# MongoDB数据介绍、下载、及安装

## 介绍

MongoDB是一个基于分布式文件存储的数据库。由C++语言编写。旨在为WEB应用提供可扩展的高性能数据存储解决方案。 MongoDB是一个高性能，开源，无模式的文档型数据库，是当前NoSql数据库中比较热门的一种。它在许多场景下可用于替代传统的关系型数据库或键/值存储方式。是一个介于[关系数据库](http://baike.baidu.com/view/68348.htm)和非关系数据库之间的产品，是非关系数据库当中功能最丰富，**最像关系数据库的**。他支持的数据结构非常松散，是类似json的bson格式，因此可以存储比较复杂的数据类型。Mongo最大的特点是他支持的查询语言非常强大，其语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且还支持对数据建立[索引](http://baike.baidu.com/view/262241.htm" \t "_blank)。MongoDB介于缓存与数据库之间，**存取速度逊于缓存但远远高于传统数据库**。**MongoDB加入了对LINQ的支持。**

**特点:**

　　高性能、易部署、易使用，存储数据非常方便。主要功能特性有：

* 面向集合存储，易存储对象类型的数据。
* 模式自由。
* 支持动态查询。
* 支持完全索引，包含内部对象。
* 支持查询。
* 支持复制和故障恢复。
* 使用高效的二进制数据存储，包括大型对象（如视频等）。
* 自动处理碎片，以支持云计算层次的扩展性
* 支持Python，PHP，Ruby，**Java**，C，**C#**，**Javascript**，Perl及C++语言的驱动程序，社区中也提供了对Erlang及**.NET**等平台的驱动程序。
* 文件存储格式为BSON（一种JSON的扩展）。
* 可通过网络访问。

**功能:**

* **面向集合的存储：**适合存储对象及**JSON**形式的数据。
* **动态查询：**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。
* **完整的索引支持：**包括文档内嵌对象及数组。Mongo的查询优化器会分析查询表达式，并生成一个高效的查询计划。
* **查询监视：**Mongo包含一个监视工具用于分析数据库操作的性能。
* **复制及自动故障转移：**Mongo数据库支持服务器之间的数据复制，支持主-从模式及服务器之间的相互复制。复制的主要目标是提供冗余及自动故障转移。
* **高效的传统存储方式：**支持二进制数据及大型对象（如照片或图片）
* **自动分片以支持云级别的伸缩性：**自动分片功能支持水平的数据库集群，可动态添加额外的机器。

**适用场合:**

* 网站数据：Mongo非常适合实时的插入，更新与查询，并具备网站实时数据存储所需的复制及高度伸缩性。
* 缓存：由于性能很高，Mongo也适合作为信息基础设施的缓存层。在系统重启之后，由Mongo搭建的持久化缓存层可以避免下层的数据源 过载。
* 大尺寸，低价值的数据：使用传统的关系型数据库存储一些数据时可能会比较昂贵，在此之前，很多时候程序员往往会选择传统的文件进行存储。
* 高伸缩性的场景：Mongo非常适合由数十或数百台服务器组成的数据库。Mongo的路线图中已经包含对MapReduce引擎的内置支持。
* 用于对象及JSON数据的存储：Mongo的BSON数据格式非常适合文档化格式的存储及查询。

