文档数据库
-MongoDB

李鸿岐

（lihongqi@nwpu.edu.cn）

2025/4/25

目录 | contents

目录 | contents

为什么要用MongoDB?

- 键值存储非常简单，速度极快且相对容易伸缩。
- 关系型数据库较难伸缩，至少很难水平伸缩，但拥有丰富的数据模型和强大的查询语言。
- 如果能介于两者之间，就能成为一款易伸缩、能存储丰富数据结构、提供复杂查询机制的数据库。

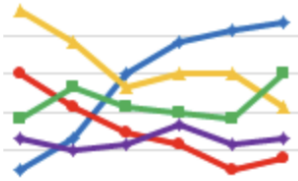
MongoDB是一个**介于关系数据库和非关系数据库之间的产品**，是非关系数据库当中功能最丰富，最像关系数据库的。在使用场景方面，非常适合用做以下应用程序的主要数据存储：Web应用程序、分析与记录应用程序，以及任何要求有中等级别缓存的应用程序。此外，由于它能方便地存储无Schema数据，MongoDB还很适合保存**事先无法知晓其数据结构**的数据。

为什么要用MongoDB?

DB-Engines Ranking

The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.

Read more about the method of calculating the scores.

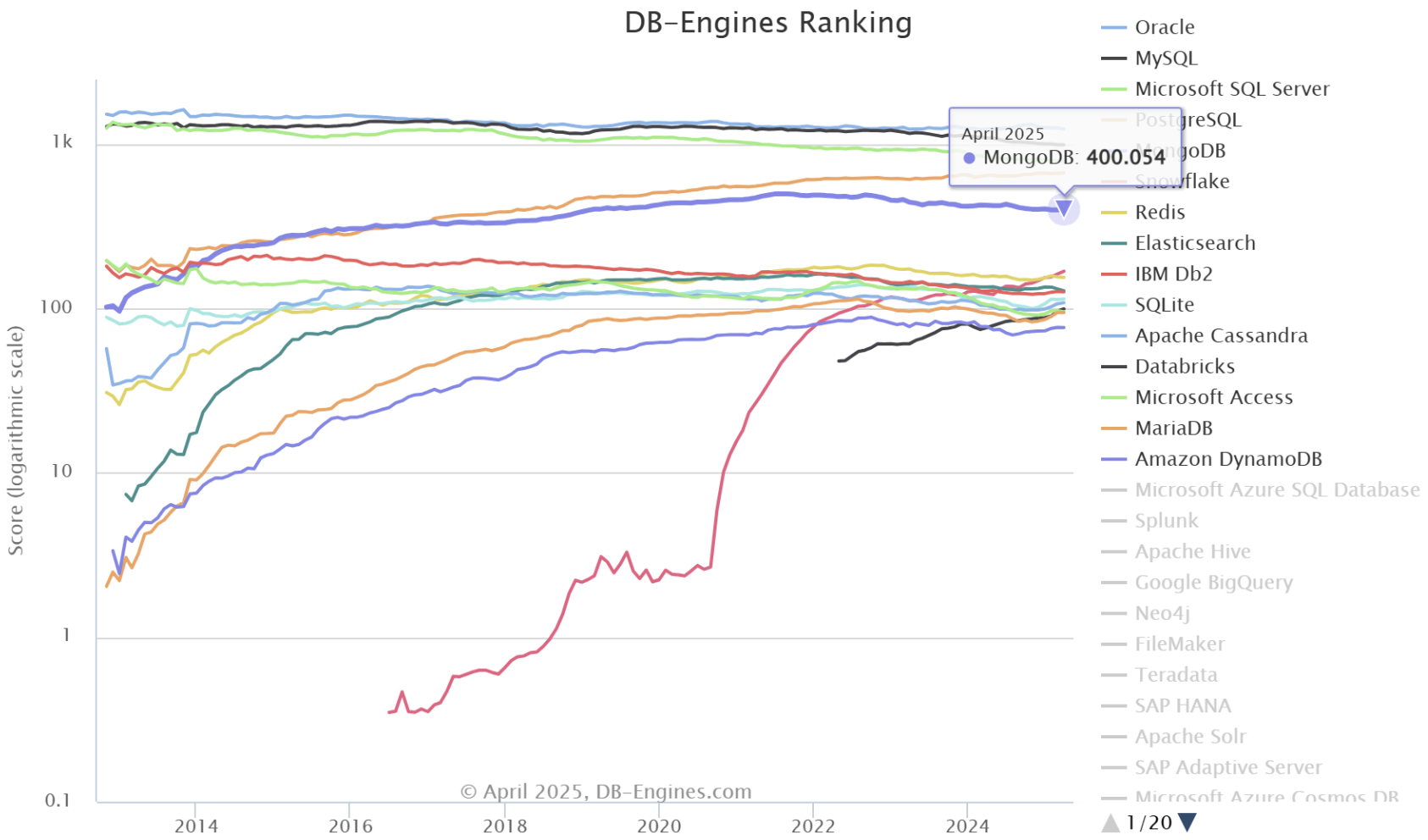


trend chart

424 systems in ranking, April 2025

Rank			DBMS	Database Model	Score		
Apr 2025	Mar 2025	Apr 2024			Apr 2025	Mar 2025	Apr 2024
1.	1.	1.	Oracle	Relational, Multi-model	1231.05	-22.03	-3.21
2.	2.	2.	MySQL	Relational, Multi-model	987.11	-1.02	-100.62
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	785.01	-3.13	-44.79
4.	4.	4.	PostgreSQL	Relational, Multi-model	667.25	+3.82	+22.20
5.	5.	5.	MongoDB	Document, Multi-model	400.05	+3.63	-23.91
6.	6.	↑ 9.	Snowflake	Relational	168.09	+6.31	+44.89
7.	7.	↓ 6.	Redis	Key-value, Multi-model	154.11	-1.25	-2.33
8.	8.	↓ 7.	Elasticsearch	Multi-model	128.08	-3.30	-6.70
9.	9.	↓ 8.	IBM Db2	Relational, Multi-model	126.04	-0.53	-1.45
10.	10.	10.	SQLite	Relational	114.09	+1.01	-1.92

为什么要用MongoDB?



01-MongoDB简介

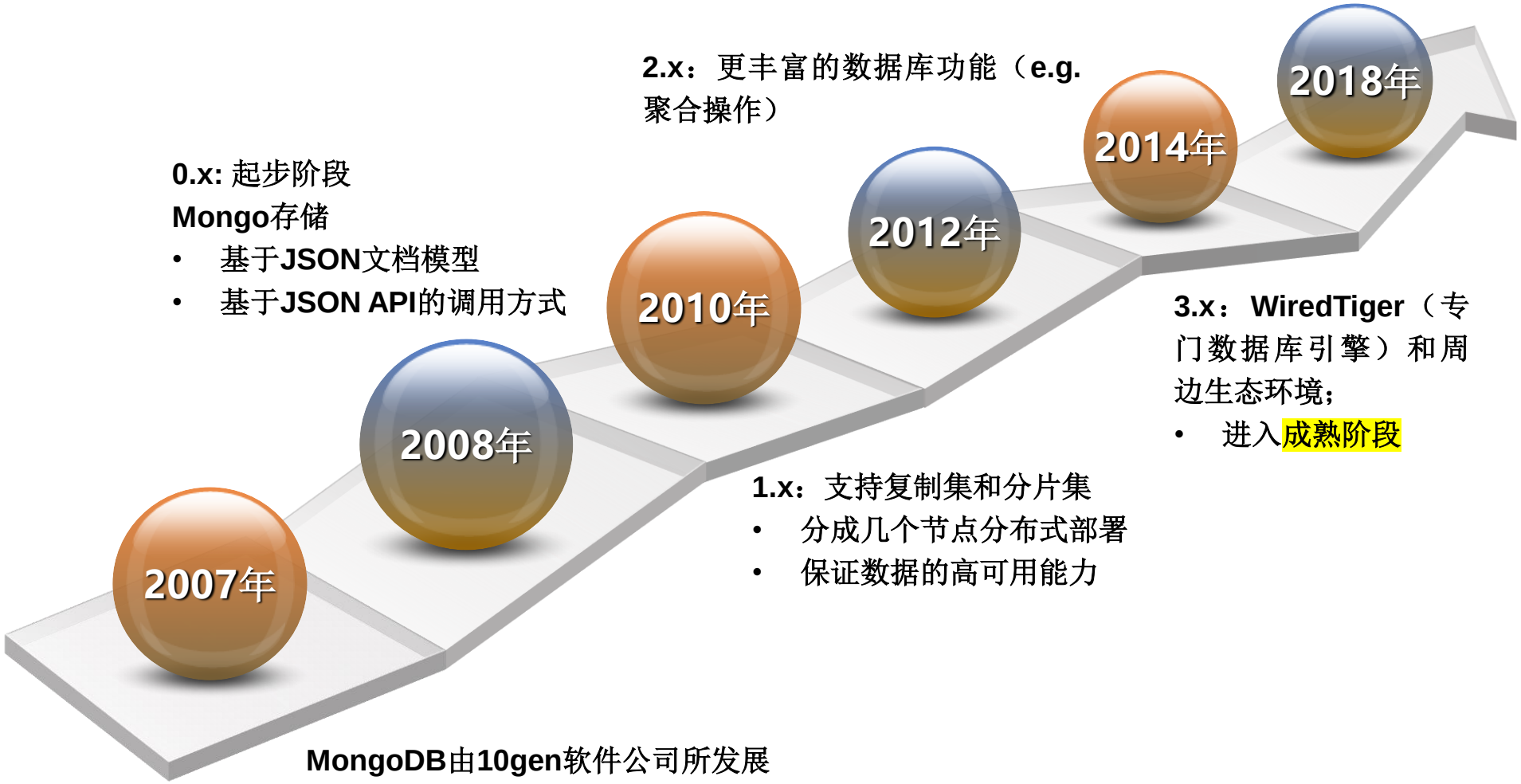
MongoDB：源于“Humongous”

