MongoDB 索引

李焕贞

河北师范大学软件学院

https://docs.mongodb.com/manual/indexes/

本章大纲

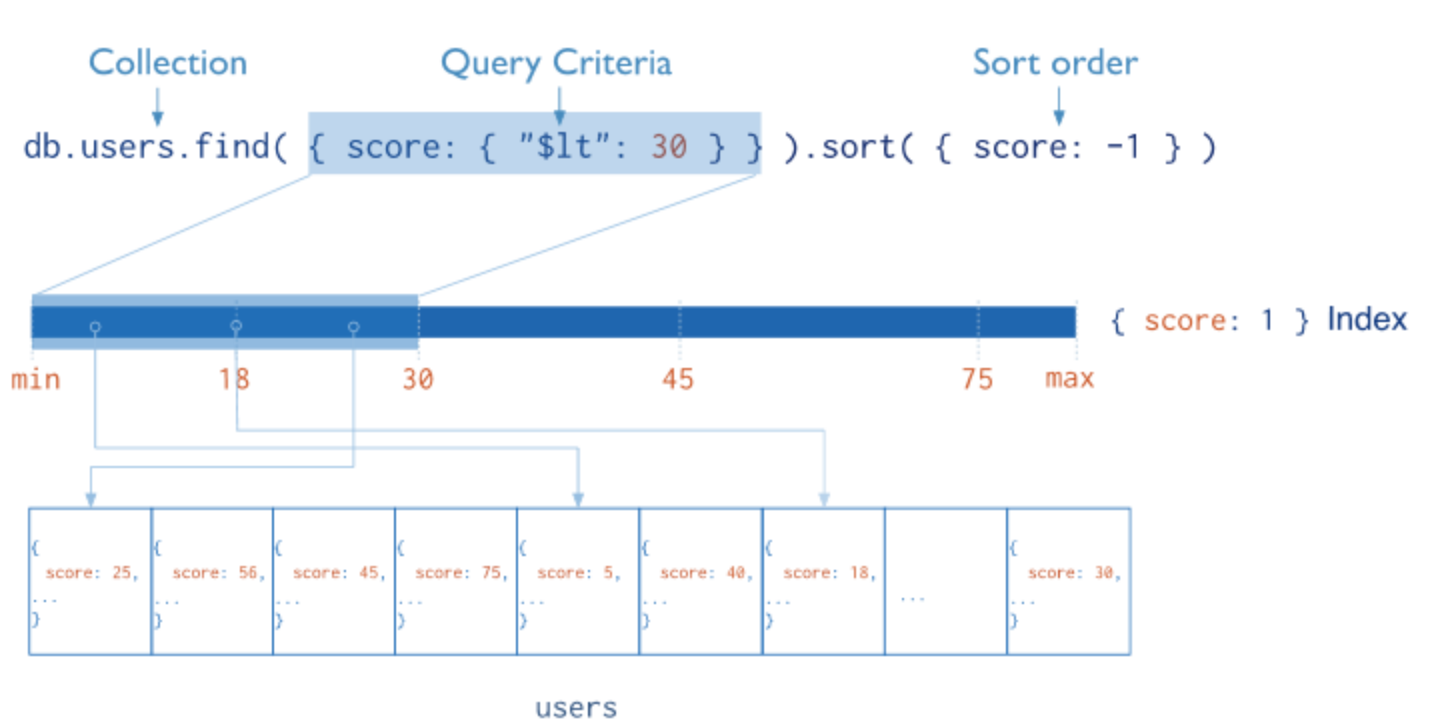
- MongoDB索引的类型
- MongoDB索引的属性
- MongoDB索引的管理

MongoDB索引的类型

数据库索引是对数据库表中一列或多列的值进行排序的一种数据结构，使用索引可快速访问数据库表中的特定信息。

数据库索引的功能类似于书籍的索引，书籍有了索引就不需要翻查整本书。与此类似，在进行查询时，数据库会**首先在索引中**查找，找到相应的条目后，就可以直接跳转到**目标文档**的位置。

