MongoDB 数据模型

李焕贞

河北师范大学软件学院

本章大纲

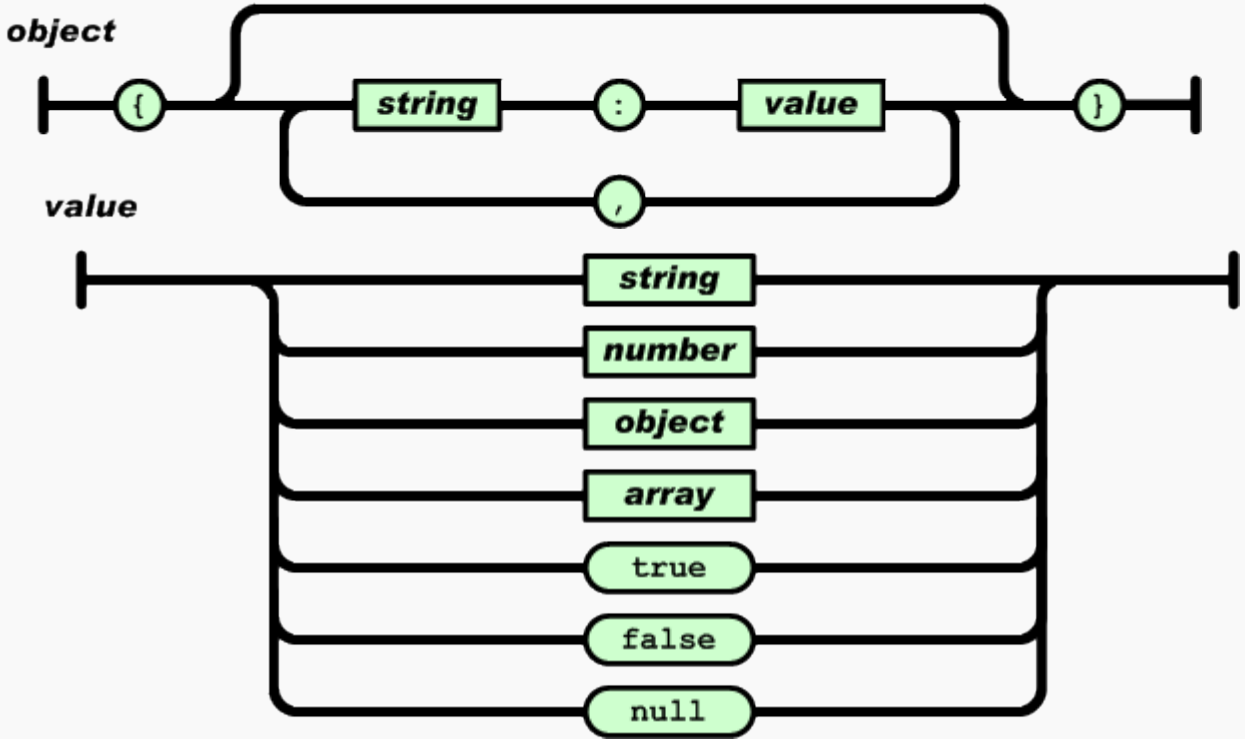
- MongoDB 数据类型
- MongoDB Shell 使用

MongoDB 数据类型

- 数据类型简介
- 基本数据类型
- Date
- Timestamp
- ObjectId
- 内嵌文档
- 数组

MongoDB 数据类型-简介

BSON可以理解为在JSON基础上添加了一些新的数据类型，包括Date，正则表达式，对数值类型的更进一步划分等。



MongoDB 数据类型-简介

数据类型	类型编号	数据类型	类型编号
Double	1	Regular Expression	11
String	2	JavaScript	13
Object	3	Symbol	14
Array	4	JavaScript(Scope)	15
Binary data	5	32-bit integer	16
Object id	7	Timestamp	17
Boolean	8	64-bit integer	18
Date	9	Min Key	255
Null	10	Max Key	127

```
db.collection.find({name:{$type:2}})
```

MongoDB 数据类型-基本数据类型

- null　　表示空值或不存在的字段　例如：

db.collection.find({"y":null})

- 布尔　　有两个值true或false　　例如：{"y":true}
- 数值类型　支持32-int、64-int以及64-double

注：JavaScript只支持64位浮点数

例如：{"y":10}　-　double

{"y":NumberInt(10)} - 32

{"y":NumberLong(10)} - 64

- 字符串使用UTF-8对字符串进行编码例如：{"y":"Hello　MongoDB"}
- 二进制数据可以保存由任意字节组成的字符串，例如：图片、视频等

MongoDB 数据类型-基本数据类型

- 正则表达式：主要用于查询，使用正则表达式作为限定条件

  例如：

{name:/foo/} name字段含有foo的文档

{name:/foo/i} name字段含有foo的文档，且不区分大小写

{name:/^foo/i} name字段以foo开头，且不区分大小写

- JavaScript代码：文档中可以包含任意的JavaScript代码

  例如：{“func”:function functionname( ){}}

MongoDB 数据类型-Date日期

- MongoDB中，日期类型是一个64位的整数，它代表的是距Unix epoch的毫秒数
- MongoDB在存储时间时，先转化为UTC时间

  北京时间(CST) = UTC + 8个小时
- MongoDBShell中可以使用new Date或ISODate来创建时间对象，在进行显示时，Shell会根据本地时间去设置显示日期对象

MongoDB 数据类型-Timestamp

- 时间戳类型有两部分组成：

- Timestamp只供MongoDB数据库服务内部使用，用于记录操作的详细时间
- Timestamp类型和Date类型是没有关系的，对于我们来说使用更多的Date类型
- 相关函数：new Timestamp()

