Mongodb教程

1. **介绍**

MongoDB 是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。

在高负载的情况下，添加更多的节点，可以保证服务器性能。

MongoDB 旨在为WEB应用提供可扩展的高性能数据存储解决方案。

MongoDB 将数据存储为一个文档，数据结构由键值(key=>value)对组成。MongoDB 文档类似于 JSON 对象。字段值可以包含其他文档，数组及文档数组。



**特点：**

MongoDB 是一个面向文档存储的数据库，操作起来比较简单和容易。

你可以在MongoDB记录中设置任何属性的索引 (如：FirstName="Sameer",Address="8 Gandhi Road")来实现更快的排序。

你可以通过本地或者网络创建数据镜像，这使得MongoDB有更强的扩展性。

如果负载的增加（需要更多的存储空间和更强的处理能力） ，它可以分布在计算机网络中的其他节点上这就是所谓的分片。

Mongo支持丰富的查询表达式。查询指令使用JSON形式的标记，可轻易查询文档中内嵌的对象及数组。

MongoDb 使用update()命令可以实现替换完成的文档（数据）或者一些指定的数据字段 。

Mongodb中的Map/reduce主要是用来对数据进行批量处理和聚合操作。

Map和Reduce。Map函数调用emit(key,value)遍历集合中所有的记录，将key与value传给Reduce函数进行处理。

Map函数和Reduce函数是使用Javascript编写的，并可以通过db.runCommand或mapreduce命令来执行MapReduce操作。

GridFS是MongoDB中的一个内置功能，可以用于存放大量小文件。

MongoDB允许在服务端执行脚本，可以用Javascript编写某个函数，直接在服务端执行，也可以把函数的定义存储在服务端，下次直接调用即可。

MongoDB支持各种编程语言:RUBY，PYTHON，JAVA，C++，PHP，C#等多种语言。

MongoDB安装简单。

**相关概念：**



**数据库**：一个mongodb中可以建立多个数据库。MongoDB的默认数据库为"db"，该数据库存储在data目录中。MongoDB的单个实例可以容纳多个独立的数据库，每一个都有自己的集合和权限，不同的数据库也放置在不同的文件中。

数据库命名规则：不能是空字符串（"")、不得含有' '（空格)、.、$、/、\和\0 (空字符)、应全部小写、最多64字节

有一些数据库名是保留的，可以直接访问这些有特殊作用的数据库。

admin： 从权限的角度来看，这是"root"数据库。要是将一个用户添加到这个数据库，这个用户自动继承所有数据库的权限。一些特定的服务器端命令也只能从这个数据库运行，比如列出所有的数据库或者关闭服务器。

local: 这个数据永远不会被复制，可以用来存储限于本地单台服务器的任意集合

config: 当Mongo用于分片设置时，config数据库在内部使用，用于保存分片的相关信息

**文档：**是MongoDB中数据的基本单元，非常类似于关系数据库中的数据行（但比数据行复杂很多），文档是一组键值(key-value)对(即BSON)。MongoDB 的文档不需要设置相同的字段，并且相同的字段不需要相同的数据类型，这与关系型数据库有很大的区别，也是 MongoDB 非常突出的特点。

**文档特点：**

（1）文档中的键/值对是有序的。





以上两个文档是完全不同的，就是因为文档是有序的。但是对我们程序开发的影响不大

（2）文档中的值不仅可以是在双引号里面的字符串，还可以是其他几种数据类型（甚至可以是整个嵌入的文档)。

（3）MongoDB区分类型和大小写。

（4）MongoDB的文档不能有重复的键。

（5）文档的键是字符串。除了少数例外情况，键可以使用任意UTF-8字符。

**文档命名规则：**

键不能含有\0 (空字符)。这个字符用来表示键的结尾。

.和$有特别的意义，只有在特定环境下才能使用。

以下划线"\_"开头的键是保留的(不是严格要求的)。

**集合：**集合就是 MongoDB 文档组，类似于关系数据库中的表，集合存在于数据库中，集合没有固定的结构，这意味着你在对集合可以插入不同格式和类型的数据，但通常情况下我们插入集合的数据都会有一定的关联性。

当第一个文档插入时，集合就会被创建。集合名默认为库名

**集合命名规则：**

（1）集合名不 能是空字符串""。

（2）集合名不能含有\0字符（空字符)，这个字符表示集合名的结尾。

（3）集合名不能以"system."开头，这是为系统集合保留的前缀。

（4）用户创建的集合名字不能含有保留字符。有些驱动程序的确支持在集合名里面包含，这是因为某些系统生成的集合中包含该字符。除非你要访问这种系统创建的集合，否则千万不要在名字里出现$。