MongoDB索引的类型



MongoDB索引的类型

MongoDB索引几乎与关系数据库的索引一样，绝大多数优化关系型数据库索引的技巧同样适用于MongoDB。

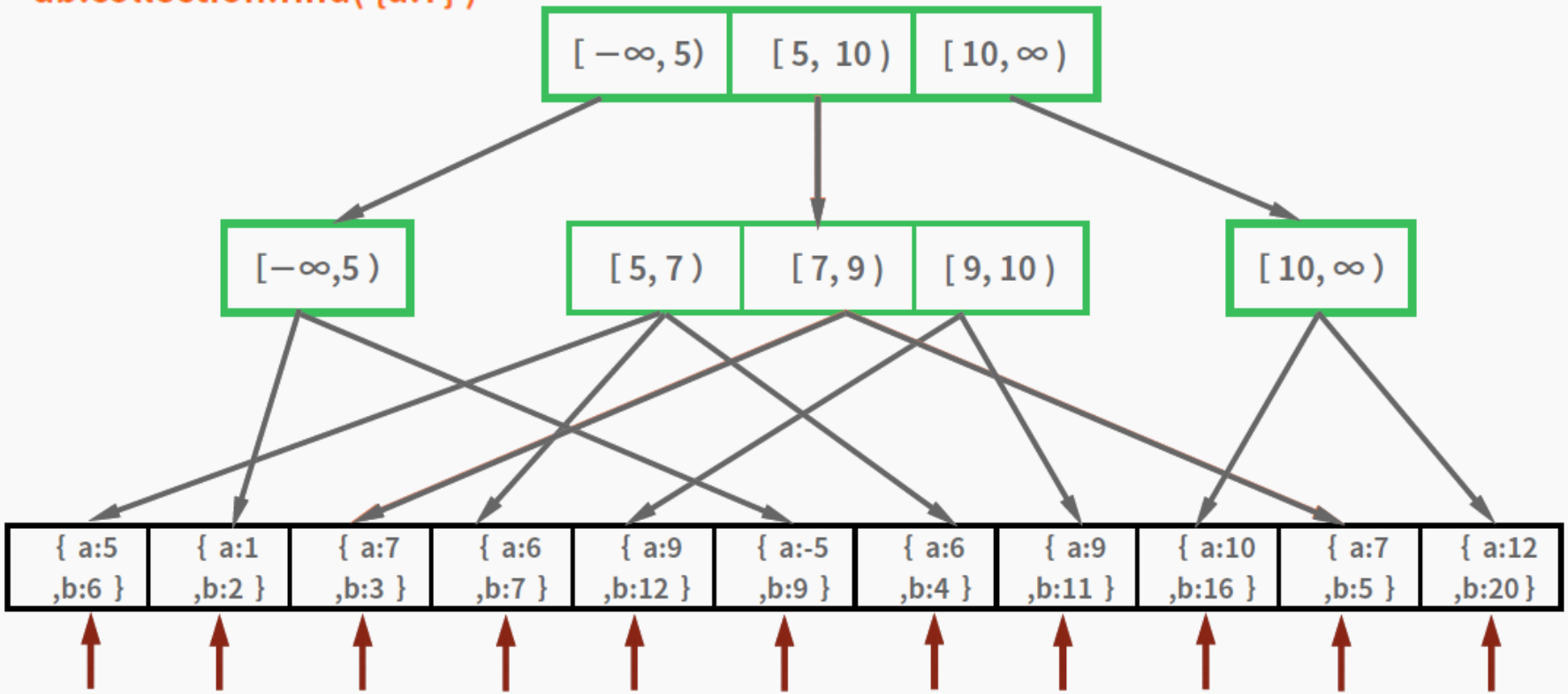
MongoDB索引不仅可以提高文档的查询速度，而且对于排序操作还可以节省内存资源的。

