## 什么是MongoDB ?

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。



## 主要特点

* 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。

## 历史

* 2007年10月，MongoDB由10gen团队所发展。2009年2月首度推出。
* 2012年05月23日，MongoDB2.1 开发分支发布了! 该版本采用全新架构，包含诸多增强。
* 2012年06月06日，MongoDB 2.0.6 发布，分布式文档数据库。
* 2013年04月23日，MongoDB 2.4.3 发布，此版本包括了一些性能优化，功能增强以及bug修复。
* 2013年08月20日，MongoDB 2.4.6 发布。
* 2013年11月01日，MongoDB 2.4.8 发布。
* ……

## 语言支持

MongoDB有官方的驱动如下：

* [C](http://github.com/mongodb/mongo-c-driver" \t "https://www.runoob.com/mongodb/_blank)
* [C++](http://github.com/mongodb/mongo" \t "https://www.runoob.com/mongodb/_blank)
* [C# / .NET](http://www.mongodb.org/display/DOCS/CSharp+Language+Center" \t "https://www.runoob.com/mongodb/_blank)
* [Erlang](https://github.com/TonyGen/mongodb-erlang" \t "https://www.runoob.com/mongodb/_blank)
* [Haskell](http://hackage.haskell.org/package/mongoDB" \t "https://www.runoob.com/mongodb/_blank)
* [Java](http://github.com/mongodb/mongo-java-driver)
* [JavaScript](http://www.mongodb.org/display/DOCS/Javascript+Language+Center" \t "https://www.runoob.com/mongodb/_blank)
* [Lisp](https://github.com/fons/cl-mongo" \t "https://www.runoob.com/mongodb/_blank)
* [node.JS](http://github.com/mongodb/node-mongodb-native" \t "https://www.runoob.com/mongodb/_blank)
* [Perl](http://github.com/mongodb/mongo-perl-driver" \t "https://www.runoob.com/mongodb/_blank)
* [PHP](http://github.com/mongodb/mongo-php-driver" \t "https://www.runoob.com/mongodb/_blank)
* [Python](http://github.com/mongodb/mongo-python-driver" \t "https://www.runoob.com/mongodb/_blank)
* [Ruby](http://github.com/mongodb/mongo-ruby-driver" \t "https://www.runoob.com/mongodb/_blank)
* [Scala](https://github.com/mongodb/casbah" \t "https://www.runoob.com/mongodb/_blank)

## MongoDB 应用案例

下面列举一些公司MongoDB的实际应用：

* Craiglist上使用MongoDB的存档数十亿条记录。
* FourSquare，基于位置的社交网站，在Amazon EC2的服务器上使用MongoDB分享数据。
* Shutterfly，以互联网为基础的社会和个人出版服务，使用MongoDB的各种持久性数据存储的要求。
* bit.ly, 一个基于Web的网址缩短服务，使用MongoDB的存储自己的数据。
* spike.com，一个MTV网络的联营公司， spike.com使用MongoDB的。
* Intuit公司，一个为小企业和个人的软件和服务提供商，为小型企业使用MongoDB的跟踪用户的数据。
* sourceforge.net，资源网站查找，创建和发布开源软件免费，使用MongoDB的后端存储。
* etsy.com ，一个购买和出售手工制作物品网站，使用MongoDB。
* 纽约时报，领先的在线新闻门户网站之一，使用MongoDB。
* CERN，著名的粒子物理研究所，欧洲核子研究中心大型强子对撞机的数据使用MongoDB。

## MongoDB 工具

有几种可用于MongoDB的管理工具。

### **监控**

MongoDB提供了网络和系统监控工具Munin，它作为一个插件应用于MongoDB中。

Gangila是MongoDB高性能的系统监视的工具，它作为一个插件应用于MongoDB中。

基于图形界面的开源工具 Cacti, 用于查看CPU负载, 网络带宽利用率,它也提供了一个应用于监控 MongoDB 的插件。

### **GUI**

* Fang of Mongo – 网页式,由Django和jQuery所构成。
* Futon4Mongo – 一个CouchDB Futon web的mongodb山寨版。
* Mongo3 – Ruby写成。
* MongoHub – 适用于OSX的应用程序。
* Opricot – 一个基于浏览器的MongoDB控制台, 由PHP撰写而成。
* Database Master — Windows的mongodb管理工具
* RockMongo — 最好的PHP语言的MongoDB管理工具，轻量级, 支持多国语言.