“庞大”，意图是可以处理大规模的数据

MongoDB：一个以JSON为数据模型的**文档数据库**

- 为什么叫文档数据库？
  - 文档来自于“JSON **Document**”
  - JSON（JavaScript Object Notation, JS 对象简谱）
  - 并非我们一般理解的PDF，WORD文档
- 由上市公司MongoDB Inc.开发，总部位于美国纽约
  - 公司提供99%的代码
  - GitHub
  - 社区版是基于SSPL，一种开源协议
  - 企业版是基于商业协议，需付费使用
- MongoDB有两个发布版本：社区版和企业版

MongoDB版本变迁



0.x: 起步阶段

Mongo存储

- 基于JSON文档模型
- 基于JSON API的调用方式

2.x：更丰富的数据库功能（e.g. 聚合操作）

4.x：分布式事务支持

1.x：支持复制集和分片集

- 分成几个节点分布式部署
- 保证数据的高可用能力

3.x：WiredTiger（专门数据库引擎）和周边生态环境；

- 进入成熟阶段

MongoDB由10gen软件公司所发展

01-MongoDB简介

应用范围：

- 应用数据库，类似于Oracle, MySQL, SQL Server、
- 海量数据处理，数据平台

  应用程序、网页应用、手机应用的后端数据库

  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
- 场景：电商、物联网、内容管理、社交、航空、电信….

场景限制：

- 高度事物性的系统：例如银行或会计系统

  需要大量原子性复杂事务的应用程序还是多采取传统的关系型数据库
- 传统的商业智能应用：

  针对特定问题的BI数据库会产生高度优化的查询方式（数据仓库可能是更合适的选择）
- 需要SQL的问题….

01-MongoDB简介

主要用途：

MongoDB主要特点

- 面向集合存储，易存储对象类型的数据。
  - 所谓“面向集合”（**Collection**-Oriented），意思是数据被分组存储在数据集中，被称为一个集合（Collection)。
  - 每个集合在数据库中都有一个唯一的标识名，并且可以包含无限数目的文档。
  - 集合的概念类似关系型数据库（RDBMS）里的表（table），不同的是它不需要定义任何模式（schema)。
- 模式自由。
  - 对于存储在mongodb数据库中的文件，我们不需要知道它的任何结构定义。
  - 如果需要的话，完全可以把不同结构的文件存储在同一个数据库里。

MongoDB主要特点

- 文件存储格式为BSON（Binary JSON，一种JSON的扩展）
  - 存储在集合中的文档，被存储为键-值对的形式。
  - 键用于唯一标识一个文档，为字符串类型，而值则可以是各种复杂的文件类型。
- 支持动态查询。
- 支持完全索引，包含内部对象。
- 支持复制和故障恢复。
- 使用高效的二进制数据存储，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RUBY，PYTHON，JAVA，C++，PHP等多种语言。
- 可通过网络访问

MongoDB vs. 关系型数据库

	MongoDB	RDBMS
数据模型	文档类型（JSON Document）	关系模型
数据库类型	OLTP	OLTP
CRUD操作	MQL/SQL	SQL
高可用	复制集	集群模式
横向扩展能力	通过原生分片完善支持	数据分区或者应用侵入式
索引支持	B-树、全文索引、地理位置索引、多键（Multikey）索引、TTL索引	B树
开发难度	容易	困难
数据容量	没有理论上限	千万、亿
扩展方式	垂直扩展 + 水平扩展	垂直扩展

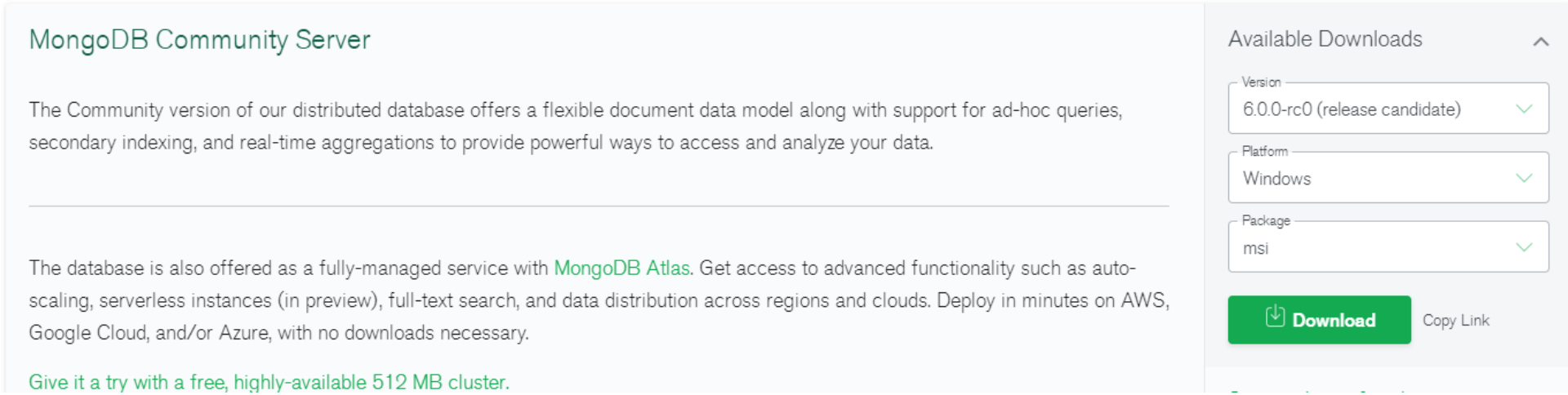
目录 | contents

02-MongoDB下载安装

西北工业大学 NORTHWESTERN POLYTECHNICAL UNIVERSITY

选择正确版本？

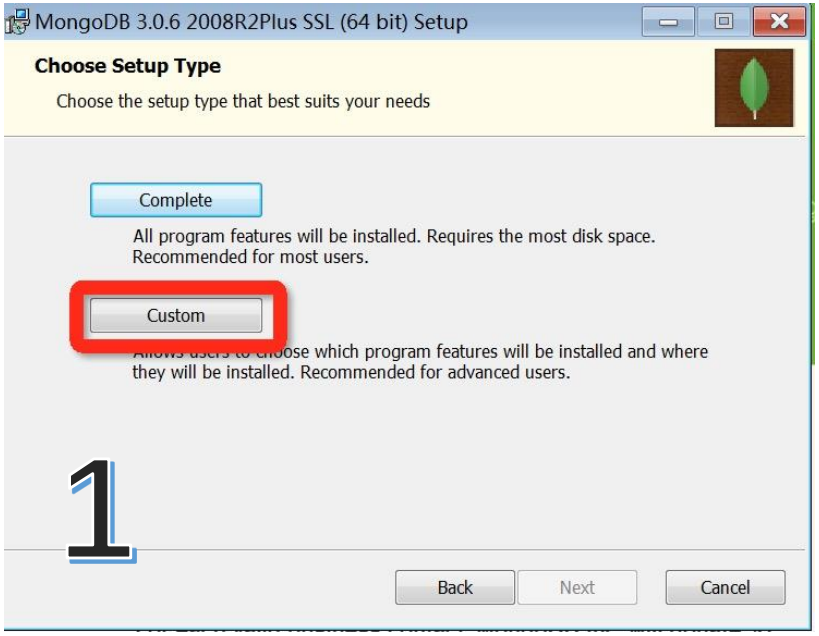
https://www.mongodb.com/download-center/community

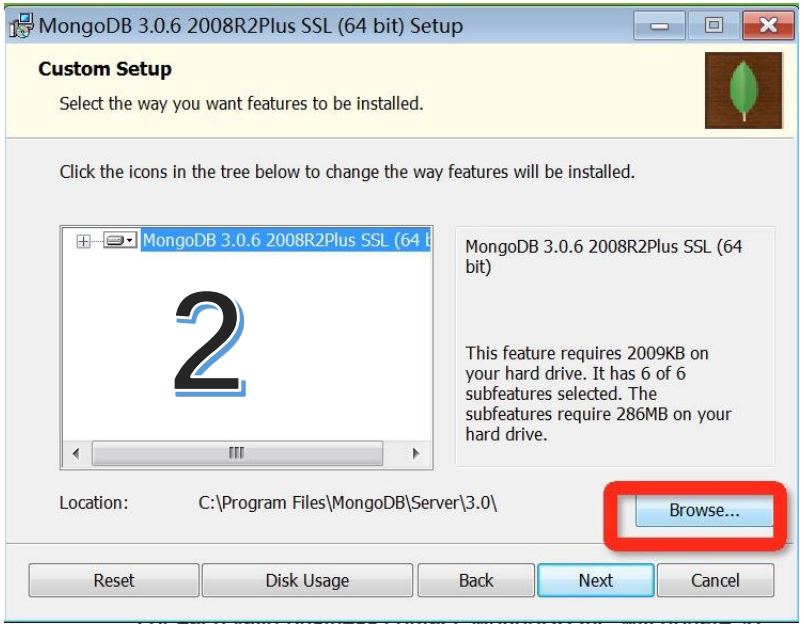


- MongoDB所使用的版本管理相当简单：**偶数号为稳定版，奇数号为开发版。**

注意：32 位产品与 64 位产品之间的区别。32 位和 64 位版本的数据库目前有着相同的功能，唯一的区别是：32 位版本将每个服务器的**数据集<?????>**总大小限制在**2GB**左右；64 位版本没有任何限制，所以在生产环境中应该优先使用 64 位版本。另外，不同的版本之间也有可能发生变化