默认总是为_id创建索引。db.collection.getIndexes()可以查看当前数据库中创建的所有索引。

MongoDB提供了多样性的索引支持，通过为索引设置特定的属性，可以实现更多复杂的功能。

MongoDB索引的类型-B-树

db.collection.find({a:7})



MongoDB索引的类型

MongoDB提供了多种类型的索引，功能十分强大，其类型如下：

类型	说明	作用
Single Filed	单字段索引	在普通字段、子文挡以及子文档的某个字段上建立的索引
Compound Index	复合索引	同时在多个字段上建立的索引
Multikey Index	多键索引	对数组建立的索引
Geospatial Index	地理空间索引	对地理位置型数据建立的索引（支持球面和平面）
Text Index	全文索引	对每一个词建立索引，支持全文搜索
Hashed Index	哈希索引	索引中存储的是被索引键的哈希值

MongoDB索引的类型

Single Field Indexes：MongoDB可以在单个字段上建立索引，字段可以是普通字段、整个子文档以及子文档的某个字段。

例如：db.student.createIndex({“address”:1})

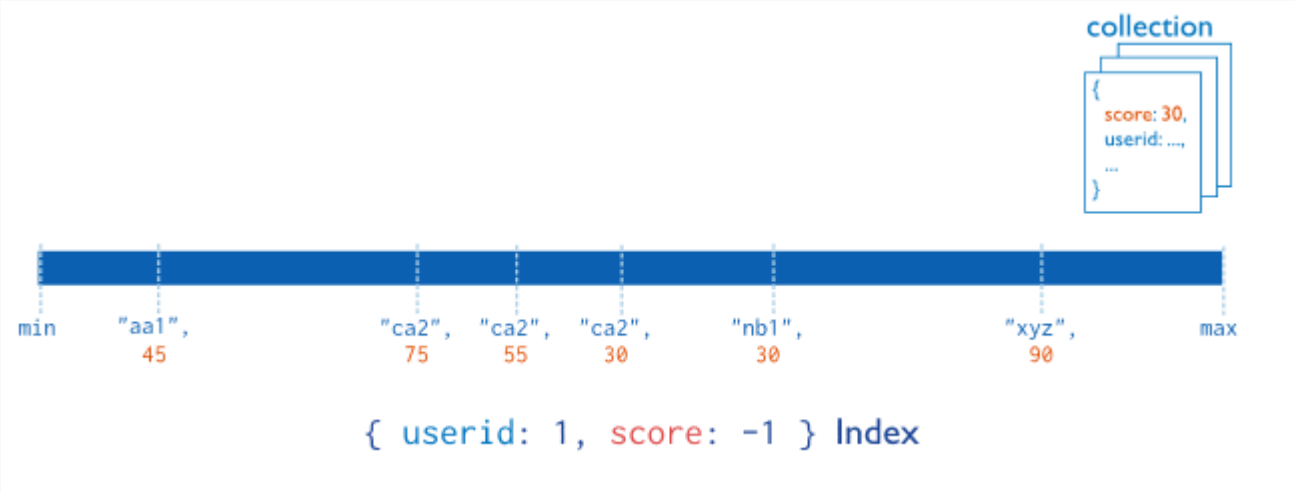
db.student.createIndex({“address.city”:1})

_id索引是系统默认创建的**单字段升序且具有唯一属性**的索引，每个集合的文档都会包含该字段，不能被删除，默认是ObjectId类型。

MongoDB索引的类型

Compound Indexes：复合索引是建立在多个字段上的索引，功能比单字段索引强大，但使用较复杂。不能超过32个键

db.collection.createIndex({ <field1>: <type>, <field2>: <type2>, ... })



MongoDB索引的类型

例如：db.student.createIndex({name:1,age:-1})

索引字段的排序方向 上面例子中，索引数据首先按照name升序排序，对于name相同的文档，按照age进行降序排序。

索引字段排序顺序 MongoDB在使用索引时，会自动进行优化，利用上面的索引可以支持如下的两个排序操作：

db.student.find().sort({name:1,age:-1})

db.student.find().sort({name:-1,age:1})