集合是无模式的，这意味着集合里面的文档可以是各式各样的，例如下面的两个文档都可以存入一个集合中。



但是我们一般都不会将不同类型的文档放入同一个集合，这样避免文档混乱、影响查询效率

1. **下载安装**

Linux下载地址：

https://www.mongodb.com/download-center#community

Linux安装步骤：

<https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/>

其他系统版本在此网站也有相关安装步骤，可以参考

配置文件：/etc/mongodb.conf

bin文件：/usr/bin

启动：./mongod 或者 service mongod start

启动的时候还可以指定ip: ./mongod –bind\_ip 0.0.0.0(此处可以指定IP为0.0.0.0，默认不指定的话是本地IP，127.0.0.1)

Mongodb在启动的时候，默认使用数据目录/data/db，如果数据目录不存在或者没有写权限，则会报错

重启：service mongod restart

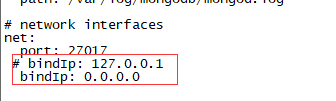
停止：service mongod stop

1. **远程连接配置**

我们在Linux上部署启动mongodb后，还需要配置远程连接，这样其他应用或者远程客户端才能进行连接操作。

Ubuntu系统：

步骤1：修改mongodb的配置文件，vim /etc/mongod.conf，找到如下地方，注释掉127.0.0.1行，新增bindIp:0.0.0.0，这样mongodb就和本地解绑，可以远程调用



步骤2：要么关闭防火墙，要么防火墙放开端口号27017，mongodb默认的 端口号是27017

Ubuntu防火墙放开端口

ufw status 查看防火墙状态

ufw enable 启动防火墙

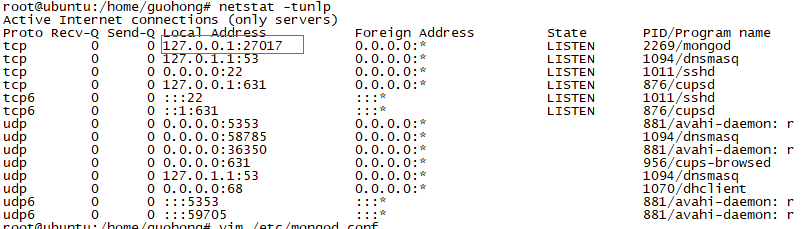
ufw allow 27017 放开27017端口

ufw disable 停止防火墙

**问题：**

此处我在配置的时候遇到一个问题，就是在进行上面步骤后，还是无法远程连接MongoDB，telnet端口号的时候不通。

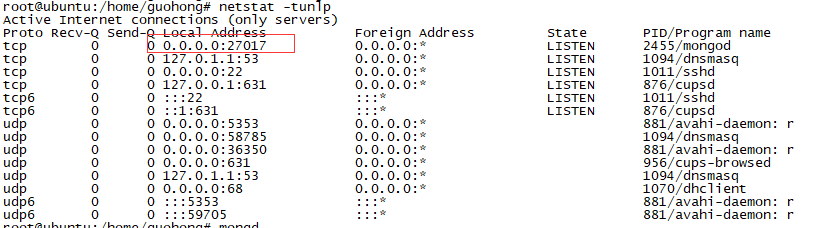
此时使用命令：netstat –tunlp



如上，发现MongoDB启动后还是绑定在127.0.0.1IP上，因此无法进行远程连接。这里我已经修改了配置文件，明显是配置文件没生效或者是MongoDB启动的时候压根儿没读取该配置文件。目前还没找到具体原因。所以采用了临时解决方案，在启动MongoDB的时候指定IP

