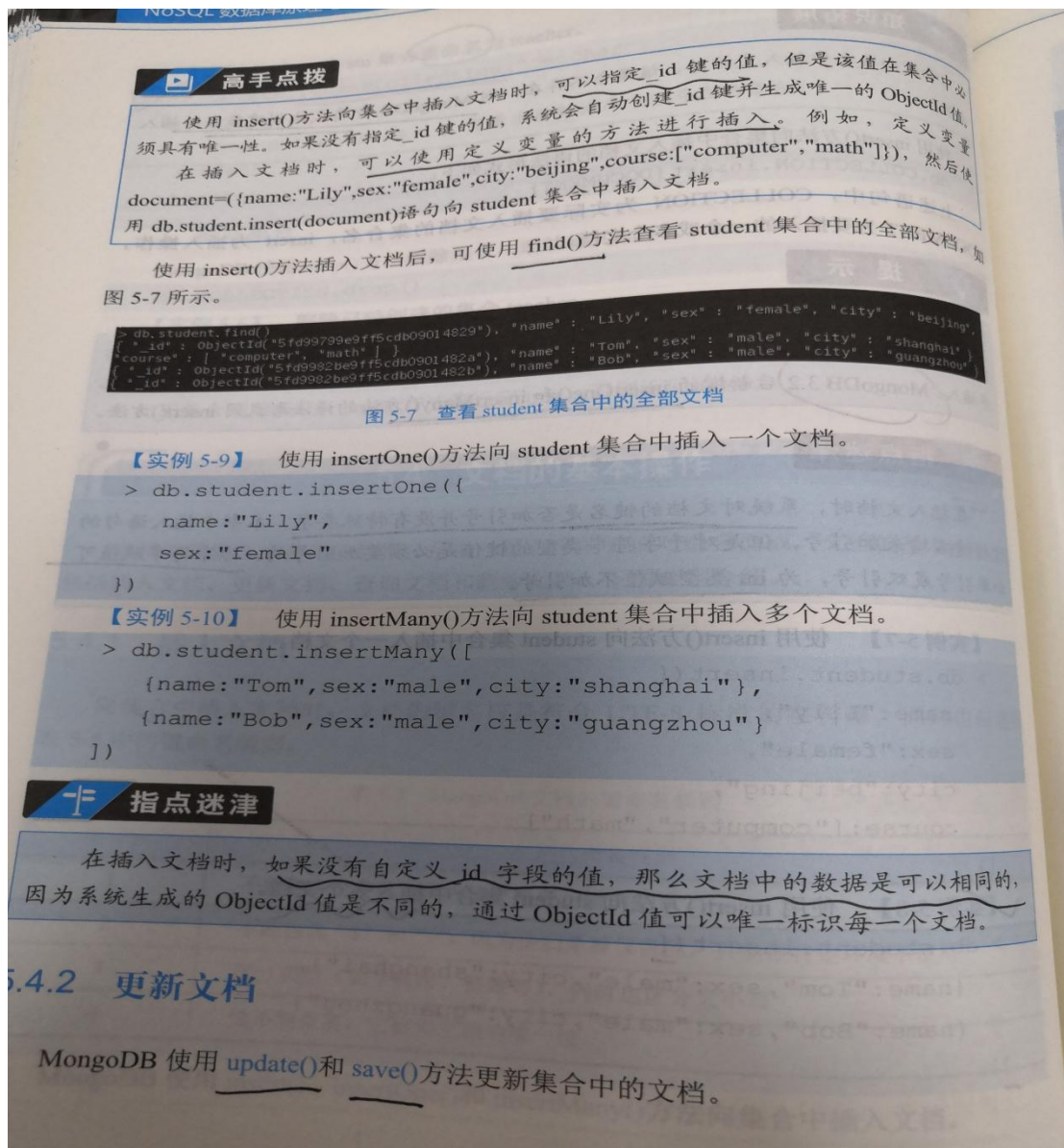


试 讲 内 容

招 聘 岗 位		计算机应用教师岗 1
试讲	具体内容	更新文档
	具体章节	5.4 文档的基本操作
	参考书目	NoSQL 数据库原理与应用案例教程
	作 者	谭秦红, 章立
	出 版 社	航空工业出版社



1. 使用 update()方法更新文档

使用 update()方法更新集合中已存在文档的语法形式如下：

```
db.COLLECTION.update(
  <query>,
  <update>,
  {
    upsert: <boolean>,
    multi: <boolean>,
    writeConcern: <document>
  })
```

上述语句中，COLLECTION 为实际要更新文档的集合名，update 为更新操作，<query>为 update 的查询条件（相当于 SQL 中 UPDATE 的 WHERE 条件），<update>为更新操作符，常见的操作符是“\$set”。upsert 表示如果不存在可以更新的记录，是否插入一个新值，true 为插入，false（默认）为不插入，此项为可选项。multi 表示是否将按条件查询出来的多个记录全部更新，true 为全部更新，false（默认）为只更新查询到的第一个记录，此项也为可选项。writeConcern 指抛出异常的级别。