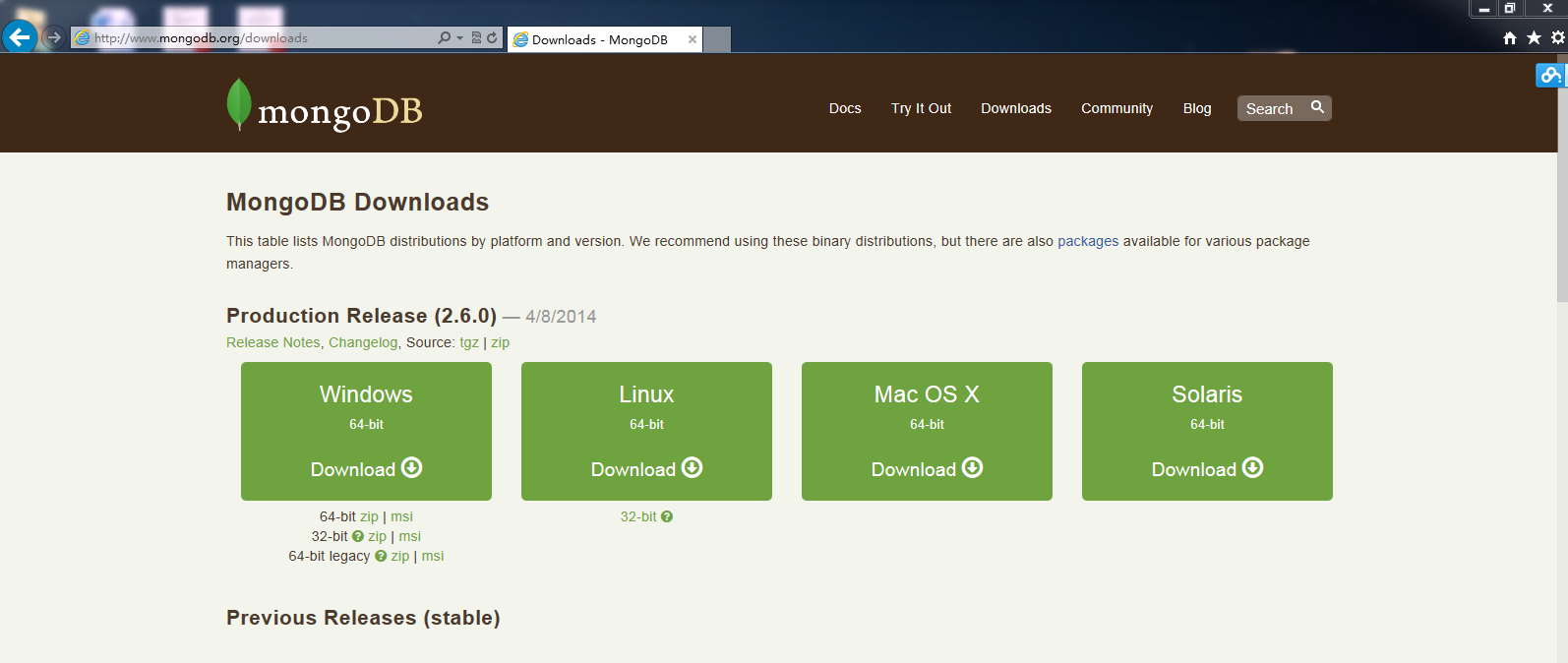
## 二、 下载

MongoDB的官网是：<http://www.mongodb.org/>

MongoDB最新版本下载在官网的DownLoad菜单下,选择合适的版本下载（例选择windows64位,生产环境最好使用64位版）：<http://www.mongodb.org/downloads>

不同版本：（1）64位，支持所有比Windows Xp高的64位系统，包括Windows7和Windows Server2008R2等

（2）32位，只用于测试和开发者使用，最大数据库容量<2G



## 三、 安装

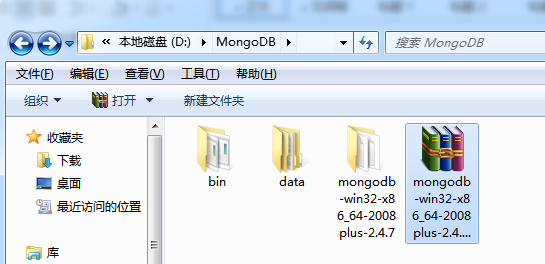
1. 解压mongodb-win32-x86\_64-2008plus-2.6.0（目前最新版本2014/4/8） ，创建路径D:\ \MongoDB(可根据自己要求创建) ，将解压后的Bin文件Copy to 此文件夹下
2. 在D:\ MongoDB(可根据自己要求创建)下建立data文件夹 D: \MongoDB\data ,然后分别建立db,log两个文件夹，至此mongodb下有以下文件夹

