# MongoDB文档操作

【<https://www.runoob.com/mongodb/mongodb-insert.html>】

**前言：**

文档的数据结构（BSON）和JSON基本一样，所有存储在集合中的数据都是BSON格式。

BSON是一种类似于JSON的二进制形式的存储格式，是Binary JSON的简称。

## 插入文档：

MongoDB使用insert()或save()方法向集合中插入文档，语法如下：

|  |
| --- |
| db.collection\_name.insert(document);  或  db.collection\_name.save(document); |

* save()：如果\_id主键存在则更新数据，如果不存在就插入数据。该方法新版本中已废弃，可以使用db.collection.insertOne()或db.collection.replaceOne()来代替。
* insert()：若插入的数据主键已经存在，则会抛出异常，提示主键重复，不保存当前数据。

3.2版本之后新增了db.collection.insertOne()和db.collection.insertMany()。

db.collection.insertOne()用于向集合中插入一个新文档。

db.collection.insertMany()用于向集合中插入一个或多个文档。

## 更新文档：

MongoDB使用update()和save()方法来更新集合中的文档。

### update()方法：

update()方法用于更新已存在的文档。语法格式如下：

|  |
| --- |
| db.collection.update(  <query>,  <update>,  {  upsert: <boolean>,  multi: <boolean>,  writeConcern: <document>  }  ) |

参数说明：

* query：update的查询条件，类似sql update的查询内的where后面的。
* update：update的对象和一些更新的操作符（如$set,$inc）等，也可以理解为sql update查询内set后面的。
* upsert：可选，这个参数的意思是，如果不存在update的记录，是否插入;true为插入，默认是false，不插入。
* multi：可选，MongoDB默认是false，只更新找到的第一条记录，如果这个参数为true，就把按条件查出来的多条记录全部更新。
* writeConcern：可选，抛出异常的级别。
* WriteConcern.NONE：没有异常抛出；
* WriteConcern.NORMAL：仅抛出网络错误异常，没有服务器错误异常；
* WriteConcern.SAFE：抛出网络错误异常、服务器错误异常；并等待服务器完成写操作；
* WriteConcern.MAJORITY：抛出网络错误异常，服务器错误异常；并等待一个主服务器完成写操作；
* WriteConcern.FSYNC\_SAFE：抛出网络错误异常，服务器错误异常；写操作等待服务器将数据刷新到磁盘；
* WriteConcern.JOURNAL\_SAFE：抛出网络错误异常，服务器错误异常；写操作等待服务器提交到磁盘的日志文件；
* WriteConcern.REPLICAS\_SAFE：抛出网络错误异常，服务器错误异常；等待至少2台服务器完成写操作；

### save()方法：

save()方法通过传入的文档来替换已有文档，\_id主键存在就更新，不存在就插入。

语法格式如下：

|  |
| --- |
| db.collection.save(  <document>,  {  writeConcern: <document>  }  ) |

参数说明：

* document：文档数据。
* writeConcern：可选，抛出异常的级别。

【更多实例】

只更新第一条记录：

### 3、注意点：

更新文档时，如果不指定“更新操作符<update>”，那么就是用新的文档替换旧的文档，而不是更新文档中某些值。

更新操作符就相当于SQL中where后面的条件。

## 删除文档：

remove()函数是用来移除集合中的数据。

remove()方法的基本语法格式如下所示：

|  |
| --- |
| db.collection.remove(  <query>,  {  justOne: <boolean>,  writeConcern: <document>  }  ) |

参数说明：

* query：（可选）删除文档的条件；
* justOne：（可选）如果设为true或1，则只删除一个文档，如果不设置该参数，或使用默认值false，则删除所有匹配条件的文档。
* writeConcern：（可选）抛出异常的级别。

【注】若想删除某个集合中所有的数据，不要使用db.collection.remove({})，因为如果集合中数据量比较大的话，这个操作比较耗时，而是应该直接删除集合db.collection.drop()。

