MongoDB - Indexing

Indexes support the efficient resolution of queries. Without indexes, MongoDB must scan every document of a collection to select those documents that match the query statement. This scan is highly inefficient and require MongoDB to process a large volume of data.

Indexes are special data structures, that store a small portion of the data set in an easy-to-traverse form. The index stores the value of a specific field or set of fields, ordered by the value of the field as specified in the index.

The createIndex() Method

To create an index, you need to use createIndex() method of MongoDB.

Syntax

The basic syntax of **createIndex()** method is as follows().

>db.COLLECTION\_NAME.createIndex({KEY:1})

Here key is the name of the field on which you want to create index and 1 is for ascending order. To create index in descending order you need to use -1.

Example

>db.mycol.createIndex({"title":1})

{

"createdCollectionAutomatically" : false,

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"ok" : 1

}

>

In **createIndex()** method you can pass multiple fields, to create index on multiple fields.

>db.mycol.createIndex({"title":1,"description":-1})

>

This method also accepts list of options (which are optional). Following is the list −

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| background | Boolean | Builds the index in the background so that building an index does not block other database activities. Specify true to build in the background. The default value is **false**. |
| unique | Boolean | Creates a unique index so that the collection will not accept insertion of documents where the index key or keys match an existing value in the index. Specify true to create a unique index. The default value is **false**. |
| name | string | The name of the index. If unspecified, MongoDB generates an index name by concatenating the names of the indexed fields and the sort order. |
| sparse | Boolean | If true, the index only references documents with the specified field. These indexes use less space but behave differently in some situations (particularly sorts). The default value is **false**. |
| expireAfterSeconds | integer | Specifies a value, in seconds, as a TTL to control how long MongoDB retains documents in this collection. |
| weights | document | The weight is a number ranging from 1 to 99,999 and denotes the significance of the field relative to the other indexed fields in terms of the score. |
| default\_language | string | For a text index, the language that determines the list of stop words and the rules for the stemmer and tokenizer. The default value is **English**. |
| language\_override | string | For a text index, specify the name of the field in the document that contains, the language to override the default language. The default value is language. |

The dropIndex() method

You can drop a particular index using the dropIndex() method of MongoDB.

Syntax

The basic syntax of DropIndex() method is as follows().

>db.COLLECTION\_NAME.dropIndex({KEY:1})

Here key is the name of the file on which you want to create index and 1 is for ascending order. To create index in descending order you need to use -1.

Example

> db.mycol.dropIndex({"title":1})

{

"ok" : 0,

"errmsg" : "can't find index with key: { title: 1.0 }",

"code" : 27,

"codeName" : "IndexNotFound"

}

The dropIndexes() method

This method deletes multiple (specified) indexes on a collection.

Syntax

The basic syntax of DropIndexes() method is as follows() −

>db.COLLECTION\_NAME.dropIndexes()

Example

Assume we have created 2 indexes in the named mycol collection as shown below −

> db.mycol.createIndex({"title":1,"description":-1})

Following example removes the above created indexes of mycol −

>db.mycol.dropIndexes({"title":1,"description":-1})

{ "nIndexesWas" : 2, "ok" : 1 }

>

The getIndexes() method

This method returns the description of all the indexes int the collection.

Syntax

Following is the basic syntax od the getIndexes() method −

db.COLLECTION\_NAME.getIndexes()

Example

Assume we have created 2 indexes in the named mycol collection as shown below −

> db.mycol.createIndex({"title":1,"description":-1})

Following example retrieves all the indexes in the collection mycol −

> db.mycol.getIndexes()

[

{

"v" : 2,

"key" : {

"\_id" : 1

},

"name" : "\_id\_",

"ns" : "test.mycol"

},

{

"v" : 2,

"key" : {

"title" : 1,

"description" : -1

},

"name" : "title\_1\_description\_-1",

"ns" : "test.mycol"

}

]

>