MongoDB

## 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 |



## 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

"show dbs" 命令可以显示所有数据的列表。



"db" 命令可以显示当前数据库对象或集合。

"use"命令，可以连接到一个指定的数据库。即切换数据库.

数据库名命名规则：

* 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

文档是一组键值(key-value)对，对应mysql的行。文档不需要设置相同的字段，并且相同字段不需要相同的数据类型，即每一条数据都是独立的，可以有不同的结构。

注意：

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。

## 集合

普通collection：当第一个文档插入时，集合就会被创建。（可以不事先创建）

capped collections（固定大小的collection）：

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。要注意的是指定的存储大小包含了数据库的头信息。

* 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 。
* 数据库不允许进行删除。使用drop()方法删除collection所有的行。
* 注意: 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。

创建集合：

db.createCollection(name, options)

检查创建的集合：

show collections

options选项：

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **描述** |
| capped | Boolean | （可选）如果为true，它启用上限集合。上限集合是一个固定大小的集合，当它达到其最大尺寸会自动覆盖最老的条目。 如果指定true，则还需要指定参数的大小。 |
| autoIndexID | Boolean | （可选）如果为true，自动创建索引\_id字段。默认的值是 false. |
| size | number | （可选）指定的上限集合字节的最大尺寸。如果capped 是true，那么还需要指定这个字段。 |
| max | number | （可选）指定上限集合允许的最大文件数。 |

删除集合：

db.collection.drop()

## 数据类型

|  |  |
| --- | --- |
| **数据类型** | **描述** |
| String | 字符串。存储数据常用的数据类型。在 MongoDB 中，UTF-8 编码的字符串才是合法的。 |
| Integer | 整型数值。用于存储数值。根据你所采用的服务器，可分为 32 位或 64 位。 |
| Boolean | 布尔值。用于存储布尔值（真/假）。 |
| 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。 |
| Arrays | 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。该数据类型基本上等同于字符串类型，但不同的是，它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。你可以指定自己的日期时间：创建 Date 对象，传入年月日信息。 |
| 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。 |

## 创建数据库

use 数据库名

若数据库不存在，则创建数据库

## 删除数据库

db.dropDatabase()

## 删除集合

db.collection.drop()

## 插入文档

db.COLLECTION\_NAME.insert(document)

若集合不在当前数据库中，会自动创建该集合并插入文档。

查看文档

db.COLLECTION\_NAME.find()

db.COLLECTION\_NAME.find().pretty()//数据显示结构化，类似于php的pre

db.COLLECTION\_NAME.save(document)

若不指定\_id,则为添加操作

若指定\_id,为更新操作

## 更新文档

update语法：将文档中的值更新

|  |
| --- |
| db.collection.update(  <query>,  <update>,  {  upsert: <boolean>,  multi: <boolean>,  writeConcern: <document>  }  )  参数说明：  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。 |

save语法：

save() 方法通过传入的文档来替换已有文档。语法格式如下：

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。

db.COLLECTION\_NAME.save({\_id:ObjectId(),NEW\_DATA})

|  |
| --- |
| >db.mycol.save(  {  "\_id" : ObjectId(5983548781331adf45ec7), "title":"Yiibai Yiibai New Topic", "by":"Yiibai Yiibai"  }  )  //替换这个\_id的值 |

## 删除文档

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。默认是全删

writeConcern :（可选）抛出异常的级别。

删除所有（类似truncate）：

db.col.remove({})

## 查询文档

db.collection.find(query, projection)

* **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）。
* db.collection.findOnde(query, projection) //只返回一个

db.col.find({},{"title":1,\_id:0})

没有查询条件，查询结果中包含title，不包含\_id; 1为显示，0为不显示

查询语句：MongoDB 与 RDBMS Where 语句比较

| **操作** | **格式** | **范例** | **RDBMS中的类似语句** |
| --- | --- | --- | --- |
| 等于 | {<key>:<value>} | db.col.find({"by":"菜鸟教程"}).pretty() | where by = '菜鸟教程' |
| 小于 | {<key>:{$lt:<value>}} | db.col.find({"likes":{$lt:50}}).pretty() | where likes < 50 |
| 小于或等于 | {<key>:{$lte:<value>}} | db.col.find({"likes":{$lte:50}}).pretty() | where likes <= 50 |
| 大于 | {<key>:{$gt:<value>}} | db.col.find({"likes":{$gt:50}}).pretty() | where likes > 50 |
| 大于或等于 | {<key>:{$gte:<value>}} | db.col.find({"likes":{$gte:50}}).pretty() | where likes >= 50 |
| 不等于 | {<key>:{$ne:<value>}} | db.col.find({"likes":{$ne:50}}).pretty() | where likes != 50 |

**AND条件**

>db.col.find({key1:value1 **,** key2:value2}).pretty()//多条件用，分割即可

**OR 条件**

>db.col.find(

{

$or: [

{key1: value1}, {key2:value2}

]

}

).pretty()