MongoDB 数据类型-ObjectId

- ObjectId由24个十六进制字符构成，每个字节存储两位十六进制数字，总共需要12字节存储空间
- 例如：{"_id":ObjectId("5a7866e75640374fb2cd5623")}
- 每个字节代表的含义如下:

相关函数：

- ObjectId()　　用于创建ObjectID
- getTimestamp()　用于取得ObjectID的时间戳
- valueOf()　　用于取得ObjectID的字符串表示

MongoDB 数据类型-内嵌文档

文档可以作为键的值，这样的文档称为内嵌文档。内嵌文档可以使数据不用保存成扁平结构的键值对，从而使数据组织方式更加自然。

例如：下面是一个与博客管理有关的文档

```
{
    _id: <ObjectIdl>,
    title: MongoDBDateModeln,
    author: foo,
    comments：[
    {who:"John", comment:"Good"},
    {who:"Joe",comment:"ExceUent"}]
}
```

MongoDB 数据类型-内嵌文档

blog collection

```
{
  _id:<ObjectId1>,
  title:"Hello MongoDB",
  author:"jike"
}
```

comment collection

```
{
   _id: <ObjectId2>,
   blog:<ObjectId1>,
   who: "John",
   comment:"Good"
}
```

Reference Data Models

VS

```
{
  _id:<ObjectId1>,
  title:"Hello MongoDB",
   comments:[
       { who:"John",
         comment:"Good"},
       { who:"Joe",
         comment:"Excellent" }]
}
```

Embedded Data Models

MongoDB 数据类型-数组

- 数组是使用[]来表示的一组值，它既可以作为有序对象（列表、栈、队列），也能作为无序对象（如集合）来操作
- 数组中可以包含不同数据类型的元素（字符串、浮点数、文档等）

  例如：[3.14，"hello"，[12,3]，{"key":"MongoDB"}]
- 针对数组MongoDB提供了许多特定的操作符，例如：$push，$pop， $pull，$slice，$addToSet等
- MongoDB可自动的为数组元素建立Multikey索引

实例

//切换数据库

use test01

//插入数据，整型，数组，子文档

db.class1.insert({_id:1,name:"tom",age:NumberInt(20),scores:[70,80,90],address:{"provice":"hebei","city":"nanjing"}})

```
> use test01
switched to db test01
> db.class1.insert({_id:1,name:"tom",age:NumberInt(20),scores:[70,80,90],address
:{"provice":"hebei","city":"nanjing"}})
WriteResult({ "nInserted" : 1 })
>
```

本章大纲

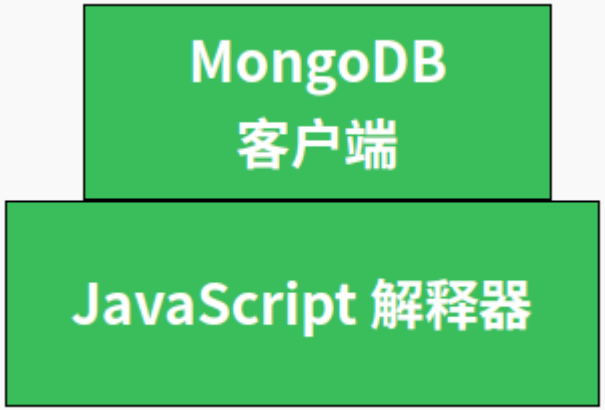
- MongoDB 数据类型
- MongoDB Shell 使用

MongoDB Shell 使用-简介

MongoDB Shell是 MongoDB自带的JavaScirpt Shell，随MongoDB一同发布，它是MonoDB客户端工具，可以在Shell中使用命令与MongoDB实例交互，对数据库的管理操作（CURD、集群配置、状态查看等）都可以通过MongoDB Shell来完成。

MongoDB Shell

= JavaScirpt解释器 + MongoDB客户端



MongoDB Shell 使用-基本功能

1、执行JavaScript命令

2、MongoDB客户端一基本命令

- 连接/切换数据库 -- use dbname;
- 数据插入 -- db.stu.insert(obj);
- 数据查询 -- db.stu.find(query);
- 数据更新 -- db.stu.update(query,obj);
- 数据删除 -- db.stu.remove(query);

MongoDB Shell 使用-基本功能

1、执行JavaScript命令

2、MongoDB客户端一基本命令

- 连接/切换数据库 -- use dbname;
- 数据插入 -- db.stu.insert(obj);
- 数据查询 -- db.stu.find(query);
- 数据更新 -- db.stu.update(query,obj);
- 数据删除 -- db.stu.remove(query);

MongoDB Shell 使用-使用技巧

1、help查看帮助

2、执行脚本

- 直接运行　例如：mongo[--quiet]script.js
- 交互式运行 例如：load("script.js")

3、.mongorc.js 文件（位于用户主目录）

4、更多命令可参考

https://docs.mongodb.com/manual/reference/mongo-shell/

实例

1、创建/切换数据库

```
use db1
```

2、查看数据库

```
show dbs
```

3、删除当前数据库

```
db.dropDatabase()
```

4、创建集合

```
db.createCollection("c1")
```

5、创建集合并添加数据

```
db.dept.insert({deptno:1,deptname:"技术部",location:"beijing"})
```

实例

6、查看集合

```
show collections
```

7、删除集合

```
db.collection_name.drop()
```

8、查看所有文档数据

```
db.dept.find()
```

9、查看单独的一个文档

```
db.dept.findOne()
```

10、删除指定文档

```
db.dept.remove({deptno:1})
```

11、更新文档

```
db.dept.update({deptno:2},{$set:{location:"shenzhen"}})"}})
```