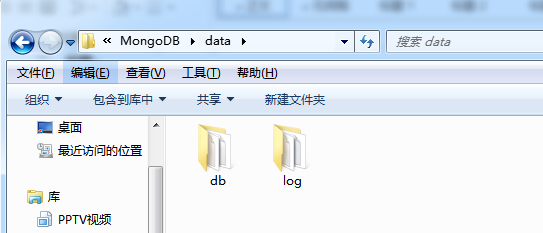
D:\ MongoDB\bin

D:\ MongoDB\db

D:\ MongoDB\data\log

在log文件夹下创建一个日志文件MongoDB.log，即C:\MongoDB\data\log\MongoDB.log





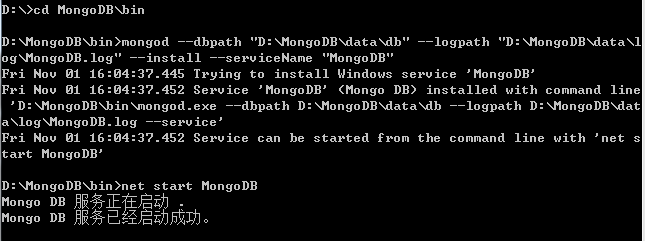
1. 以Windows service 方式安装MongoDB服务器

运行cmd.exe

> cd D:\ MongoDB\bin

> D:\ MongoDB\bin>mongod --dbpath "D:\ MongoDB\data\db" --logpath "D:\ MongoDB\data\log\MongoDB.log" --install --serviceName "MongoDB" --serviceDisplayName "MongoDB"

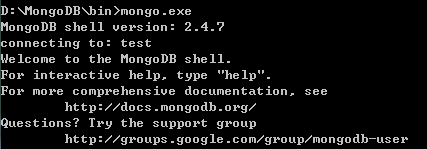
这里MongoDB.log就是开始建立的日志文件，--serviceName "MongoDB" 服务名为MongoDB



右键计算机——管理——展开服务和应用程序——单击服务——右侧可以看到



4、运行bin文件夹下mongo.exe 客户端测试



测试数据库操作

>help  (查看相关信息）

>db.foo.insert({a:1})    （往foo表插入a,1字段值,foo表为默认表)

>db.foo.find()                (查看foo表数据）



# MongoDB For C#

**using** **System**;

**using** **System.Collections.Generic**;

**using** **System.Linq**;

**using** **System.Text**;

**using** **MongoDB.Bson**;

**using** **MongoDB.Driver**;

**using** **MongoDB.Driver.Builders**;

**namespace** **ConsoleApplication1**

{

**public** **class** **Entity**

{

**public** ObjectId Id { **get**; **set**; }

**public** string Name { **get**; **set**; }

}

**class** **Program**

{

**static** **void** Main(string[] args)

{

var connectionString = "mongodb://localhost";

var client = **new** MongoClient(connectionString);

var server = client.GetServer();

var database = server.GetDatabase("test");

var collection = database.GetCollection<Entity>("entities");

var entity = **new** Entity { Name = "Tom" };

collection.Insert(entity);

var id = entity.Id;

var query = Query<Entity>.EQ(e => e.Id, id);

entity = collection.FindOne(query);

entity.Name = "Dick";

collection.Save(entity);

var update = Update<Entity>.Set(e => e.Name, "Harry");

collection.Update(query, update);

collection.Remove(query);