【实例 5-11】 将 student 集合中 name 为 Lily 的文档的 city 信息更新为“tianjin”。

```
> db.student.update(
  {name:"Lily"},
  {$set:{city:"tianjin"}}
)
```

更新后的结果如图 5-8 所示。

```
> db.student.find()
{ "_id" : ObjectId("5fd99799e9ff5cdb09014829"), "name" : "Lily", "sex" : "female", "city" : "tianjin",
  "course" : [ "computer", "math" ] }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482a"), "name" : "Tom", "sex" : "male", "city" : "shanghai" }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482b"), "name" : "Bob", "sex" : "male", "city" : "guangzhou" }
```

图 5-8 student 集合中更新后的文档



提示

需要注意的是，如果有多个满足条件的记录，那么在默认情况下，只会更新第一个满足条件的记录。

【实例 5-12】 将 student 集合中 name 为 Tom 的文档的 age 信息更新为 20。

```
> db.student.update(
```



```
{name:"Tom"},
{$set:{age:20}},true
)
```

上述语句中设置了 `upsert` 的值为 `true`，因为文档中不存在满足条件的记录，因此将“age: 20”插入文档中，执行结果如图 5-9 所示。

```
db.student.find()
{ "_id" : ObjectId("5fd99799e9ff5cdb09014829"), "name" : "Lily", "sex" : "female", "city" : "tianjin", "course" :
  "computer", "math" : 1 }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482a"), "name" : "Tom", "sex" : "male", "city" : "shanghai", "age" : 20 }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482b"), "name" : "Bob", "sex" : "male", "city" : "guangzhou" }
```

图 5-9 student 集合中更新后的文档

【实例 5-13】 将 student 集合中 sex 为 male 的文档的 city 信息更新为“beijing”。

```
> db.student.update(
  {sex:"male"},
  {$set:{city:"beijing"}},true,false
)
```

上述语句中设置了 `multi` 的值为 `false`，所以尽管满足条件的记录有两个，但是最终只更新满足条件的第一个记录，执行结果如图 5-10 所示。

```
> db.student.find()
{ "_id" : ObjectId("5fd99799e9ff5cdb09014829"), "name" : "Lily", "sex" : "female", "city" : "tianjin", "course" :
  "computer", "math" : 1 }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482a"), "name" : "Tom", "sex" : "male", "city" : "beijing", "age" : 20 }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482b"), "name" : "Bob", "sex" : "male", "city" : "guangzhou" }
```

图 5-10 student 集合中的文档

上述语句中，如果将 `multi` 设置为 `true`，则会更新所有满足条件的记录，执行语句及结果如图 5-11 所示。

```
> db.student.update({sex:'male'},{$set:{city:'beijing'}},true,true)
WriteResult({ "nMatched" : 2, "nUpserted" : 0, "nModified" : 2 })
> db.student.find()
{ "_id" : ObjectId("5fd99799e9ff5cdb09014829"), "name" : "Lily", "sex" : "female", "city" : "tianjin", "course" :
  "computer", "math" : 1 }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482a"), "name" : "Tom", "sex" : "male", "city" : "beijing", "age" : 20 }
{ "_id" : ObjectId("5fd9982be9ff5cdb0901482b"), "name" : "Bob", "sex" : "male", "city" : "beijing" }
```

图 5-11 student 集合中的文档

指点迷津

再次更新时，对于已经符合更新要求的记录，则不会再进行重复更新。

2. 使用 save() 方法更新文档

使用 save() 方法更新集合中已存在文档的语法形式如下：

```
db.COLLECTION.save(object)
```

上述语句中，object 表示要更新的对象。

【实例 5-14】 更新_id 为 ObjectId("5fd9982be9ff5cdb0901482a") 的文档。

```
> db.student.save({
  _id:ObjectId("5fd9982be9ff5cdb0901482a"),
  name:"Tom",
  sex:"male",
  city:"shanghai"
})
```

在使用 save() 方法更新文档时，如果集合中没有与更新语句中 id 值相同的文档，系统会直接向集合中插入文档。

5.4.3 查询文档

MongoDB 使用 find()、pretty() 和 findOne() 方法查询集合中的文档，语法形式如下：

```
db.COLLECTION.find(query,projection)
```

```
db.COLLECTION.find().pretty()
```

上述语句中，query 和 projection 都为可选项。query 为使用查询操作符指定查询条件，projection 为使用投影操作符指定返回的键，默认返回所有键。使用 pretty() 方法查询文档的结果如图 5-12 所示。

```
> db.student.find().pretty()

  '_id' : ObjectId('5fd99799e9ff5cdb09014829'),
  'name' : 'Lily',
  'sex' : 'female',
  'city' : 'tianjin',
  'course' : {
    'computer',
    'math'
  }

  '_id' : ObjectId('5fd9982be9ff5cdb0901482a'),
  'name' : 'Tom',
  'sex' : 'male',
  'city' : 'shanghai'

  '_id' : ObjectId('5fd9982be9ff5cdb0901482b'),
  'name' : 'Bob',
  'sex' : 'male',
  'city' : 'beijing'
```

图 5-12 使用 pretty() 方法查询文档