./mongod –bind\_ip 0.0.0.0

启动后我们再netstat –tunlp



发现此时IP已经变了，这时再进行远程连接就可以了。

我们暂时使用的客户端连接软件是：database master

1. **常用命令**

show dbs:查看数据库列表

db:查看当前数据库名称

show collections:查看集合列表

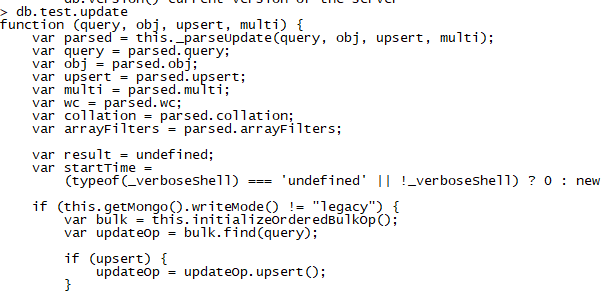
use test:连接到test数据库



help :查看帮助信息

db.help():查看数据库级别的命令帮助文档

db.test.update:查看update函数的源码



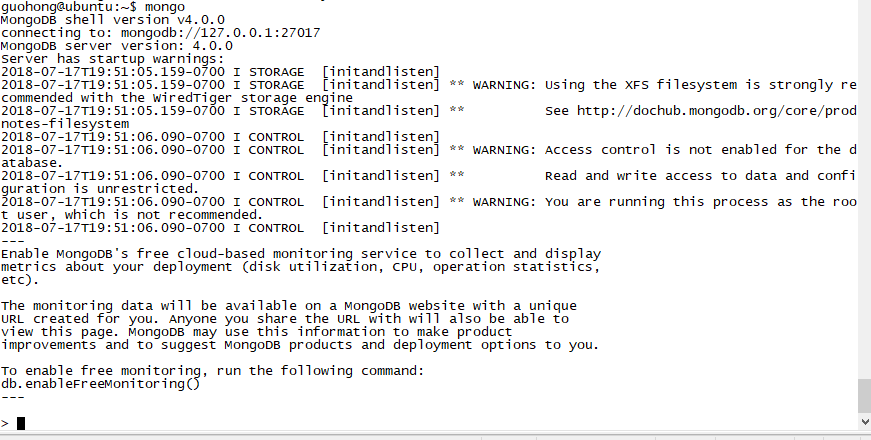
db.version();查看MongoDB的版本号

db.getCollection("test")：得到test集合，如果有属性和集合重名，这种方式可以避免出错，db.test这种方式可能会出错

1. **MongoDB自带的客户端shell**

启动MongoDB客户端：找到MongoDB安装目录的bin文件夹，输入如下命令就可以连接本地的MongoDB服务器，然后进行相关操作。该连接默认连接的是MongoDB的test数据库

./mongo



该客户端shell是功能完备的JavaScript解释器，可以运行任何JavaScript程序。

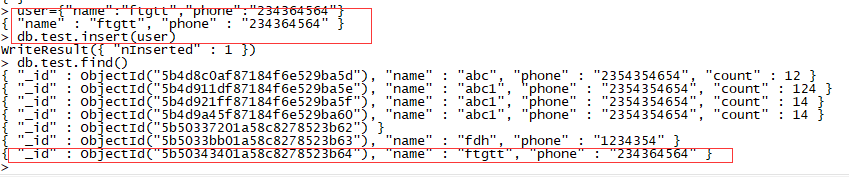
**插入数据：**

方式1：



其中test是集合名称

方式2：



**查询数据：**

查询集合test中的所有数据,使用find时，shell自动显示最多20个匹配的文档，但可以获取更多文档

db.test.find()

查询集合test中的第一条数据

db.test.findOne()