}

}

}

# 3．MongoDB高级特性

* 1. **Capped Collection**

Capped Collection是性能出色的有着固定大小的集合，在创建时要预先指定

大小，如空间用完，则新添加的对象将会取代集合中最旧的对象。

示例代码：

db.createCollection(“mycoll”,{capped:true,size:1024,max:100})

将普通的Collection装换为Capped Collection:

db.runCommand({“convertToCapped”:”mycoll”,sieze:1024,max:100})

注：size:为固定大小，max:存放的对大文档树

* 1. **GridFS 实现海量存储**
  2. **MapReduce编程模型**

MapDB的MapReduce相当于Sqlserver的“group by”，在MongoDB

中使用MapReduce执行并行数据统计很容易。

1. Map进行分组

示例代码：

M=function() {emit(this.classid, 1)}

注：Map函数调用emit（key，value）返回键值对，对表按classid进行分组

1. Reduce聚合计算

示例代码：

R=function（key，values）{

var x=0；

values.forEach（function（v）{x+=v}）;

return x;

}

注：Reduce函数接收的参数类似Group效果，将Map返回的键值序列组合成{key，[value1,value2,value3,….]}形式传给reduce

1. Result获取结果

示例代码：

Res=db.runCommand({

mapreduce:”students”,----------------操作的目标集合

map:M,

reduce:R,

out:”students\_res”---计算结果存储在students\_res集合

finalize:f, -----格式化输出

query:{age:{$lt:20}} ---过滤结果集

});

注：finalize（）可以对reduce的结果进行输出样式的格式化处理，示例代码：

f=function (key,value) {return { classid:key, count:value }; }

# 4. 数据管理

**4.1 导出工具mongoexport**

Mongoexport工具可以把一个collection导出成json格式或csv格式，也可以根据给定的条件导出数据。

mongoexport -h 222.195.148.178:27017 -d my\_mongodb –c student -f id,name,ge –q ‘{age:$lt>30}’ -0 E:\students.dat

注：h：服务器主机ip

u：数据库用户名

p：数据库密码

d：数据库名字

c：集合collection的名字

f：指出要导出哪些列

q：指出数据过滤条件

o：指出导出文件名

**4.2 导入工具mongoimport**

Mongoexport可以将json或者csv格式的数据导入到某个collection中。

mongoinport -h 222.195.148.178:27017 –d my\_mongodb –c students –type csv –headerline –file students\_csv.dat

注：h：服务器主机ip

u：数据库用户名

p：数据库密码

d：数据库名字

c：导入集合collection的名字

f：指出要导入哪些列

o：指出导入文件名

4.3 数据的备份和恢复

（1）备份工具mongodump

Mongodump –d my\_mongodb –o my\_mongodb\_dump

注：-d：源数据库 -o：备份数据库目录

（2）恢复工具mongorestore

mongorestore –d my\_mongodb my\_mongodb\_dump

第五节 管理命令

5.1 远程克隆collection

db.runCommand({cloneCollection:”test.students”,from:”222.195.148.178:27017”,copyIndexs:false/true})

注：将远程服务器222.195.148.178:27017上的collection复制到本地，可以方便的传输数据，否则需要在远程服务器上将数据导出，然后拷贝到本地

5.2 复制数据库copyDatabase

db.copyDatabase(“fromdb1”,”todb2”,”222.195.148.178:27017”);

注：fromdb1----源数据库名称

todb2 ----目标数据库名称

fromhost --源数据库地址

5.3 本地复制

db.copyDatabase(“fromdb1”,”todb2”,”localhost”);

5.4 刷新磁盘(主要用于备份)

在mongodb中使用“db.runCommand({fsync:1，lock:1})”命令可以将内存尚未写入磁盘的信息写入磁盘，并锁住对数据库更新的操作，单独操作可以继续。

解锁：db.$cmd.sys.unlock.findOne()

查看数据可读写状态：db.currentLOp()