但是不支持： db.student.find().sort({name:1,age:1})

MongoDB索引的类型

对于复合索引，MongoDB支持前缀匹配（复合索引的子集）

例如：db.student.createIndex({name:1,age:1,address:1})

前缀索引包括：{name:1}与{name:1,age:1}

使用上面的索引可以在如下的字段上进行查询：

name

name和age

name、age和address

不支持在下面的字段上查询：age、address、age和address

MongoDB索引的类型

Multikey Indexes：多键索引是对数组列建立的索引，而不是对数组本身建立索引。MongoDB对数组中的每一个元素创建索引，从而使得数组列进行查询时，能够有效的提高查询速度。

db.coll.createIndex({ <field>: < 1 or -1 > })

例如：obj1={name:’bob’,score:[95,98]}

obj2={name:’joe’,score:[95,99]}

db.student.createIndex({score:1})

创建后的索引大致如下：

95=>[obj1,obj2]

98=>[obj1]

99=>[obj2]

MongoDB创建索引的时候检查，当前field是否是数组，如果是的话，MongoDB就会自动创建Multikey Index，不需要特殊的指定。

MongoDB索引的类型

关于多键索引需要注意的几个问题：

1、对于一个索引，只能对一个field创建多键索引，如果存在2个field的值都是数组，那么mongoDB是不允许对这两个field同时创建多键索引的。但是可以对2个field创建2个多键索引

例如：{a:[1,2],b:[1,2]} a和b都是数组类型，无法创建这样的索引：{a:1,b:1}

2、多键索引不支持哈希索引

3、多键索引不能对shard key做索引

MongoDB索引的类型

当数组元素是文档类型时，可以为文档的某个字段建立多键索引

{ “_id”:ObjectId(…),”title”:”Grocery Quality”,

“comments”:[

{ author_id:ObjectId(...),

date:Date(…),

text:”please expand the cheddar selection.”},

…]

}

db.feedback.createIndex({“commments.text”:1})

MongoDB索引的类型

哈希索引项中存储的是索引键的哈希值，哈希索引只支持等值查询，不支持范围查找。

例如：**db.student.createIndex({name:”hashed”})**

哈希索引主要用于分片的集合上，可以作为片键来使用，能够将数据比较均匀的分散存储在各个分片上。

注意：子文档建立哈希索引，不能在数组字段建立哈希索引

本章大纲

- MongoDB索引的类型
- **MongoDB索引的属性**
- MongoDB索引的管理

MongoDB索引的属性

MongoDB除了拥有众多的索引类型外，还有很多的属性，这些属性包括：唯一性、稀疏性以及超时删除的TTL属性。

使用这些属性创建的索引分别叫做：

- 唯一索引（Unique Index）
- 稀疏索引（Spare Index）
- TTL索引 （Time-To-Live Index）
- 部分索引 （Partial Indexes）
- 忽略大小写的索引（Case Insensitive Indexes）

https://docs.mongodb.com/manual/reference/method/db.collection.createIndex/#ensureindex-options

MongoDB索引的属性

Unique Indexes保证了集合中对于做了唯一索引的field不会存2条完全一样的数据。

创建唯一索引使用createIndex函数，将unique选项设置true。例如：

```
db.collection.createIndex( <key and index type specification>,
{ unique: true } )
```

MongoDB索引的属性

使用唯一索引时需要注意的几个问题：

- _id唯一索引是在创建集合时由数据库自动创建，与其它唯一索引不同的是，它不能被删除
- 为复合索引设置唯一属性时，只能保证组合索引字段的是唯一性，不能确保单个或索引字段子集的唯一性
- 不能为哈希索引指定唯一属性
- 不建议对数组实施唯一属性

