MongoDB和SQL基础概念对比：

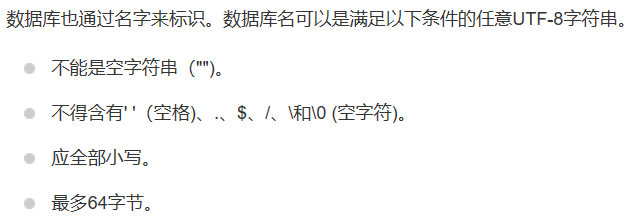
|  |  |  |
| --- | --- | --- |
| SQL术语/概念 | MongoDB术语/概念 | 解释/说明 |
| database | database | 数据库 |
| table | collection | 数据库表/集合 |
| row | document | 数据记录行/文档 |
| column | field | 数据字段/域 |
| index | index | 索引 |
| table joins |  | 表连接，MongoDB不支持 |
| primary key | primary key | 主键，MongoDB自动将\_id字段设置为主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

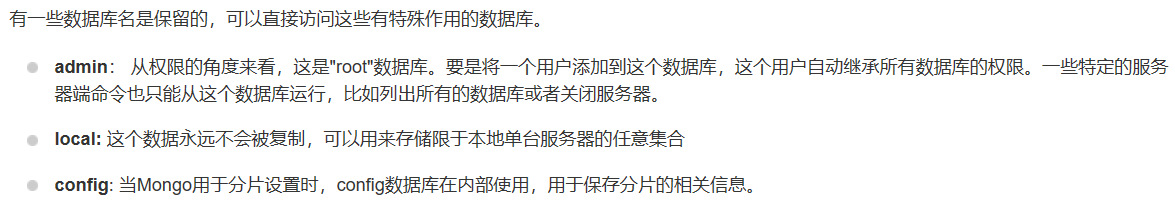
实例对比：



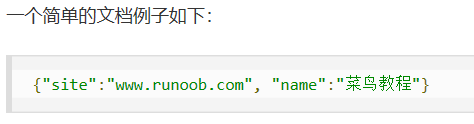
**数据库：**

一个mongodb中可以建立多个数据库，show dbs显示所有数据库，db显示当前数据库对象，use + 数据库名 连接到一个指定的数据库。





**文档：**是一组键值对。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。



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（域） |
| 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/Orcale | 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字符串，还可以是其他数据类型（嵌入的文档）
3. 区分类型和大小写
4. 文档中不能有重复的键
5. 文档的键是字符串

文档键的命名规则：

1. 键不能含有\0（空字符）。这个字符用来表示键的结尾。
2. .和$有特别的意义，只有在特定环境下才使用
3. 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中的表格。集合存在于数据库中，集合没有固定的结构，可以对集合插入不同格式和类型的数据，通常情况下插入集合的数据都会有一定的关联性。



**Capped collections：**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）。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，必须显示的创建一个capped collection，指定一个collection的大小，单位是字节。collection的数据存储空间是提前分配的。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。



**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，如下：



**MongoDB的数据类型：**



几个比较重要的数据类型：





**MongoDB连接：**

**启动：**



当前安装后，默认mongodb是系统的一个服务，并且在系统中设置为了自动启动，可以修改为手动启动。

**连接：**



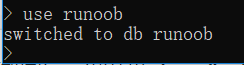
插入：**MongoDB用户名和密码配置：**

详见：<https://blog.csdn.net/fofabu2/article/details/78983741>

**创建MongoDB数据库：**

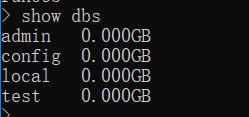
语法：use database\_name，如果数据库不存在，则创建数据库，否则切换到指定数据库。

实例：use runoob

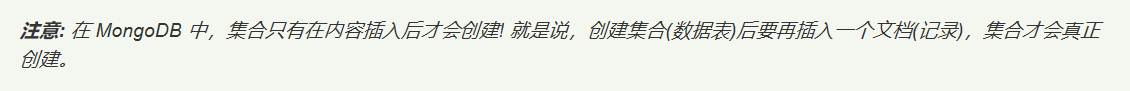




但是，新创建的数据库中没有数据时，使用show dbs，不会显示出来。



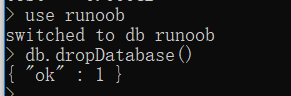
注：MongoDB默认的数据库是test，如果你没有创建新的数据库，集合将存放在test数据库中。