## 查询文档：

### 1、find()：

MongoDB中查询文档使用find()方法。

find()方法以非结构化的方式来显示所有文档。

MongoDB中查询数据的语法格式如下：

|  |
| --- |
| db.collection.find(query, projection) |

* query：可选，使用查询操作符指定查询条件；
* projection：可选，使用投影操作符指定返回的键。查询时返回文档中所有键值，只需省略该参数即可（默认省略）。

如果你需要以易读的方式来读取数据，可以使用pretty()方法，语法格式如下：

|  |
| --- |
| >db.col.find().pretty() |

pretty()方法以格式化的方式来显示所有文档。

### 2、MongoDB与RDBMS Where语句比较

|  |  |  |  |
| --- | --- | --- | --- |
| 操作 | 格式 | 范例 | RDBMS中的类似语句 |
| 等于 | {<key>:<value>} | db.col.find({  “by”:”菜鸟教程”  }  ) | where by = ‘菜鸟教程’ |
| 小于 | {<key>:{$lt:<value>}} | db.col.find({  “likes”:{$lt:50}  }  ) | where likes < 50 |
| 小于或等于 | {<key>:{$lte:<value>}} | db.col.find({  “likes”:{$lte:50}  }  ) | where likes <= 50 |
| 大于 | {<key>:{$gt:<value>}} | db.col.find({  “likes”:{$gt:50}  }  ) | where likes > 50 |
| 大于或等于 | {<key>:{$gte:<value>}} | db.col.find({  “likes”:{$gte:50}  }  ) | where likes >= 50 |
| 不等于 | {<key>:{$ne:<value>}} | db.col.find({  “likes”:{$ne:50}  }  ) | where likes != 50 |

### MongoDB AND条件：

MongoDB的find()方法可以传入多个key，每个key以逗号隔开，即常规SQL的AND条件。

语法格式如下：

|  |
| --- |
| >db.col.find({key1:value1, key2:value2}).pretty() |

### MongoDB OR条件：

MongoDB OR条件语句使用了关键字$or，语法格式如下：

|  |
| --- |
| >db.col.find(  {  $or: [  {key1: value1}, {key2:value2}  ]  }  ).pretty() |

查询province是hubei或者age小于20的文档

|  |
| --- |
| > db.students.find()  { "\_id" : ObjectId("5ee38588d238551cfbd0c305"), "age" : 28, "name" : "zhangsan", "province" : "hubei" }  { "\_id" : ObjectId("5ee385a4d238551cfbd0c306"), "name" : "lisi", "age" : 19 }  { "\_id" : ObjectId("5ee385ddd238551cfbd0c307"), "name" : "wangwu", "age" : 20, "province" : "hubei", "city" : "ezhou" }  > db.students.find({$or:[{province:"hubei"},{age:{$lt:20}}]});  { "\_id" : ObjectId("5ee38588d238551cfbd0c305"), "age" : 28, "name" : "zhangsan", "province" : "hubei" }  { "\_id" : ObjectId("5ee385a4d238551cfbd0c306"), "name" : "lisi", "age" : 19 }  { "\_id" : ObjectId("5ee385ddd238551cfbd0c307"), "name" : "wangwu", "age" : 20, "province" : "hubei", "city" : "ezhou" } |

### AND和OR联合使用：

where province=hubei AND (name=zhangsan OR age >= 19)

|  |
| --- |
| > db.students.find()  { "\_id" : ObjectId("5ee38588d238551cfbd0c305"), "age" : 28, "name" : "zhangsan", "province" : "hubei" }  { "\_id" : ObjectId("5ee385a4d238551cfbd0c306"), "name" : "lisi", "age" : 19 }  { "\_id" : ObjectId("5ee385ddd238551cfbd0c307"), "name" : "wangwu", "age" : 20, "province" : "hubei", "city" : "ezhou" }  > db.students.find({province:"hubei",$or:[{name:"zhangsan"},{age:{$gte:19}}]});  { "\_id" : ObjectId("5ee38588d238551cfbd0c305"), "age" : 28, "name" : "zhangsan", "province" : "hubei" }  { "\_id" : ObjectId("5ee385ddd238551cfbd0c307"), "name" : "wangwu", "age" : 20, "province" : "hubei", "city" : "ezhou" } |