02-MongoDB下载安装





自定义安装目录时，建议不要选带空格的目录

```
C:\Users\LHQ>D:\Program Files (x86)\MongDB\Install\anzhuang\bin\mongod --dbpath D:\data\db
'D:\Program' 不是内部或外部命令，也不是可运行的程序
或批处理文件。
```

02-MongoDB下载安装

- 创建数据目录

MongoDB 将数据目录存储在 db 目录下。但是这个数据目录不会主动创建，我们在安装完成后需要创建它。请注意，数据目录应该放在根目录下

mongod - The database server.

```
cd D:\
md "\data\db"
```

命令提示符 - D:\mongodb\bin\mongod --dbpath D:\data\db

D:\mongoDB\bin\mongod --dbpath d:\data\db

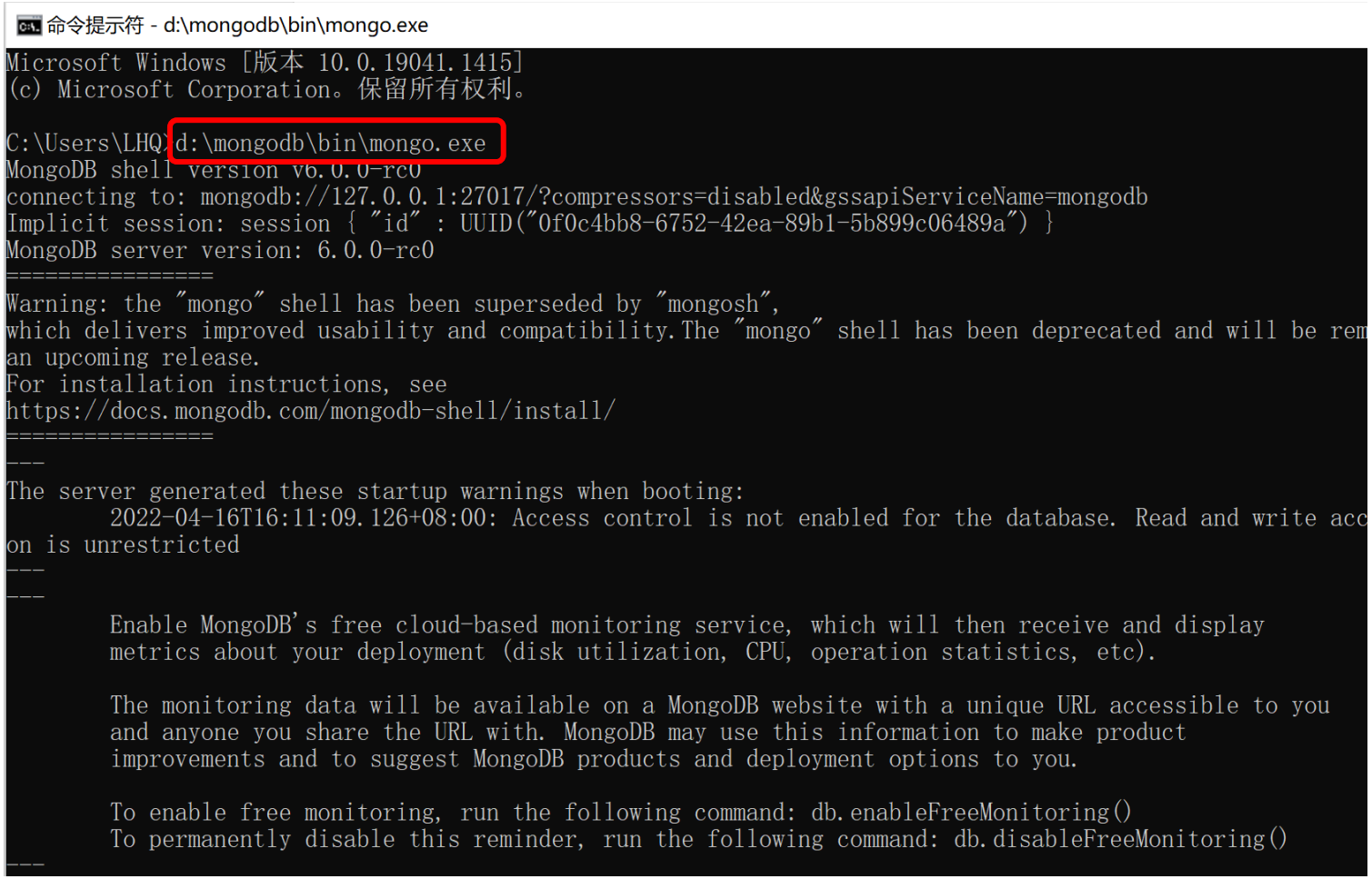
```
Microsoft Windows [版本 10.0.19041.1415]
(c) Microsoft Corporation。保留所有权利。

C:\Users\LHQ>D:\mongodb\bin\mongod --dbpath D:\data\db
{"t":{"$date":"2022-04-16T16:27:35.107+08:00"},"s":"I",  "c":"NETWORK",  "id":491570
ngExternalClient":{"minWireVersion":0,"maxWireVersion":17},"incomingInternalClient":
reVersion":17},"isInternalClient":true}}}
{"t":{"$date":"2022-04-16T16:27:35.108+08:00"},"s":"I",  "c":"CONTROL",  "id":23285,
0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2022-04-16T16:27:35.109+08:00"},"s":"I",  "c":"NETWORK",  "id":4648602
{"t":{"$date":"2022-04-16T16:27:35.111+08:00"},"s":"I",  "c":"REPL",     "id":5123008
rvice":"TenantMigrationDonorService","namespace":"config.tenantMigrationDonors"}}
{"t":{"$date":"2022-04-16T16:27:35.111+08:00"},"s":"I",  "c":"REPL",     "id":5123008
rvice":"TenantMigrationRecipientService","namespace":"config.tenantMigrationRecipient
{"t":{"$date":"2022-04-16T16:27:35.111+08:00"},"s":"I",  "c":"REPL",     "id":5123008
rvice":"ShardSplitDonorService","namespace":"config.tenantSplitDonors"}}
{"t":{"$date":"2022-04-16T16:27:35.111+08:00"},"s":"I",  "c":"CONTROL",  "id":5945603
{"t":{"$date":"2022-04-16T16:27:35.112+08:00"},"s":"I",  "c":"CONTROL",  "id":4615611
7017,"dbPath":"D:/data/db","architecture":"64-bit","host":"LAPTOP-UVIDLP10"}}
{"t":{"$date":"2022-04-16T16:27:35.112+08:00"},"s":"I",  "c":"CONTROL",  "id":23398,
":{"targetMinOS":"Windows 7/Windows Server 2008 R2"}}
{"t":{"$date":"2022-04-16T16:27:35.112+08:00"},"s":"I",  "c":"CONTROL",  "id":23403,
0.0-rc0","gitVersion":"87a5b5494b4a5a3647cf24c96557811d7987c6b7","modules":[],"alloca
t_arch":"x86_64"}}}}
```