**MongoDB删除数据库：**

语法：db.dropDatabase()，删除当前数据库，默认为test，可以使用db命令查看当前数据库名。

实例：删除runoob数据库



**MongoDB创建集合：**

语法：db.createCollection(name, options)

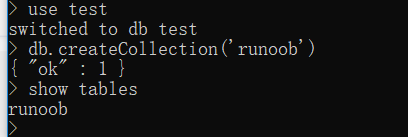
参数说明：

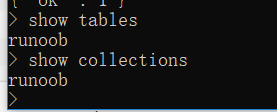
name：要创建的集合名称

options：可选参数，指定有关内存大小及索引选项

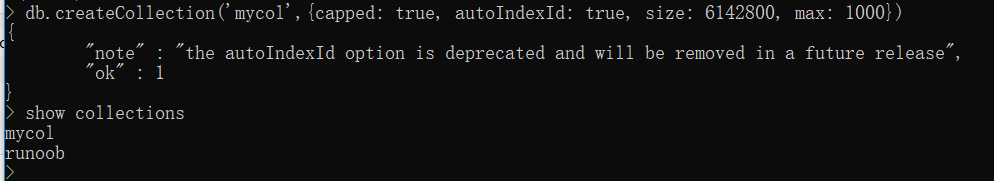


实例：在test数据库中创建runoob集合

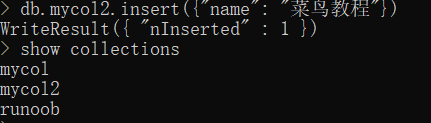




实例：创建固定集合mycol，整个集合空间的大小6142800KB，文档最大个数为1000个





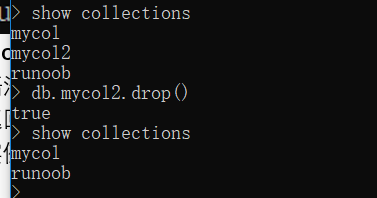


**MongoDB删除集合：**

语法：db.collection\_name.drop()

返回值：如果删除成功，返回true，否则，返回false

实例：删除数据库test中的集合mycol2



**MongoDB插入文档：**

文档的数据结构和JSON基本一样。

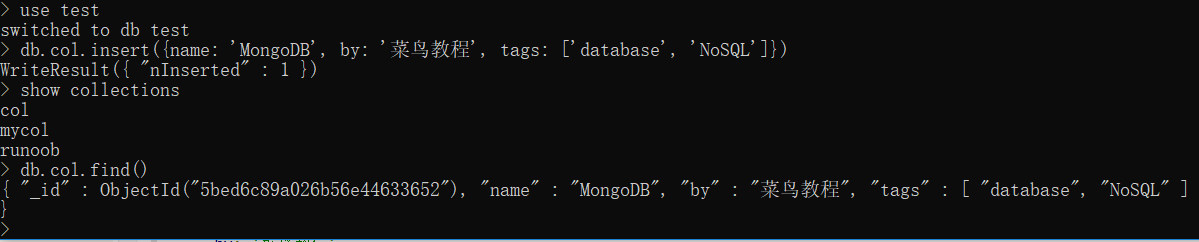
所有存储在集合中的数据都是BSON格式。

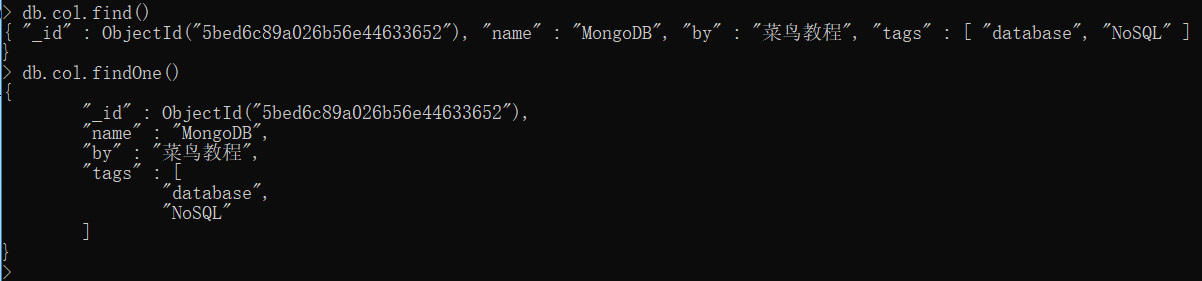
BSON是一种类JSON的一种二进制形式的存储格式，简称Binary JSON。