**使用用户名和口令**

MongoDB默认的启动不需要验证用户名和密码，并且对所有的数据库具有操作权限。可以在启动服务的时候设置登录验证模块,指定auth参数：

mongod --auth --dbpath E:\\ -logpath E:\\ --port

注：要使登录验证生效，必须在admin数据库中添加一个用户，同时要指定“auth”参数

**建立系统root用户**

进入admin数据库，通过addUser添加用户，并通过auth设置数据库连接验证

db.addUser(“root”,”111”) -------添加系统用户

db.auth(“root”,”111”) ---------数据库连接验证

连接数据库：

 mongo –u toor –p

**建立特定数据库的用户**

use test 切换到test数据库

向test数据库添加用户

db.addUser(“map”,”111”);

登录test数据库

mongo 127.0.0.1:27017/test –u map –p

# 5. MongoDB副本集和主从复制

我们知道sql server能够做到读写分离，双机热备份和集群部署，当然mongodb也能做到，实际应用中我们不希望数据库采用单点部署，如果碰到数据库宕机或者被毁灭性破坏那是多么的糟糕。

**1. 复制或副本集**

复制集或副本集（Replica Set）就是有自动故障恢复功能的主从集群。通俗的讲就是用多台机器进行同一数据的异步同步，从而使多台机器拥有同一数据的多个副本，并且当主库当掉时在不需要用户干预的情况下自动切换其他备份服务器做主库。而且还可以利用副本服务器做只读服务器，实现读写分离，提高负载。需要指出的是：Mongodb的主从复制模式官方已经不在推荐使用了。

**第一步**：建立集群，给集群起一个名字，名字是为了易于与别的副本集区分，也是为了方便的将整个集合视为一个整体。这里命名为：**i4People**。

**第二步：**

**一、启动多台服务器**

（1）启动节点服务器1（222.195.148.178:27017）。--replSet作用是让服务器至知晓这个“i4People”副本集中还有其他的同伴指向（222.195.148.120:27017）：

mongod --dbpath "D:\MongoDB\data\db" –logpath "D:\MongoDB\data\log\MongoDB.log" --replSet -- autoresync

i4People/222.195.148.120:27017 --install --serviceName "MongoDB27017" --serviceDisplayName "MongoDB27017"

（2） 启动节点服务器2（222.195.148.120:27017）。--replSet作用是让服务器至知晓这个“i4People”副本集中还有其他的同伴指向（222.195.148.178:27017）：

mongod --dbpath "D:\MongoDB\data\db" –logpath "D:\MongoDB\data\log\MongoDB.log" --replSet i4People/222.195.148.178:27017 --install --serviceName "MongoDB27017" --serviceDisplayName "MongoDB27017"

（3） 启动节点服务器3（仲裁:只参与投票，不接受复制的数据）（222.195.158.169:27017）。--replSet作用是让服务器至知晓这个“i4People”副本集中还有其他的同伴指向（222.195.148.178:27017）：

mongod --dbpath "D:\MongoDB\data\db" –logpath "D:\MongoDB\data\log\MongoDB.log" --replSet i4People/222.195.148.178:27017 --install --serviceName "MongoDBArbiter27017" --serviceDisplayName "MongoDBArbiter27017"

**二、在shell中初始化副本集**

在shell中，连接以上三个服务器中的一个，例如：222.195.158.169:27017。初始化命令只能执行一次：

Mongo 222.195.158.169:27017/admin

>db.runCommand({“replSetInitiate”:{

…”\_id”:”i4People”,

…”members”:[

…{“\_id”:1,”host”:”222.195.148.178:27017”，”priority”:1} //priority设置优先权

…{“\_id”:2,”host”:”222.195.148.120:27017”}

…{“\_id”:3,”host:”222.195.158.169:27017”,”arbiterOnly”:trye}

]