02-MongoDB下载安装

- 连接MongoDB

在命令窗口中运行 mongo.exe 命令即可连接上 MongoDB，执行如下命令：

```
D:\mongodb\bin\mongo.exe
```

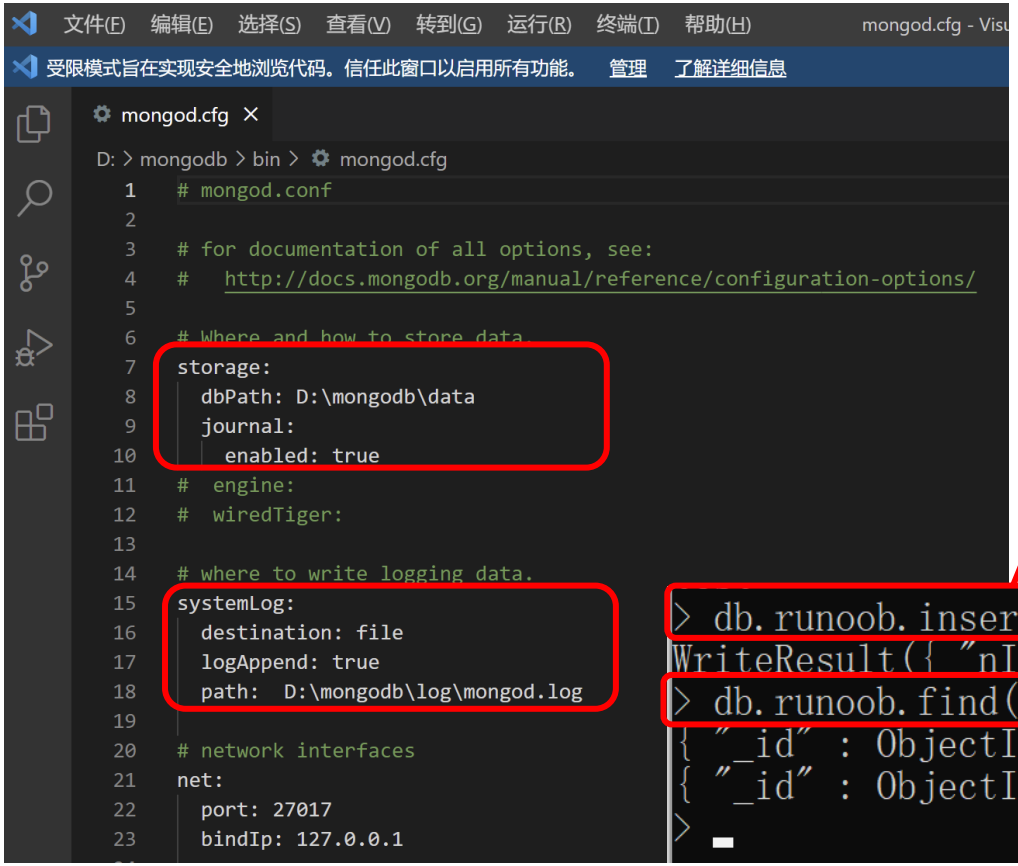


02-MongoDB下载安装

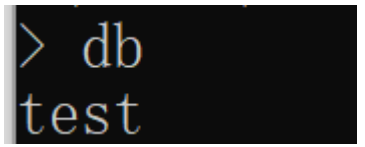
- 配置 MongoDB 服务

一些新版本的 MongoDB 安装时已经自行完成大部分配置，如果以下目录已经存在，可以直接跳过这部分内容。

创建配置文件



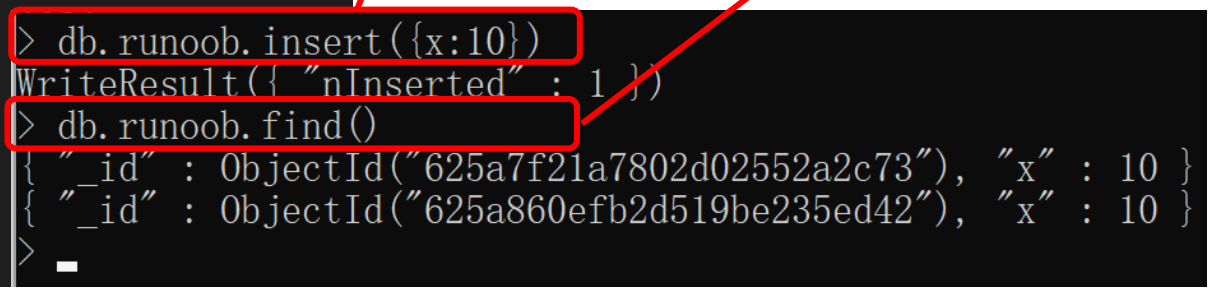
db 命令用于查看当前操作的文档（数据库）：



插入一些简单的记录并查找它：

将数字 10 插入到 runoob 集合的 x 字段

使用find() 方法来读取集合中的文档

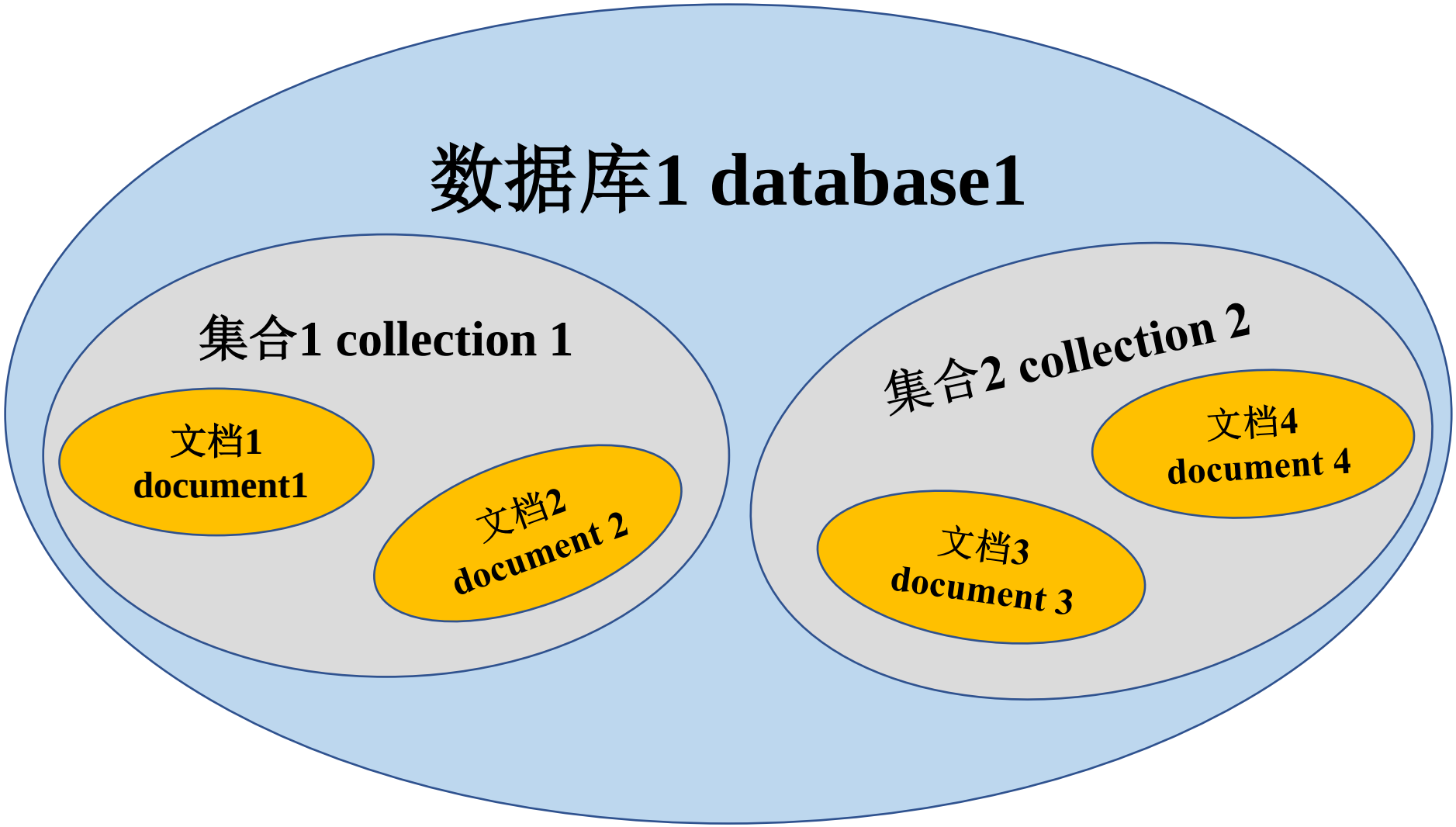


目录 | contents

MongoDB概念解析

SQL术语/概念	MongoDB术语/概念	解释/说明
database	database	数据库
table	collection	数据库表/集合
row	document	数据记录行/文档
column	field	数据字段/域
index	index	索引
table joins		表连接，MongoDB不支持
primary key	primary key	主键，MongoDB自动将_id字段设置为主键

MongoDB层次结构图



MongoDB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。
- 单个实例可容纳多个独立的数据库，各含自己的集合与权限

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
> db
test
> use local
switched to db local
> db
local
>
```

"show dbs" 命令可以显示所有数据的列表。

执行 "db" 命令可以显示当前数据库对象或集合。

运行"use"命令，可以连接到一个指定的数据库。

- 数据库也通过名字来标识。
- 数据库名可为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。

MongoDB 文档（Document）

文档是一组键值(key-value)对(即 BSON)。

{“ site ” : “ https://www.baidu.com/index.php?tn”, “ name ” : "百度"}

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)。**

注意：

- 文档中的键/值对是有序的。
- 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)。
- MongoDB区分类型和大小写。
- MongoDB的文档不能有重复的键。
- 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。

MongoDB 集合（Collection）

- 集合就是 MongoDB 文档组，类似于 RDBMS 中的表格。
- MongoDB 集合存在于数据库中，没有固定的结构

对集合可以插入不同格式和类型的数据，但通常插入集合的数据会有一定的关联性。

```
{"site":"www.baidu.com"}
{"site":"www.google.com","name":"Google"}
{"site":"www.runoob.com","name":“Mongo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$。

MongoDB 集合（Collection）

capped collections 固定大小的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
- 更新后的文档不可以超过之前文档的大小