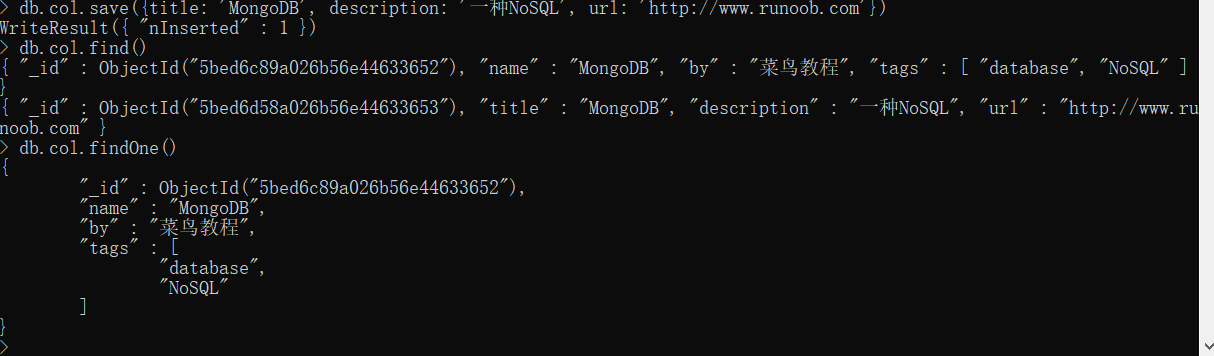
语法：db.collection\_name.insert(document)

db.collection\_name.save(document)

实例：向数据库test中的col集合中插入一条文档











**MongoDB更新文档：**

MongoDB使用update()和save()方法来更新集合中的文档。

**update()方法：**update()方法用于更新已经存在的文档。

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

参数说明：

query：update的查询条件，类似sql update的where后面的。

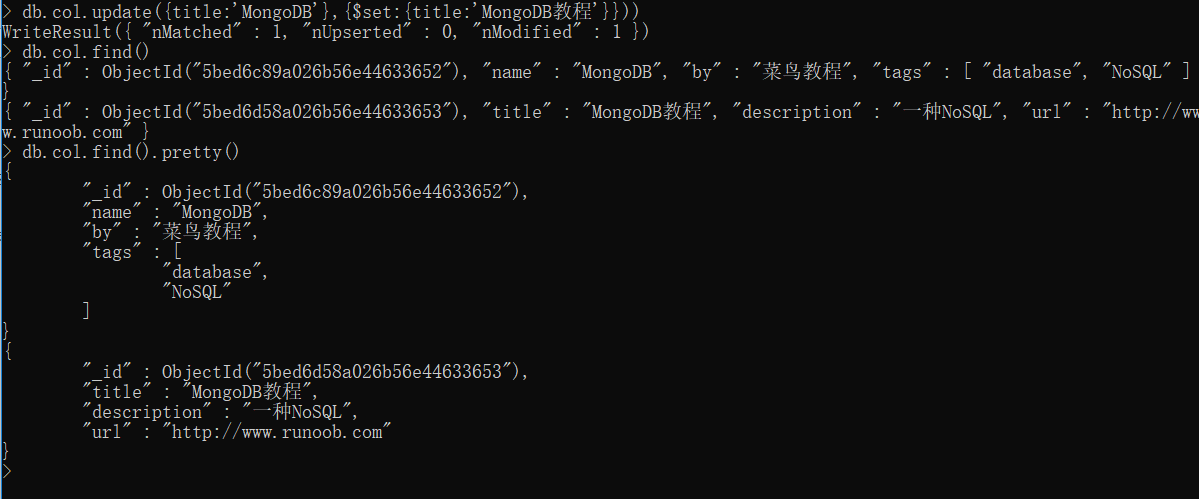
update：update的对象和一些更新的操作符等，也可以理解为sql update查询内set后面的。

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，默认为false，不插入。

multi：可选，mongodb默认为false，只更新找到的第一条记录，如果这个参数为true，就把按条件查出来的多条记录全部更新。

writeConcern：可选，抛出异常的级别。

实例：更新数据库test的col集合中title为“MongoDB”的文档 ，更新其title为“MongoDB教程”



**save()方法：**save()方法通过传入的文档替换已有的文档。

语法：db.collection\_name.save(<document>, {writeConcern: <document>})