MongoDB索引的属性

稀疏索引指的是只为索引字段存在的文档建立索引，即使索引字段的值为null，但不会为索引字段不存在的文档建立索引。

创建稀疏索引使用createIndex函数，将sparse选项设置为true。

例如：db.student.createIndex({name:1}, {sparse:true})

MongoDB索引的属性

Partial Indexes：sparse index的进化版

通过设置query条件指定范围建立索引

- equality expressions (i.e. field: value or using the $eq operator),
- $exists: true expression,
- $type expressions,
- $and operator at the top-level only
- $type expressions
- $gt, $gte, $lt, $lte expressions,

MongoDB索引的属性

注意事项：

1. 不能仅对限制条件做修改，而对partial index
2. 不能同时创建partial index和sparse index
3. 早期的版本不支持partial index
4. _id不能当做partial index
5. shard key不能当做partial index
6. 查询条件作为partial index的子集

```
db.restaurants.createIndex( { cuisine: 1, name: 1 },
{ partialFilterExpression: { rating: { $gt: 5 } } } )
```

MongoDB索引的属性

```
db.restaurants.createIndex( {
        { cuisine: 1, name: 1 },
        { partialFilterExpression: { rating: { $gt: 5 } }}
)
```

db.restaurants.find({ cuisine: “Italian”, rating: { $gte: 8 } })使用索引
db.restaurants.find({ cuisine: “Italian” })不使用索引
db.restaurants.find({ cuisine: “Italian”, rating: { $lt: 8 } }) 不使用索引

MongoDB索引的属性

TTL（Time-To-Live）索引可以为文档设置一个超时时间，当达到预设置的时间后，该文档会被数据库自动删除。这种类型的索引对于一个缓存问题（比如会话的保存）非常有用。

创建TTL索引使用createIndex函数，使用expireAfterSeconds选项来指定超时时间，单位为秒，仅对date类型或者数组中带有date的值有作用。例如：

```
db.eventlog.createIndex( { "lastModifiedDate": 1 },
{ expireAfterSeconds: 3600 } )
```

本章大纲

- MongoDB索引的类型
- MongoDB索引的属性
- MongoDB索引的管理

MongoDB索引的管理

- 索引的命名规则
- 索引的创建
- 索引的查看
- 索引的重建
- 索引的删除
- 索引的查询解释器

索引的命名规则

MongoDB索引默认命名规则为：keyname1_dir1_keyname2_dir2_.....，其中keynameX是索引的键，dirX是索引的方向，1表示升序，-1表示降序。

除了使用默认的命名方法外，在创建索引时，可以使用createIndex函数的第二个参数为索引指定一个名字。

例如：　db.student.createIndex({name:1,age:1},{name:'name_age'})

```
{
        "v" : 2,
        "key" : {
                "name" : 1,
                "age" : -1
        },
        "name" : "name_1_age_-1",
        "ns" : "student.a1"
}
```

创建索引

db.collection.createIndex(<key and index type specification>, <options>)

Parameter	Type	Description
`keys`	document	A document that contains the field and value pairs where the field is the index key and the value describes the type of index for that field. For an ascending index on a field, specify a value of `1`; for descending index, specify a value of `-1`. MongoDB supports several different index types including text, geospatial, and hashed indexes. See index types for more information. Starting in 3.6, you cannot specify `*` as the index name.
`options`	document	Optional. A document that contains a set of options that controls the creation of the index. See Options for details.

创建索引

db.collection.createIndex(<key and index type specification>, <options>)

- 主要有两个参数

1、keys　　文档类型，用来指明创建索引的字段以及排序方向，可以由多组键值对组成

例如：db.student.createIndex({name:1,age:-1})

2、options 用来设置索引的属性以及其它辅助选项，例如：为索引指定别名、在后台创建索引以及设置索引的版本号等

创建索引

Parameter	Type	Description
`background`	boolean	Optional. Builds the index in the background so the operation does *not* block other database activities. Specify `true` to build in the background. The default value is `false`.
`unique`	boolean	Optional. Creates a unique index so that the collection will not accept insertion or update of documents where the index key value matches an existing value in the index. Specify `true` to create a unique index. The default value is `false`. The option is *unavailable* for hashed indexes.
`name`	string	Optional. The name of the index. If unspecified, MongoDB generates an index name by concatenating the names of the indexed fields and the sort order. Whether user specified or MongoDB generated, index names including their full namespace (i.e. `database.collection`) cannot be longer than the `Index Name Limit`.