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

注意

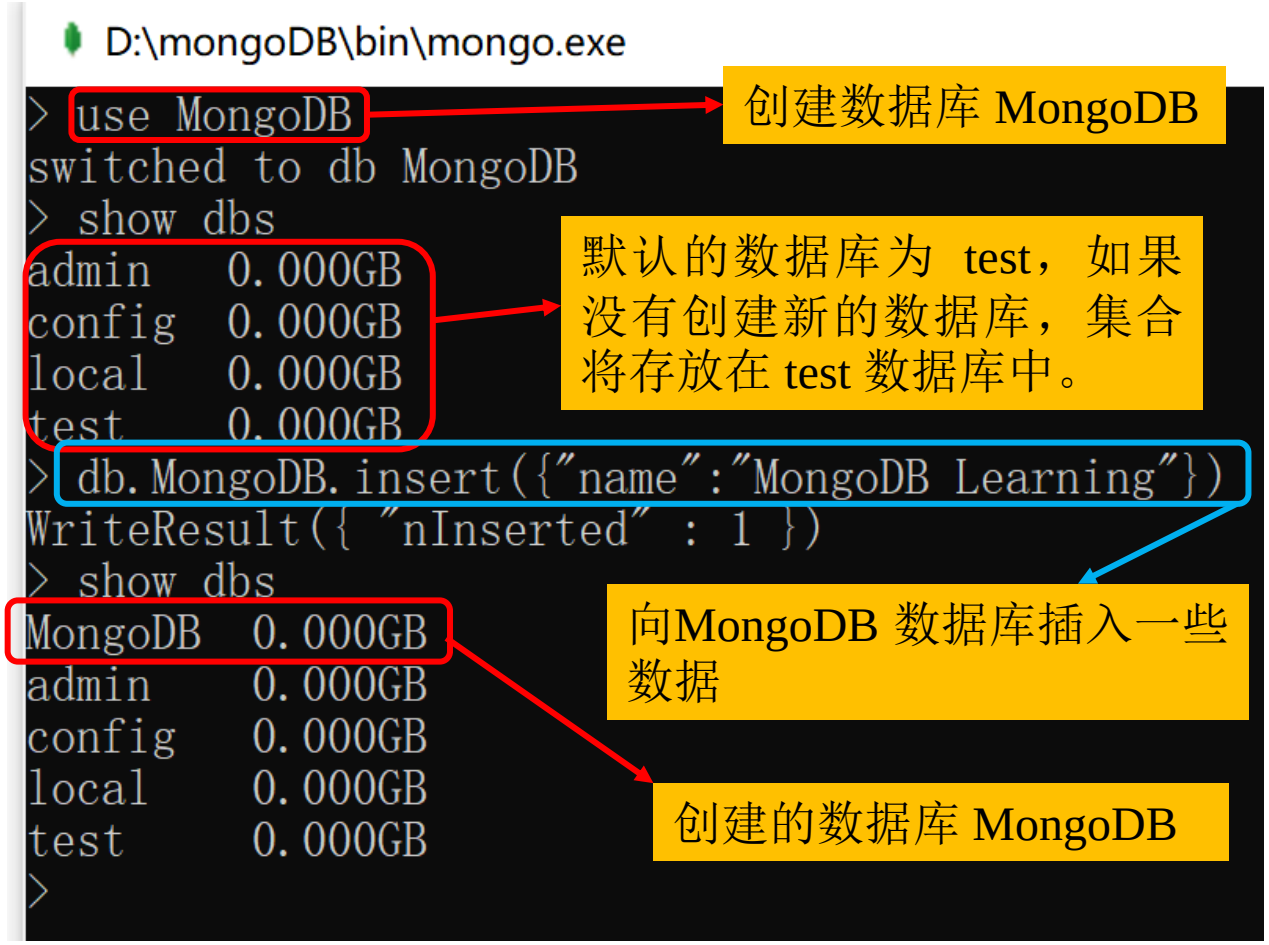
- 在 capped collection 中，你能添加新的对象。
- 能进行更新，然而，对象不会增加存储空间。如果增加，更新就会失败 。
- 使用 Capped Collection 不能删除一个文档，可以使用 drop() 方法删除 collection 所有的行。
- 删除之后，你必须显式的重新创建这个 collection。
- 在32bit机器中，capped collection 最大存储为 1e9(1X109)个字节。

03-MongoDB数据类型

数据类型	描述
String	字符串。在 MongoDB 中，UTF-8 编码的字符串才是合法的。
Integer	用于存储整型数值。根据采用的服务器，可分为 32 位或 64 位。
Boolean	布尔值。用于存储布尔值（ture|false）。
Double	双精度浮点值。用于存储浮点值。
Min/Max keys	将一个值与 BSON（二进制的 JSON）元素的最低值和最高值相对比。
Array	用于将数组或列表或多个值存储为一个键。
Timestamp	时间戳。记录文档修改或添加的具体时间。
Object	用于内嵌文档。
Null	用于创建空值。
Symbol	符号。基本上等同于字符串类型，但一般用于采用特殊符号类型的语言。
Date	日期时间。用 UNIX 时间格式来存储当前日期或时间。
Object ID	对象 ID。用于创建文档的 ID。
Binary Data	二进制数据。用于存储二进制数据。
Code	代码类型。用于在文档中存储 JavaScript 代码。
Regular expression	正则表达式类型。用于存储正则表达式。

MongoDB 创建数据库

use DATABASE_NAME



注意:

在 MongoDB 中，集合只有在内容插入后才会创建!

就是说，创建集合(数据表)后要再插入一个文档(记录)，集合才会真正创建。

MongoDB 删除数据库

db.dropDatabase()

MongoDB 创建集合

参数说明：

```
db.createCollection(name, options)
```

name: 要创建的集合名称

options: 可选参数, 指定有关内存大小及索引的选项

options 可以是如下参数：

字段	类型	描述
Capped	Boolean	true：创建固定集合。固定集合是指有着固定大小的集合，当达到最大值时，它会自动覆盖最早的文档。 当该值为 true 时，必须指定 size 参数。
autoIndexId	Boolean	3.2 之后不再支持该参数。如为 true，自动在 _id 字段创建索引。默认为 false。
size	数值	为固定集合指定一个最大值，即字节数。 如果 capped 为 true，也需要指定该字段。
max	数值	指定固定集合中包含文档的最大数量。

在插入文档时，MongoDB 首先检查固定集合的 size 字段，然后检查 max 字段。

MongoDB 删除集合

db.collection.drop()

D:\mongoDB\bin\mongo.exe

```
> use MongoDB
switched to db MongoDB
> db.MongoDB.insert({"name":"MongoDB Learning"})
WriteResult({ "nInserted" : 1 })
> db.createCollection("MongoDBcollection")
{ "ok" : 1 }
> db.createCollection("MongoDBcol", {capped: true, autoIndexId: true, size: 500, max: 10})
{
        "note" : "The autoIndexId option is deprecated and will be removed in a future release",
        "ok" : 1
}
> show collections
MongoDB
MongoDBcol
MongoDBcollection
> db.MongoDBcollection.drop()
true
> show tables
MongoDB
MongoDBcol
>
```

创建集合 MongoDBcollection

查看已有集合，也可用show tables

不需要创建集合。当插入一些文档时，MongoDB会自动创建集合。

删除集合 MongoDBcollection

MongoDB 插入文档

使用 insert() 或 save() 方法向集合中插入文档，语法如下：

```
db.COLLECTION_NAME.insert(document)
db.COLLECTION_NAME.save(document)
```

- save()：如果 _id 主键存在则更新数据，如果不存在就插入数据。

  该方法新版本中已废弃，可以使用 db.collection.insertOne() 或 db.collection.replaceOne() 来代替。

- insert()：若插入的数据主键已经存在，则会抛 org.springframework.dao.DuplicateKeyException 异常，提示主键重复，不保存当前数据。

```
> db.MongoDB.insert({title: 'MongoDB 教程', description: 'MongoDB 是一个 Nosql 数据库'})
WriteResult({ "nInserted" : 1 })
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d5eec43843c35f5000919"), "title" : "MongoDB 教程", "description" :
"MongoDB 是一个 Nosql 数据库" }
```

查看已插入文档：

03-MongoDB常用操作

西北工业大学 NORTHWESTERN POLYTECHNICAL UNIVERSITY

MongoDB 插入文档

3.2版本之后：

db.collection.insertOne() ◘ 用于向集合插入一个新文档：

db.collection.insertMany() ◘ 用于向集合插入一个多个文档

```
> AddDoc = ({title: 'MongoDB 教程', description: 'MongoDB是一个Nosql数据库'})
{ "title" : "MongoDB 教程", "description" : "MongoDB是一个Nosql数据库" }
> db. MongoDBcol.insertOne(AddDoc)
{
        "acknowledged" : true,
        "insertedId" : ObjectId("625d61cf43843c35f500091a")
}
> db.MongoDBcol.find()
{ "_id" : ObjectId("625d61cf43843c35f500091a"), "title" : "MongoDB 教程", "description" :
"MongoDB是一个Nosql数据库" }
```

- ◘ 将数据定义为一个变量
- ◘ 执行插入操作

```
> var y = db.MongoDBcol.insertMany([{"Var": 4},{"No": 3}])
> y
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("625d646843843c35f500091c"),
                ObjectId("625d646843843c35f500091d")
        ]
}
```