**更新数据：**

1. **数据类型**

**基本数据类型：**

MongoDB的文档类似于json，json只包含6种数据类型（null、布尔、数字、字符串、数组、对象）。MongoDB在保留json基本的键值对特性的基础上，添加了其他的一些数据类型，在不同的编程语言下这些类型的表示有些许差异。

null:用于表示空值或不存在的字段

布尔：有两个值true和false

32位整数：此类型在shell中不可用，JavaScript仅支持64位浮点数，所以32位整数会被自动转换。

64位整数：shell也不支持这个类型

64位浮点数：shell中的数字都是这种类型

字符串：这是用于存储数据的最常用的数据类型。MongoDB中的字符串必须为UTF-8

符号：

对象ID：

日期：日期类型存储的是从标准纪元开始的毫秒数，不存储时区

正则表达式：文档中可以包含正则表达式

代码：文档中还可以包含JavaScript代码

二进制数据：二进制数据可以由任意字节的串组成，不过shell中无法使用

最大值：BSON包含一个特殊类型，表示可能的最大值，shell中没有这个类型

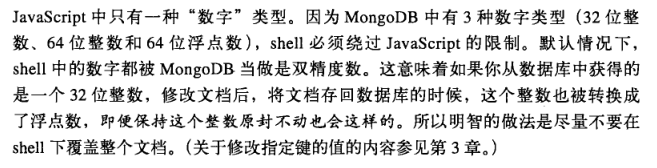
最小值：BSON包含一个特殊类型，表示可能的最小值，shell中没有这个类型

未定义类型：文档中也可以使用未定义类型，比如undefined

数组：值的集合或列表可以表示为数组

内嵌文档：文档可以包含别的文档，也可以作为值嵌入到父文档中。

**数字：**

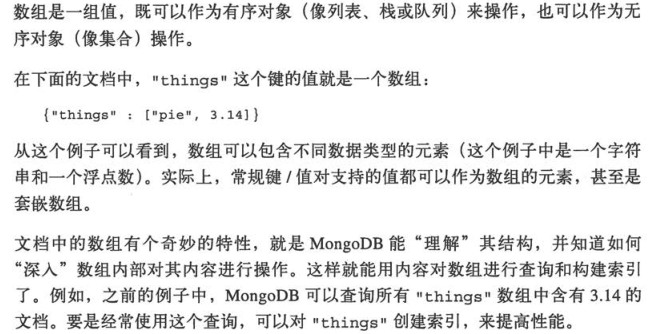


**日期：**

在JavaScript中，Date对象用做MongoDB的日期类型，创建一个新的Date对象时，通常会调用new Date()而不只是Date（）。调用Date（）构造函数不使用new，实际上会返回日期的字符串表示，而不是真正的date对象。这不是MongoDB的特性，这是JavaScript本身的特性，要是不小心忘了而使用Date构造函数，最后会导致日期和字符串混淆，字符串和日期不能相互匹配，所以会给上次、更新、查询等很多操作带来问题。

Shell中的日期显示时使用本地时区设置，但是，日期在数据中是以从标准纪元开始的毫秒数的形式存储的，没有与之相关的时区信息（当然可以把时区信息作为其他键的值存储）

**数组：**



**内嵌文档：**

内嵌文档就是把整个MongoDB文档作为另一个文档中键的一个值，这样数据可以组织得更自然些，不用非得存成扁平结构的。

例如：用一个文档来表示一个人，同时还要保存他的地址，可以将地址内嵌到人的文档中。

同数组一样，MongoDB能够理解内嵌文档的结构，并能深入其中的构建索引、执行查询、或者更新

这样做也有坏处，因为MongoDB会储存更多重复的数据，这样是反规范化的，如果在关系数据库中，只需要一个地方维护地址，更新的时候，用到该地址的其他关联用户的地址信息都会更新，但在MongoDB中，则需要在每个人的文档中更新。

**ObjectId：**

ObjectId 类似唯一主键，可以很快的去生成和排序，包含12bytes,含义是：

前4个字节表示创建unix时间戳,格林尼治时间 UTC时间，比北京时间晚了8个小时

接下来的3个字节是机器标识码

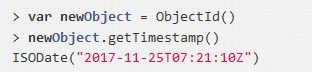
紧接的两个字节由进程id组成（PID

最后三个字节是随机数



MongoDB中存储的文档必须有一个\_id键。这个键的值可以是任何类型的，默认是个ObjectId对象

由于ObjectId中保存了创建的时间戳，所以你不需要为你的文档保存时间戳字段，你可以通过 getTimestamp 函数来获取文档的创建时间:



1. **aa**