),

"url" : "www.example.com",

"pageview" : 54

}

使用$set修改内嵌文档，当前，文档user中relationships" : { "friends" : 32, "enemies" : 2}

> db.user.findOne()

{

"\_id" : ObjectId("56d2a103d66803eeed7a475b"),

"name" : "joe",

"friends" : 32,

"enemies" : 2,

"relationships" : {

"friends" : 32,

"enemies" : 2

}

}

> db.user.update({ "\_id" : ObjectId("56d2a103d66803eeed7a475b")},

... {"$set":{"relationships.friends":33}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

> db.user.findOne()

{

"\_id" : ObjectId("56d2a103d66803eeed7a475b"),

"name" : "joe",

"friends" : 32,

"enemies" : 2,

"relationships" : {

"friends" : 33,

"enemies" : 2

}

}

小结：$inc与$set功能相似，但$inc只能用来增加数字:整数、长整形、双精度浮点。用在其他类型会失败。

另外，$inc键的类型必须是数字，不能使其他的字符串、数组或其他类型。

## 2.2使用数组修改器

### push/pops

数组操作，顾名思义是用来应用在值为数组的键上，例如不能对整数进行push和pop。如果指定的键已经存在，则”$push”会在数组末尾加一个元素，要是没有则创建新的数组。例：

> db.blog.findOne()

{

"\_id" : ObjectId("56d1518ea487e246283c7a89"),

"title" : "My Blog Post",

"content" : "Here is my blog post",

"date" : ISODate("2016-02-27T07:34:20.950Z")

}

>

>

> db.blog.update({"title" : "My Blog Post"},{"$push":{"conmments"

... :

... {"name":"joe","email":"joe@example.com","content":"nice post." }

... }})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>

插入后，查询得到：

> db.blog.findOne()

{

"\_id" : ObjectId("56d1518ea487e246283c7a89"),

"title" : "My Blog Post",

"content" : "Here is my blog post",

"date" : ISODate("2016-02-27T07:34:20.950Z"),

"conmments" : [

{

"name" : "joe",

"email" : "joe@example.com",

"content" : "nice post."

}

]

}

> db.blog.update({ "title" : "My Blog Post"},{"$push":{"name":"bob","email":"bob

@example.com","content":"good post."}})

### addToSet/$each

在键值为数组的情况下，想要在数组中添加新的值，则可以用addToSet，如果某元素不再列表中，则可以用其添加进去，如果已经在列表中，则添加不进去。如**blog**:

> db.blog.findOne()

{

"\_id" : ObjectId("56d1518ea487e246283c7a89"),

"title" : "My Blog Post",

"content" : "Here is my blog post",

"date" : ISODate("2016-02-27T07:34:20.950Z"),

"conmments" : [

{

"name" : "joe",

"email" : "joe@example.com",

"content" : "nice post."

},

{

"name" : "bob",

"email" : "bob@example.com",

"content" : "good post."

}

],

"visitors" : [

"joe",

"john"

]

}

用addToSet添加

db.blog.update({"title" : "My Blog Post"}, {"$addToSet":{"visitors":"johndan"}

结果：

> db.blog.findOne()

{

"\_id" : ObjectId("56d1518ea487e246283c7a89"),

"title" : "My Blog Post",

"content" : "Here is my blog post",

"date" : ISODate("2016-02-27T07:34:20.950Z"),

"conmments" : [

{

"name" : "joe",

"email" : "joe@example.com",

"content" : "nice post."

},

{

"name" : "bob",

"email" : "bob@example.com",

"content" : "good post."

}

],

"visitors" : [

"joe",

"john",

"johndan"

]

}

若再次进行添加 "johndan"，则结果不变,**避免了重复数据**。

也可以和“$each”结合起来使用，可一次添加某个键的多个值。

例：> db.blog.update({"title" : "My Blog Post"},{"$addToSet":{"visitors":{"$each":["joxi","joyi","john"]}}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>

> db.blog.findOne()

{

"\_id" : ObjectId("56d1518ea487e246283c7a89"),

"title" : "My Blog Post",

"content" : "Here is my blog post",

"date" : ISODate("2016-02-27T07:34:20.950Z"),

"conmments" : [

{

"name" : "joe",

"email" : "joe@example.com",

"content" : "nice post."

},

{

"name" : "bob",

"email" : "bob@example.com",

"content" : "good post."

}

],

"visitors" : [

"joe",

"john",

"johndan",

"joxi",

"joyi"

]

}