参数说明：

document：文档数据

writeConcern：可选，抛出的异常级别

实例：



**MongoDB删除文档：**

remove()函数用来移除集合中的数据。



实例：删除test数据库中col集合中title为“MongoDB教程”的文档



如果想要删除所有数据：db.collection\_name.remove()

注：





**MongoDB查询文档：**

MongoDB使用find()方法以非结构化的方式来显示所有文档。

语法：db.collection\_name.find(query, projection)

参数说明：

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。查询时返回文档中的所有键值，省略该参数。

如果需要以易读的方式来读取数据，可以使用pretty()方法，语法格式：db.collection.find().pretty()，pretty()以格式化的方式来显示所有文档。

**MongoDB与RDBMS where语句比较：**



**MongoDB AND条件：**

find()方法可以传入多个键，每个键以逗号隔开。

语法格式：db.col.find({key1:value1, key2:value2}).pretty()

**MongoDB OR条件：**

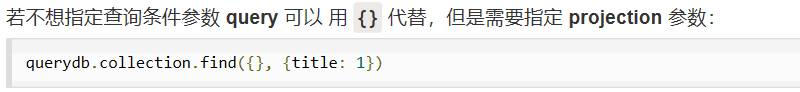
MongoDB OR条件语句使用了关键字$or，语法格式如下：

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

**AND 和 OR联合使用：**





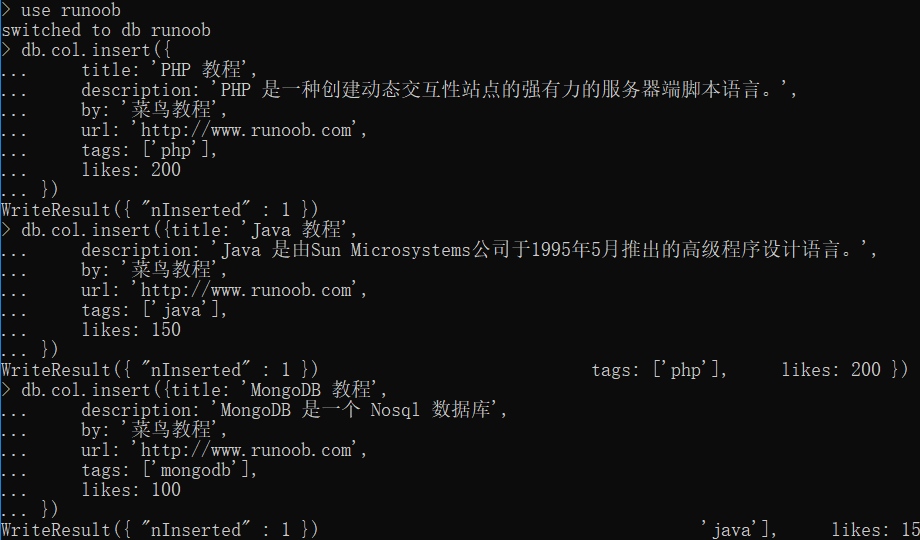


**MongoDB条件操作符：**

描述：lt（小于）gt（大于）lte（小于等于）gte（大于等于）

实例：

1. 插入数据



1. 查询数据



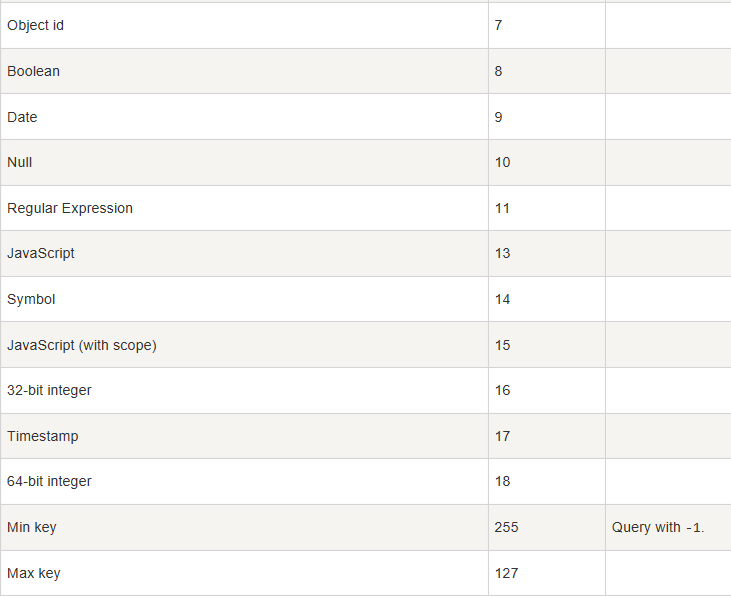


**MongoDB $type操作符：**

$type操作符是基于BSON类型来检索集合中匹配的数据类型，并返回结果。

类型如下：

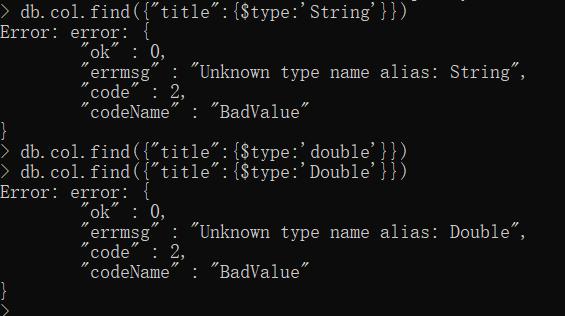




实例：获取col集合中title为String的数据



注意：string首字母小写。