**AND 和 OR** 联合使用

db.col.find({"likes": {$gt:50} **,** $or: [{"by": "菜鸟教程"},{"title": "MongoDB 教程"}]}).pretty()

**选择显示字段：**

db.COLLECTION\_NAME.find({},{KEY:1}) 第二个中括号指明字段显示详情，1：显示，0：不显示，指明的字段会显示，没指明的都不显示，但是，\_id永远会显示。

## 条件操作符

* > 大于 - $gt
* < 小于 - $lt
* >= 大于等于 - $gte
* <= 小于等于 - $lte

db.col.find({‘likes’ : {$lte : 150}})

用< >联合查询

db.col.find({likes : {$lt :200, $gt : 100}})

## $type数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **类型** | | **数字** | | **备注** |
| Double | 1 | |  | |
| String | 2 | |  | |
| Object | 3 | |  | |
| Array | 4 | |  | |
| Binary data | 5 | |  | |
| Undefined | 6 | | 已废弃。 | |
| Object id | 7 | |  | |
| Boolean | 8 | |  | |
| Date | 9 | |  | |
| Null | 10 | |  | |
| Regular Expression | 11 | |  | |
| JavaScript | 13 | |  | |
| Symbol | 14 | |  | |
| JavaScript (with scope) | 15 | |  | |
| 32-bit integer | 16 | |  | |
| Timestamp | 17 | |  | |
| 64-bit integer | 18 | |  | |
| Min key | 255 | | Query with -1. | |
| Max key | 127 | |  | |

db.col.find({"title" : {$type : 2}})

## Limit与Skip方法

db.col.find().skip(1).limit(1);

skip参数为从第几条开始，默认从0开始，limit()参数为取多少条

## sort()方法

db.COLLECTION\_NAME.find().sort({KEY:1}) // 1 为升序排列，而-1是用于降序排列。

## 索引

查看索引：

db.collectionName.getIndexes()

创建单列索引：

db.collectionName.ensureIndex({field:1/-1})//1：升序;-1:降序

创建多列索引：

db.collectionName.ensureIndex({field1:1/-1,field2:1/-1})

ensureIndex({field:1},**{}**) 接收可选参数，可选参数列表如下：

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| background | Boolean | 建索引过程会阻塞其它数据库操作，background可指定以后台方式创建索引，即增加 "background" 可选参数。 "background" 默认值为**false**。 |
| unique | Boolean | 建立的索引是否唯一。指定为true创建唯一索引。默认值为**false**. |
| name | string | 索引的名称。如果未指定，MongoDB的通过连接索引的字段名和排序顺序生成一个索引名称。 |
| dropDups | Boolean | 在建立唯一索引时是否删除重复记录,指定 true 创建唯一索引。默认值为 **false**. |
| sparse | Boolean | 对文档中不存在的字段数据不启用索引；这个参数需要特别注意，如果设置为true的话，在索引字段中不会查询出不包含对应字段的文档.。默认值为 **false**. |
| expireAfterSeconds | integer | 指定一个以秒为单位的数值，完成 TTL设定，设定集合的生存时间。 |
| v | index version | 索引的版本号。默认的索引版本取决于mongod创建索引时运行的版本。 |
| weights | document | 索引权重值，数值在 1 到 99,999 之间，表示该索引相对于其他索引字段的得分权重。 |
| default\_language | string | 对于文本索引，该参数决定了停用词及词干和词器的规则的列表。 默认为英语 |
| language\_override | string | 对于文本索引，该参数指定了包含在文档中的字段名，语言覆盖默认的language，默认值为 language. |

A、删除单个索引：

db.collectionName.dropIndex({filed:1/-1})

B、删除所有索引：\_id列的索引不会删除。

db.collectionName.dropIndexes()

索引管理：

* 1. 查看所有索引

db.system.indexes.find()

* 1. 分析查询性能及索引

db.collectionName.find(查询表达式).explain()



"cursor" :"BasicCursor" ——>表示索引没有发挥作用

"nscanned":1 ——>表示查询了多少个文档。

"n"，:1 ——>表示返回的文档数量。

"millis":0 ——>表示整个查询的耗时。

"nscannedObjects" : 11, ——>理论上需要扫描多少行

## MongoDB聚合

## PHP中使用mongodb

|  |
| --- |
| $m = new MongoClient(); 默认连接localhost:27017  $db = $m->test; 获取‘test’数据库  $collection = $db->createCollection("runoob");数据库里创建集合  $document = array(  "title" => "MongoDB",  "description" => "database",  "likes" => 100,  "url" => "http://www.runoob.com/mongodb/",  "by", "菜鸟教程"  );  $collection->insert($document); 集合中插入文档  $cursor = $collection->find(); 查找文档  $collection->update(array("title"=>"MongoDB"), array('$set'=>array("title"=>"MongoDB 教程"))); 更新文档  $collection->remove(array("title"=>"MongoDB 教程"), array("justOne" => true));删除title为MongoDB 教程 的一条记录 |