- ◘ 定义变量
- ◘ 执行多条数据插入操作

03-MongoDB常用操作

西北工業大學 NORTHWESTERN POLYTECHNICAL UNIVERSITY

MongoDB ObjectId

- ObjectId是"_id"的默认类型。
- ObjectId使用12字节的存储空间，每个字节二位十六进制数字，是一个24位的字符串

```
> y
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("625d646843843c35f500091c"),
                ObjectId("625d646843843c35f500091d")
        ]
```

0 | 1 | 2 | 3	4 | 5 | 6	7 | 8	9 | 10 | 11
时间戳	机器	PID	计数器

- 时间戳：时间不断变化的
- 机器：主机的唯一标识码。通常是机器主机名的散列值，这样可以确保不同主机生成不同的ObjectId，不产生冲突。
- PID:为了确保在同一台机器上并发的多个进程产生的ObjectId是唯一的，所以加上进程标识符(PID).
- 计数器：前9个字节保证了同一秒钟不同机器不同进程产生的ObjectId是唯一的。后3个字节就是一个自动增加的计数器，确保相同进程同一秒产生的ObjectId也是不一样。同一秒最多允许每个进程拥有16 777 216个不同的ObjectId。

MongoDB 更新文档

MongDB 使用 update() 和 save() 方法来更新集合中的文档

update

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

- query: update的查询条件，类似sql update查询内where后面的。
- update: update的对象和一些更新的操作符（如$,$inc...）等，也可以理解为sql update查询内set后面的
- upsert: 可选，这个参数的意思是，如果不存在update的记录，是否插入objNew,true为插入，默认是false，不插入。
- multi: 可选，mongodb 默认是false,只更新找到的第一条记录，如果这个参数为true,就把按条件查出来多条记录全部更新。
- writeConcern :可选，抛出异常的级别。

MongoDB 更新文档

MongDB 使用 update() 和 save() 方法来更新集合中的文档

```
D:\mongoDB\bin\mongo.exe
> show dbs
MongoDB  0.000GB
admin    0.000GB
config   0.000GB
local    0.000GB
test     0.000GB
> use MongoDB
switched to db MongoDB
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d5eec43843c35f5000919"), "title" : "MongoDB", "description" : "MongoDB 是一个 Nosql 数据库" }
> db.MongoDB.update({'title':'MongoDB'},{$set:{'title':'MongoDB Learning'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.MongoDB.find().pretty()
{ "_id" : ObjectId("625d57e543c35f5000918"), "name" : "MongoDB Learning" }
{
        "_id" : ObjectId("625d5eec43843c35f5000919"),
        "title" : "MongoDB Learning",
        "description" : "MongoDB 是一个 Nosql 数据库"
}
```

1 2 3 4 5

- 使得查询出来的相关数据在命令行展示得更加美观

修改多条相同的文档，则需要设置 multi 参数为 true。

>db. MongoDB.update({'title':'MongoDB '},{$set:{'title':'MongoDB Learning'}},{multi:true})

03-MongoDB常用操作

西北工業大學 NORTHWESTERN POLYTECHNICAL UNIVERSITY

MongoDB 更新文档

使用 update() 和 save() 方法来更新集合中的文档

save

- 通过传入的文档来替换已有文档，_id 主键存在就更新，不存在就插入：

```
db.collection.save(
   <document>,
  {     writeConcern: <document>
   }
)
```

参数说明：

- document : 文档数据。
- writeConcern : 可选，抛出异常的级别。

- 更新时需带主键，否则将插入新的内容并创建相应主键

```
D:\mongoDB\bin\mongo.exe
> use MongoDB
switched to db MongoDB
> show tables
MongoDB
MongoDBcol
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d5eec43843c35f5000919"), "title" : "MongoDB Learning", "description" : "MongoDB 是一个 Nosql 数据库" }
{ "_id" : ObjectId("625d70324e8434e0bc3d4177"), "title" : "MongoDB 教程", "description" : "MongoDB属于Nosql数据库" }
> db.MongoDB.save({ "_id" : ObjectId("625d5eec43843c35f5000919"), "title" : "MongoDB", "description" : "一个Nosql 数据库" }
... )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d5eec43843c35f5000919"), "title" : "MongoDB", "description" : "一个Nosql 数据库" }
{ "_id" : ObjectId("625d70324e8434e0bc3d4177"), "title" : "MongoDB 教程", "description" : "MongoDB属于Nosql数据库" }
>
```

MongoDB 删除文档

MongoDB remove() 函数用来移除集合中的数据

```
db.collection.remove(
   <query>,
   {
      justOne: <boolean>,
      writeConcern: <document>
   }
)
```

参数说明：

- query :（可选）删除的文档的条件。
- justOne :（可选）如果设为 true 或 1，则只删除一个文档，如果不设置该参数，或使用默认值 false，则删除所有匹配条件的文档。
- writeConcern :（可选）抛出异常的级别。

```
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d5eec43843c35f5000919"), "title" : "MongoDB", "description" : "一个Nosql 数据库" }
{ "_id" : ObjectId("625d70324e8434e0bc3d4177"), "title" : "MongoDB 教程", "description" : "MongoDB属于Nosql数据库" }
> db.MongoDB.remove({'title':'MongoDB'})
WriteResult({ "nRemoved" : 1 })
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d70324e8434e0bc3d4177"), "title" : "MongoDB 教程", "description" : "MongoDB属于Nosql数据库" }
>
```

MongoDB 删除文档

只删除第一条找到的记录可以设置 justOne 为 1

```
db.COLLECTION_NAME.remove(DELETION_CRITERIA,1)
```

删除所有数据，可以使用以下方式

```
db.COLLECTION_NAME.remove({})
```

```
> db.MongoDB.find()
{ "_id" : ObjectId("625d57e543843c35f5000918"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625d70324e8434e0bc3d4177"), "title" : "MongoDB 教程", "description" : "MongoDB属于Nosql数据库" }
> db.MongoDB.remove({})
WriteResult({ "nRemoved" : 2 })
> db.MongoDB.find()
>
```

- 删除了两条数据
- 没有数据

此外，还有目前官方推荐使用的 deleteOne() 和 deleteMany() 方法

- 删除集合全部文档。 db.COLLECTION_NAME.deleteMany({})
- 删除 status 等于 A 的全部文档 db.COLLECTION_NAME.deleteMany({ status : "A" })
- 删除 status 等于 D 的一个文档 db.COLLECTION_NAME. deleteOne({ status: "D" })

MongoDB 查询文档

MongoDB 查询文档使用 find() 方法，以非结构化的方式来显示所有文档

db.collection.find(query, projection)

db.col.find().pretty()

- query：可选，使用查询操作符指定查询条件
- projection ：可选，使用投影操作符指定返回的键。查询时返回文档中所有键值， 只需省略该参数即可（默认省略）。

除使用 find() 方法，findOne() 方法可只返回一个文档

```
命令提示符 - D:\mongoDB\bin\mongo.exe
> db.MongoDB.find()
{ "_id" : ObjectId("625e73b472c77a92a1225dd8"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625e73cd72c77a92a1225dd9"), "title" : "MongoDB 教程", "description" : "MongoDB 是一个 Nosql 数据库" }
> db.MongoDB.findOne()
{ "_id" : ObjectId("625e73b472c77a92a1225dd8"), "name" : "MongoDB Learning" }
> db.MongoDB.find().pretty()
{ "_id" : ObjectId("625e73b472c77a92a1225dd8"), "name" : "MongoDB Learning" }
{
        "_id" : ObjectId("625e73cd72c77a92a1225dd9"),
        "title" : "MongoDB 教程",
        "description" : "MongoDB 是一个 Nosql 数据库"
}
>
```

MongoDB 查询文档

MongoDB 查询文档使用 find() 方法，以非结构化的方式来显示所有文档

db.collection.find(query, projection)

db.col.find().pretty()

- query：可选，使用查询操作符指定查询条件
- projection ：可选，使用投影操作符指定返回的键。查询时返回文档中所有键值， 只需省略该参数即可（默认省略）。

除使用 find() 方法，findOne() 方法可只返回一个文档

MongoDB 条件语句查询

操作	格式	RDBMS中的类似语句
等于	{<key>:<value>}	where = ''
小于	{<key>:{$lt:<value>}}	where <
小于或等于	{<key>:{$lte:<value>}}	where <=
大于	{<key>:{$gt:<value>}}	where >
大于或等于	{<key>:{$gte:<value>}}	where >=
不等于	{<key>:{$ne:<value>}}	where !=

db.col.find({"by":“MongoDB教程"}).pretty()		where by = ‘MongoDB教程’
db.col.find({"likes":{$lt:50}}).pretty()		where likes < 50
db.col.find({"likes":{$lte:50}}).pretty()	⟺	where likes <= 50
db.col.find({"likes":{$gt:50}}).pretty()		where likes > 50
db.col.find({"likes":{$gte:50}}).pretty()		where likes >= 50
db.col.find({"likes":{$ne:50}}).pretty()		where likes != 50