**MongoDB Limit与Skip方法：**

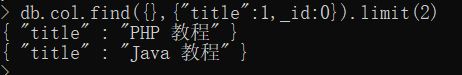
**Limit()方法：**

描述：读取指定数量的数据记录

语法：db.collection\_name.find().limit(number)

参数：number为指定的记录条数

实例：显示查询文档中的两条记录

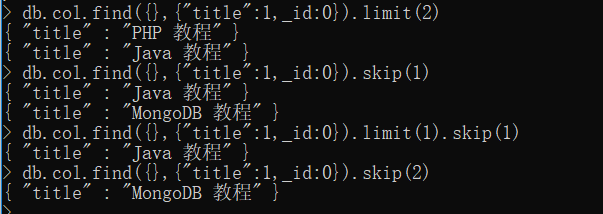


**Skip()方法：**

描述：跳过指定数量的数据

语法：db.collection\_name.find().limit(number).skip(number)

实例：



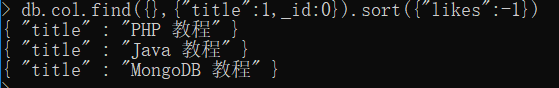
**MongoDB排序：**

**sort()方法：**

描述：对数据进行排序，sort()方法可以通过参数指定排序的字段，并使用1和-1来指定排序的方式，其中1为升序排列，-1为降序排列。

语法：db.collection\_name.find().sort({KEY:-1})

实例：



**MongoDB索引：**

描述：索引通常能够极大的提高查询的效率，如果没有索引，MongoDB在读取数据时必须通过扫描集合中的每个文件并选取那些符合查询条件的记录。这种扫描全集合的查询效率是非常低的，特别在处理大量的数据时，查询可以要花费几十秒甚至几分钟，这对网站的性能是非常致命的。索引是特殊的数据结构，索引存储在一个易于遍历读取的数据集合中，索引是对数据库中一列或多列的值进行排序的一种结构。查询的时候，先查索引，因为索引相比于数据记录是很小的，然后通过索引去定位所要查询的数据。

语法：db.collection\_name.createIndex(keys, options)

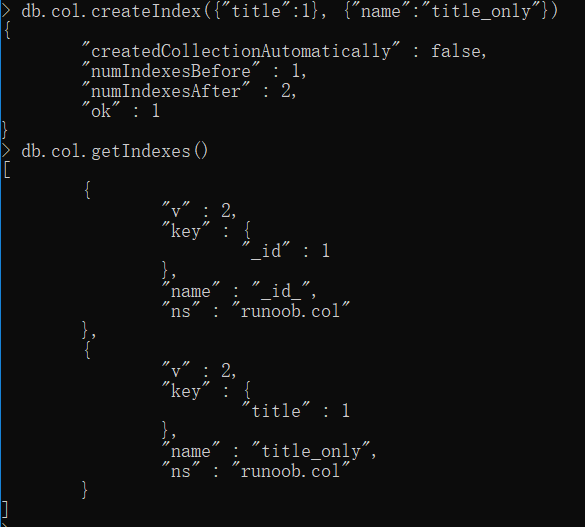
参数：keys是要创建的索引字段，1为指定按升序创建索引，如果想按降序来创建索引指定为-1即可。

可选参数列表：



实例：

1. 单个字段索引



1. 多个字段索引（关系型数据库中称作复合索引）



参考博客：<http://www.cnblogs.com/williamjie/p/9297670.html>

1. 查看索引db.col.getIndexes()
2. 删除索引db.col.dropIndex([indexName])
3. 重建索引db.col.reIndex()