# Linux平台安装MongoDB

MongoDB 提供了 linux 各个发行版本 64 位的安装包，你可以在官网下载安装包。

安装前我们需要安装各个 Linux 平台依赖包。

sudo yum install libcurl openssl

下载[https://www.mongodb.com/download-center#community](https://www.mongodb.com/download-center" \l "community" \t "https://www.runoob.com/mongodb/_blank)

上传解压

tar -zxvf mongodb-linux-x86\_64-ubuntu1604-4.2.8.tgz # 解压

mv mongodb-src-r4.2.8 /usr/local/mongodb4

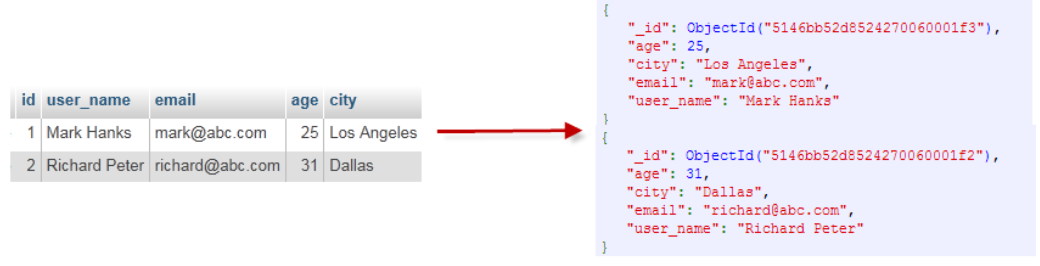
MongoDB 的可执行文件位于 bin 目录下，所以可以将其添加到 PATH 路径中：

export PATH=/usr/local/mongodb4/bin:$PATH

# MongoDB 概念解析



通过下图实例，我们也可以更直观的了解Mongo中的一些概念：



## 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。

"show dbs" 命令可以显示所有数据的列表。

$ ./mongoMongoDB shell version: 3.0.6

connecting to: test> show dbslocal 0.078GB

test 0.078GB>

执行 "db" 命令可以显示当前数据库对象或集合。

$ ./mongoMongoDB shell version: 3.0.6

connecting to: test> db

test>

运行"use"命令，可以连接到一个指定的数据库。

> use local

switched to db local> dblocal>

以上实例命令中，"local" 是你要链接的数据库。

在下一个章节我们将详细讲解MongoDB中命令的使用。

数据库也通过名字来标识。数据库名可以是满足以下条件的任意UTF-8字符串。

* 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(Document)

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。

一个简单的文档例子如下：

{"site":"www.runoob.com", "name":"菜鸟教程"}

下表列出了 RDBMS 与 MongoDB 对应的术语：

|  |  |
| --- | --- |
| **RDBMS** | **MongoDB** |
| 数据库 | 数据库 |
| 表格 | 集合 |
| 行 | 文档 |
| 列 | 字段 |
| 表联合 | 嵌入文档 |
| 主键 | 主键 (MongoDB 提供了 key 为 \_id ) |
| **数据库服务和客户端** | |
| Mysqld/Oracle | mongod |
| mysql/sqlplus | mongo |

需要注意的是：

1. 文档中的键/值对是有序的。
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。

文档键命名规范：

* 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 RDBMS （关系数据库管理系统：Relational Database Management System)中的表格。

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。

比如，我们可以将以下不同数据结构的文档插入到集合中：

{"site":"www.baidu.com"}{"site":"www.google.com","name":"Google"}{"site":"www.runoob.com","name":"菜鸟教程","num":5}

当第一个文档插入时，集合就会被创建。

### **合法的集合名**

* 集合名不能是空字符串""。
* 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。
* 集合名不能以"system."开头，这是为系统集合保留的前缀。
* 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$。