03-MongoDB常用操作

西北工业大学 NORTHWESTERN POLYTECHNICAL UNIVERSITY

MongoDB AND条件

MongoDB 的 find() 方法可以传入多个键(key)，每个键(key)以逗号隔开。

```
> db.col.find({key1:value1, key2:value2}).pretty()
```

```
> db.MongoDB.find()
{ "_id" : ObjectId("625e73b472c77a92a1225dd8"), "name" : "MongoDB Learning" }
{ "_id" : ObjectId("625e73cd72c77a92a1225dd9"), "title" : "MongoDB 教程", "description" : "MongoDB 是一个 Nosql 数据库" }
> db.MongoDB.find({"title" : "MongoDB 教程", "description" : "MongoDB 是一个 Nosql 数据库"}).pretty()
{
        "_id" : ObjectId("625e73cd72c77a92a1225dd9"),
        "title" : "MongoDB 教程",
        "description" : "MongoDB 是一个 Nosql 数据库"
}
>
```

类似于 WHERE 语句：

WHERE title= 'MongoDB 教程' AND description='MongoDB 是一个 Nosql 数据库'

MongoDB OR条件

MongoDB OR 条件语句使用关键字 $or

```
>db.col.find({$or: [{key1: value1}, {key2:value2}]}).pretty()
```

MongoDB 索引

- 索引存储在一个易于遍历读取的数据集合中。
- 索引能提高读操作和查询性能
  - 没有索引，必须扫描集合中的每一个文档，然后选择与查询条件匹配的文档
- 索引是对数据库表中一列或多列的值进行排序的一种特殊数据结构

```
> db.collection.createIndex(keys, options)
```

- Key 值为要创建的索引字段，1 为指定按升序创建索引，-1为指定按降序来创建
- 设置字段创建索引：

```
db.MongoDB.createIndex({"title":1})
```

- 设置使用多个字段创建索引：

```
db.MongoDB.createIndex({"title":1,"description":-1})
```

- 删除索引：

```
db.MongoDB.dropIndexes()
```

目录 | contents

GridFS

- 一种将大型文件存储在MongoDB 数据库中的文件规范
  - 所有官方支持的驱动均实现了 GridFS 规范。
- 为什么要用GridFS?
  - GridFS规范将一个大文件分割成为多个较小的文档。
  - 存储和恢复那些超过16M（BSON文件限制）的文件(如：图片、音频、视频等)。
  - 将大文件对象分割成多个小的chunk(文件片段),一般为256k/ 个，每个chunk将作为MongoDB的一个文档(document)被存储在chunks集合。
  - 用两个集合来存储一个文件：fs.files与fs.chunks。

GridFS

如何实现海量存储？

- GridFS 规范指定了一个将文件分块的标准 。
- 每个文件都将在文件集合对象中保存一个元数据对象，一个或多个 chunk 块对象可被组合保存在一个 chunk 块集合中。

大多数情况下，无需了解此规范中细节，将注意力放在各个语言版本的驱动中有关 GridFS API 的部分或是如何使用 mongofiles 工具上。

语言支持?

- Java, Perl, PHP, Python, Ruby 等程序语言均支持，且提供了良好的API接口。

GridFS

GridFS 使用两个表来存储数据：

- files 包含元数据对象。
- chunks 包含其他一些相关信息的二进制块
- 为了使多个 GridFS 命名为一个单一的数据库，文件和块都有一个前缀，默认情况下，前缀是 fs，所以任何默认的 GridFS 存储将包括命名空间 fs.files 和 fs.chunks。

mongofiles 是从命令行操作 GridFS 的一种工具

- 保存文件：

  > G:\mongodb\bin>mongofiles put C:\test.txt

- 获取文件：

  > G:\mongodb\bin>mongofiles get C:\test.txt

- 删除文件：

  > G:\mongodb\bin>mongofiles delete C:\test.txt

GridFS

- 每个文件的实际内容被存在chunks(二进制数据)中

```
{"files_id": ObjectId("534a75d19f54bfec8a2fe44b"),
  "n": NumberInt(0),
  "data": "Mongo Binary Data"}
```

fs.chunks

- 和文件有关的meta数据(filename,content_type,还有用户自定义的属性)将会被存在files集合中

```
{  "filename": "test.txt",
  "chunkSize": NumberInt(261120),
  "uploadDate": ISODate("2014-04-13T11:32:33.557Z"),
  "md5": "7b762939321e146569b07f72c62cca4f",
  "length": NumberInt(646) }
```

fs.file

目录 | contents

MongoDB复制

- 将数据同步在多个服务器的过程
  - 即使一台或者多台服务器出错，也可以保证应用程序正常运行和数据的安全。
- 复制提供了数据的冗余备份，并在多个服务器上存储数据副本，提高了数据的可用性， 并可以保证数据的安全性。
- 复制的特点：
  - 保障数据的安全性
  - 数据高可用性（24*7）
  - 灾难恢复
  - 无需停机维护（如备份，重建索引，压缩）
  - 分布式读取数据

MongDB副本集

副本集是一组服务器，其中一个为**主服务器**，用于处理客户端的请求，还有很多备份服务器，用于保存主服务器的数据副本，如果主服务器崩溃了，备份服务器会**自动将一个成员升级为新的主服务器**。

在MongoDB中创建一个副本集之后就可以使用复制功能了。使用复制功能时，如果有一台服务器宕机了，仍然可以从副本集的其他服务器上访问数据，如果服务器上的数据损坏或者不可访问，可以从副本集的某个成员中创建一份新的数据副本