**MongoDB聚合：**

描述：MongoDB中聚合（aggregate）主要用于处理数据（诸如统计平均值，求和等），并返回计算后的数据结果。有点类似sql语句的count(\*)。

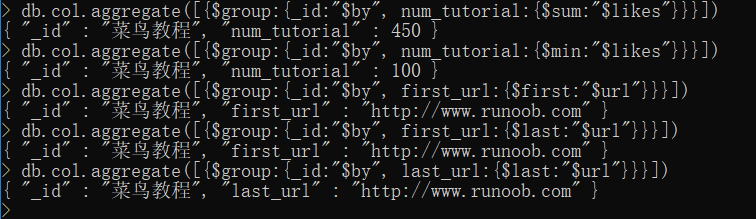
aggregate()方法：MongoDB中聚合的方法使用aggregate()。

语法：db.collection\_name.aggregate(AGGREGATE\_OPERATION)

实例：按照作者进行分组统计条数

  
聚合表达式：





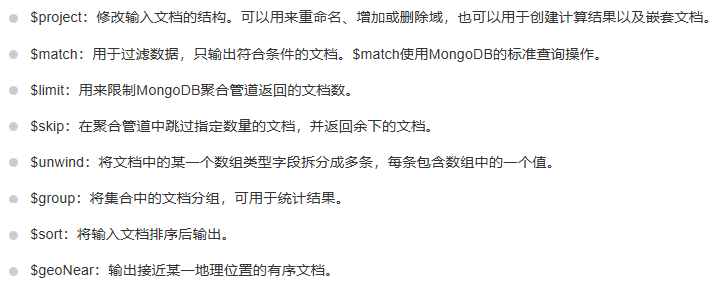
**管道的概念：**

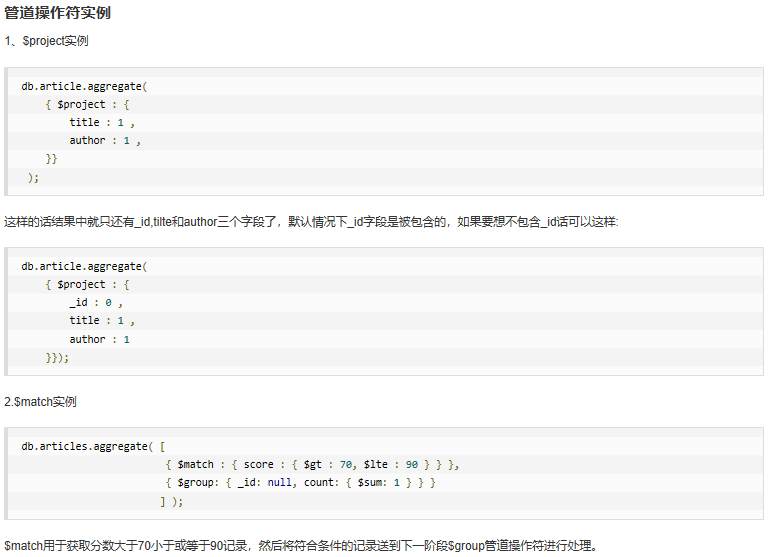
描述：管道在Unix和Linux中一般用于将当前命令的输出结果作为下一个命令的参数。

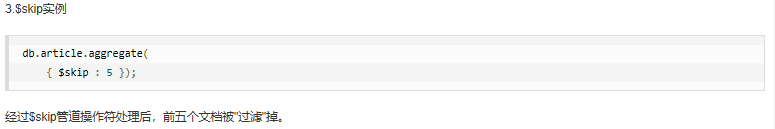
MongoDB的聚合管道将MongoDB文档在一个管道处理完毕后将结果传递给下一个管道处理。管道操作是可以重复的。

表达式：处理输入文档并输出。表达式是无状态的，只能用于计算当前聚合管道的文档，不能处理其他的文档。

聚合框架中常用的几个操作：





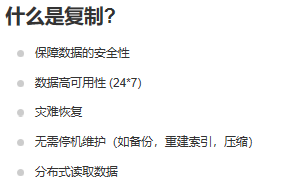


**MongoDB复制（副本集）：**

描述：MongoDB复制是将数据同步在多个服务器的过程。

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。

复制还允许从硬件故障和服务中断中恢复数据。



**MongoDB复制原理：**