利用db.isMaster()查看哪一台被选为活跃节点

复制集的节点：1.standard：常规节点，存储为完整的数据备份，参与选举投票，有可能成为活跃节点。

2.passive：存储了完整的数据副本，参与投票，不能成为活跃节点

3.arbiter：仲裁节点，只参与投票，不接受复制的数据，也不能成为活跃节点

**管理维护副本集：**

**（1）读写分离**

假如222.195.148.178:27017为主数据库，可以向主数据库插入数据，然后让后从222.195.148.120：27017从数据库中查询数据，实现读写分离，分离主数据库的大量查询请求。具体步骤：

>mongo 222.195.148.120:27017

SECONDARY>db.getMongo().setSlaveOk();

SECONDARY>db.table1.find();

（2） **增减节点**

一、 配置并启动新节点

mongod --dbpath "D:\MongoDB\data\db" –logpath

"D:\MongoDB\data\log\MongoDB.log" –port 27016 --replSet

i4People/222.195.148.178:27017,2.195.148.120：27017

--install –serviceName “MongoDBArbiter27016" --serviceDisplayName "MongoDB27016"

二、添加新节点到复制集

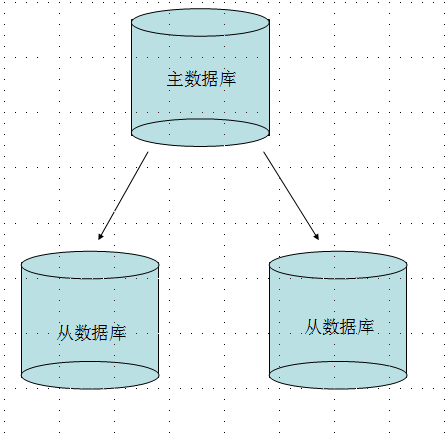
PRIMARY>rs.add(“222.195.148.178:27016”);

三、删除节点

PRIMARY>rs.remove(“222.195.148.178:27016”);

**2. 主从复制（最新版本不推荐，建议使用复制集）**

主从复制是mongodb最常用的复制方式,这种方式很灵活.可用于备份,故障恢复,读扩展等.最基本的设置方式就是建立一个主节点和一个或多个从节点（从节点最好不要超过12个）,每个从节点要知道主节点的地址。示意图：



主从复制的优点是：  <1>  数据备份。

    <2>  数据恢复。

    <3>  读写分离。

**第一步：创建主数据库服务器（ip：222.195.158.169）**

D:\MongoDB\bin>mongod --dbpath=E:\MongoDB\data --logpath=E:\MongoDB\log\master.log –-port 27017 --logappend --master --oplogSize 64 --install --serviceName "MongoDBMaster" --serviceDisplayName "MongoDBMaster"

**第二步：创建从服务器（slave1 ip：222.195.148.178,slave2 ip:222.195.148.120）**

**从数据库服务器Slave1：**

D:\MongoDB\bin>mongod --dbpath=E:\MongoDB\data --logpath=E:\MongoDB\log\slave1.log –-port 27017 --logappend --slave -- source 222.195.158.169:27017 --oplogSize 64 --install --serviceName "MongoDBSlave1" --serviceDisplayName "MongoDBSlave1"

**从数据库服务器 Slave2**

D:\MongoDB\bin>mongod --dbpath=E:\MongoDB\data --logpath=E:\MongoDB\log\slave2.log –-port 27017 --logappend --slave -- source 222.195.158.169:27017 --oplogSize 64 --install --serviceName "MongoDBSlave2" --serviceDisplayName "MongoDBSlave2"

**动态添加从节点（事先不指定源）**

D:\MongoDB\bin>mongod --dbpath=E:\MongoDB\data --logpath=E:\MongoDB\log\slave2.log –-port 27017 --logappend --slave --install --serviceName "MongoDBSlave2" --serviceDisplayName "MongoDBSlave2"

添加：>use local

>db.sources.insert({“host”:”222.195.158.169:27017”})

移除：>db.sourves.remove({“host”:”222.195.158.169:27017”})