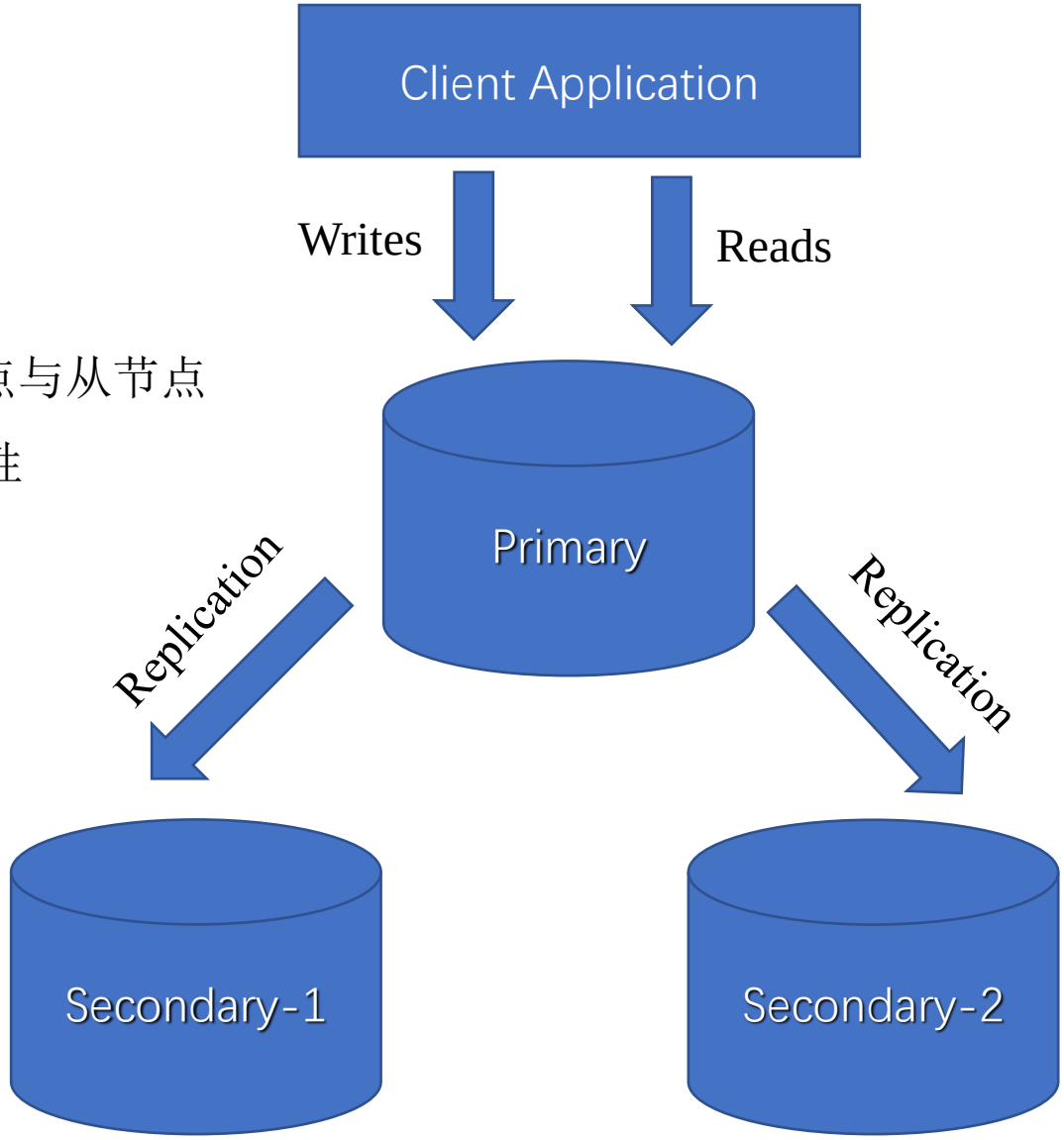
05-MongoDB 复制与分片

西北工业大学 NORTHWESTERN POLYTECHNICAL UNIVERSITY

MongDB复制原理

Replica Sets 副本集：

- 客户端从主节点读取/写入数据
- 客户端进行数据操作时，主节点与从节点进行数据交互保障数据的一致性



副本集架构图

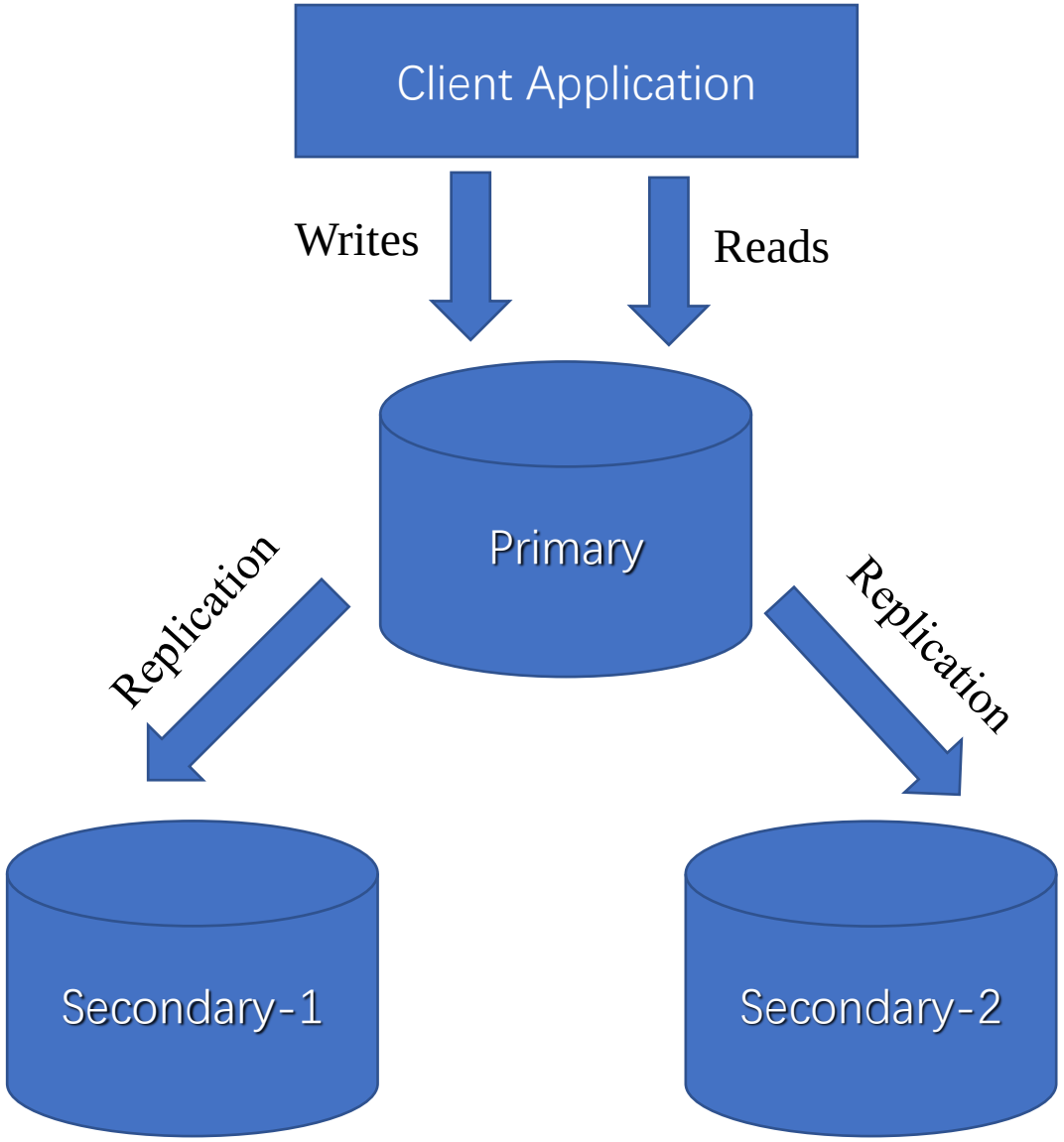
副本集特征：

- N个节点的集群
- 任何节点可作为主节点
- 所有操作写在主节点上
- 自动故障转移
- 自动恢复

MongDB复制原理

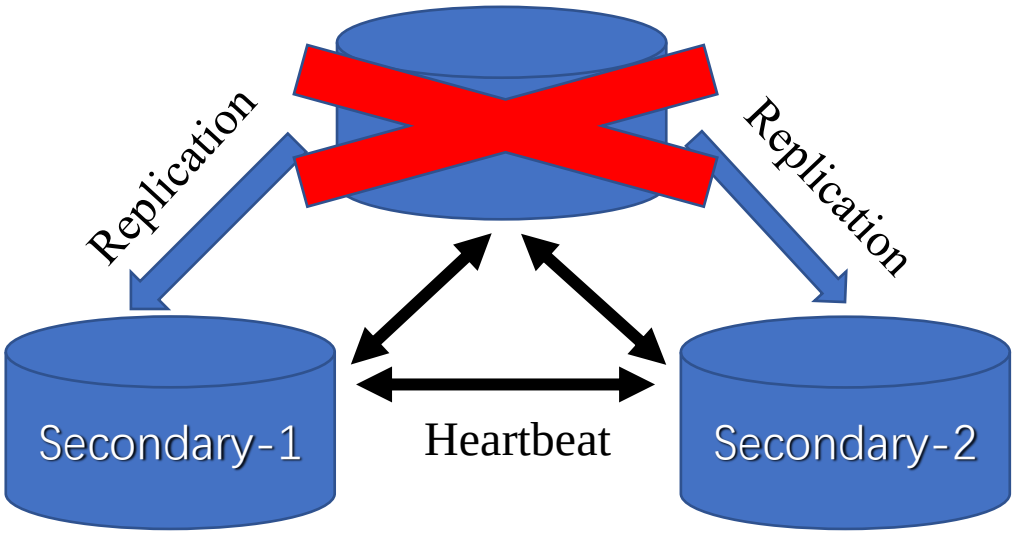
Replica Sets 副本集：

- Primary主服务器，**写操作只能在其身上发生**，通过保存操作日志（oplog），然后异步同步到多个 Secondary 上。
- Secondary从服务器，热机备份主服务器上的数据，分担主机读压力，**当主机故障发生，随时待命接管主机工作**。



副本集架构图

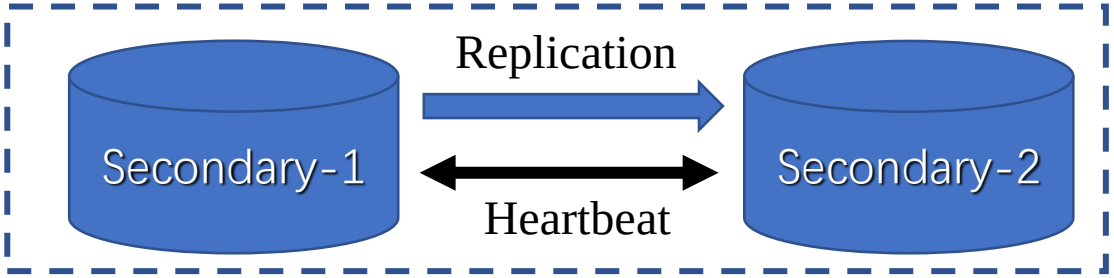
MongDB复制原理



复制集架构图

- Heartbeat监听状态
- 若主服务器失效，则从节点进行选举
- 产生新的主节点

Election for New Primary



MongoDB分片

- 指将用户拆分，将其分散存放在不同机器上的过程，用于处理更大的负载
  - 当存储海量的数据时，一台机器可能不足以存储数据，也可能不足以提供可接受的读写吞吐量。
  - 满足MongoDB数据量大量增长的需求
- 为什么使用分片？
  - 复制所有的写入操作到主节点
  - 延迟的敏感数据会在主节点查询
  - 单个副本集限制在12个节点
  - 当请求量巨大时会出现内存不足
  - 本地磁盘不足
  - 垂直扩展价格昂贵

MongoDB分片

复制与分片的区别：

- 复制是将数据同步的存在不同服务器上，每个节点的数据都是相同的
- 分片每个节点的数据是不相同的，拆分数据

分片机制介绍

- MongoDB的分片是指定一个分片key来进行，数据按范围分成不同的chunk。
- 通过分片能够增加更多的机器，来应对不断增加的负载和数据。
- MongoDB何时分片：

  机器的磁盘不够用

  单个mongod已经不能满足写数据的性能需求

  想将大量的数据放在内存中提高性能

  一般来说，集群先从不分片开始，然后在需要时才转换成分片。

目录 | contents

07-MongoDB 总结

1. MongoDB 是一个介于关系数据库和非关系数据库之间的开源产品
2. 面向集合存储，易存储对象类型的数据。
3. GridFS规范将一个大文件分割成为多个较小的文档。
4. MongoDB通过复制将数据同步在多个服务器的过程。
5. MongoDB通过分片是指将数据拆分，将其分散在不同机器上。

名称	说明
db.collection.insert()	在集合中创建一个新文档。
db.collection.save()	提供insert()和update ()插入新文件的包装。
db.collection.update()	修改集合中的文档。
db.collection.find()	集合上执行查询，并返回一个游标对象。
db.collection.findOne()	执行查询，并返回一个单独的文档。
db.collection.remove()	从集合中删除的文件。


I
A LOT
mongoDB

敬请批评指正！

软件学院 李鸿岐