如下实例：

db.col.findOne()

### **capped collections**

Capped collections 就是固定大小的collection。

它有很高的性能以及队列过期的特性(过期按照插入的顺序). 有点和 "RRD" 概念类似。

Capped collections 是高性能自动的维护对象的插入顺序。它非常适合类似记录日志的功能和标准的 collection 不同，你必须要显式的创建一个capped collection，指定一个 collection 的大小，单位是字节。collection 的数据存储空间值提前分配的。

Capped collections 可以按照文档的插入顺序保存到集合中，而且这些文档在磁盘上存放位置也是按照插入顺序来保存的，所以当我们更新Capped collections 中文档的时候，更新后的文档不可以超过之前文档的大小，这样话就可以确保所有文档在磁盘上的位置一直保持不变。

由于 Capped collection 是按照文档的插入顺序而不是使用索引确定插入位置，这样的话可以提高增添数据的效率。MongoDB 的操作日志文件 oplog.rs 就是利用 Capped Collection 来实现的。

要注意的是指定的存储大小包含了数据库的头信息。

db.createCollection("mycoll", {capped:true, size:100000})

* 在 capped collection 中，你能添加新的对象。
* 能进行更新，然而，对象不会增加存储空间。如果增加，更新就会失败 。
* 使用 Capped Collection 不能删除一个文档，可以使用 drop() 方法删除 collection 所有的行。
* 删除之后，你必须显式的重新创建这个 collection。
* 在32bit机器中，capped collection 最大存储为 1e9( 1X109)个字节。

## 元数据

数据库的信息是存储在集合中。它们使用了系统的命名空间：

dbname.system.\*

在MongoDB数据库中名字空间 <dbname>.system.\* 是包含多种系统信息的特殊集合(Collection)，如下:



对于修改系统集合中的对象有如下限制。

在{{system.indexes}}插入数据，可以创建索引。但除此之外该表信息是不可变的(特殊的drop index命令将自动更新相关信息)。

{{system.users}}是可修改的。 {{system.profile}}是可删除的。

## MongoDB 数据类型

下表为MongoDB中常用的几种数据类型。



下面说明下几种重要的数据类型。

### **ObjectId**

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

> var newObject = ObjectId()

> newObject.getTimestamp()

ISODate("2017-11-25T07:21:10Z")

ObjectId 转为字符串

> newObject.str

5a1919e63df83ce79df8b38f

### **字符串**

BSON 字符串都是 UTF-8 编码。

### **时间戳**

BSON 有一个特殊的时间戳类型用于 MongoDB 内部使用，与普通的 日期 类型不相关。 时间戳值是一个 64 位的值。其中：

* 前32位是一个 time\_t 值（与Unix新纪元相差的秒数）
* 后32位是在某秒中操作的一个递增的序数

在单个 mongod 实例中，时间戳值通常是唯一的。

在复制集中， oplog 有一个 ts 字段。这个字段中的值使用BSON时间戳表示了操作时间。

BSON 时间戳类型主要用于 MongoDB 内部使用。在大多数情况下的应用开发中，你可以使用 BSON 日期类型。

### **日期**

表示当前距离 Unix新纪元（1970年1月1日）的毫秒数。日期类型是有符号的, 负数表示 1970 年之前的日期。

> var mydate1 = new Date() //格林尼治时间

> mydate1

ISODate("2018-03-04T14:58:51.233Z")

> typeof mydate1

object

> var mydate2 = ISODate() //格林尼治时间

> mydate2

ISODate("2018-03-04T15:00:45.479Z")

> typeof mydate2

object

这样创建的时间是日期类型，可以使用 JS 中的 Date 类型的方法。

返回一个时间类型的字符串：

> var mydate1str = mydate1.toString()

> mydate1str

Sun Mar 04 2018 14:58:51 GMT+0000 (UTC)

> typeof mydate1str

string

或者

> Date()

Sun Mar 04 2018 15:02:59 GMT+0000 (UTC)