创建索引

- 返回值是一个文档类型，其包含的字段如下：

```
db.student.createIndex({name:1})
{
    “createdCollectionAutomatically”:true,      //集合是否自动创建
   “numIndexesBefore”:1,                        //创建之前索引个数
   “numIndexesAfter”:2,                         //创建之后索引个数
   “ok”:1                                       //表示创建索引成功
}
```

创建索引

创建索引需要注意的几个问题：

- 创建不同索引，可设置不同的options选项，具体可以参考官方文档
- 索引一旦创建就不能修改，如果需要修改只能将其删除，然后重建创建
- 如果重复多次创建一个索引，只有第一次会成功，其它会因重复创建而失败

查看索引

使用**db.collection.getIndexes()**可以查看集合拥有的索引，它的返回值是一个数组，会列出所有的索引，每个数组元素主要包含以下字段：

```
{  “v”:1,                                    //索引的版本号
    “key”:{“name”:1,”age”:-1},          //索引字段
    “name”:”name_1_age_-1”,             //索引的名字，可以指定
   “ns”:”soft.index_manage”,
//索引的所属的命名空间，数据库名_集合名
   “background”:true          //其他option选项
}
```

db.collection.getIndexKeys()　　查询建立索引的键名

重建索引

db.collection.reIndex() 重建当前集合的所有索引（很少用到）

先删除，再重建

注意：碎片化的集合是导致重建的原因

db.a1.totalIndexSize() 查询索引的大小

删除索引

db.collection.dropIndex(index)

- index 参数可以是索引的名字，也可以是创建索引时的keys文档参数

例如：db.student.dropIndex('name_age') //索引名

db.student.dropIndex({name:1,age-1}) //keys参数

- 返回值如下：

{ "nIndexesWas":2 //删除之前，集合拥有的索引数目

"ok":1 //删除索引成功

}

db.collection.dropIndex() 删除集合中所有的索引

查询解释器

又叫执行计划，提供开发者包括DBA一种直观的了解查询语句性能的方式**db.collection.find().explain()**

Parameter	Type	Description
verbose	string	Optional. Specifies the verbosity mode for the explain output. The mode affects the behavior of `explain()` and determines the amount of information to return. The possible modes are: `"queryPlanner"`, `"executionStats"`, and `"allPlansExecution"`. Default mode is `"queryPlanner"`.

queryPlanner：查询计划的选择器，首先进行查询分析，最终选择一个winningPlan，是explain返回的默认层面。

executionStats：为执行统计层面，返回winningPlan的统计结果

allPlansExecution：为返回所有执行计划的统计，包括rejectedPlan

explain函数

explain函数返回结果使用由多个阶段(Stage)组成的树形结构来表示查询计划（Query Plan）。

查询计划自下而上来执行，树的叶节点用来访问文档或者索引，内部节点处理下层节点返回的文档或索引数据，根节点是最后阶段，用于输出查询结果。

查询过程分阶段来执行，每个阶段代表一种操作类型，主要包括以下几种：

- COLLSCAN for a collection scan（全表扫描）
- IXSCAN for scanning index keys（索引扫描）
- FETCH for retrieving documents（根据索引去检索指定的文档）
- SHARD_MERGE for merging results from shards（分片数据进行合并）

hint函数

当查询使用的索引与希望使用的索引不一致时，可以使用hint函数强制MongoDB使用特定的索引。

db.集合.find().hint(index) index参数可以是索引的名字（字符串）或者创建索引时使用的keys参数。

例如： db.student.find().hint({age:1})

db.student.find().hint({“age_1”})