SNS COLLEGE OF TECHNOLOGY

COIMBATORE

AN AUTONOMOUS INSTITUTION

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE New Delhi & affiliated to the Anna University, Chennai

DEPARTMENT OF MCA

Course Name : 19CAT609 - DATA BASE MANAGEMENT SYSTEM

Class : I Year / II Semester

Unit V - COLUMN ORIENTED DATABASE

Topic III – Indexing and ordering datasets

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.

2

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}) >

The createIndex() Method

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.

4

The createIndex() Method

Parameter	Type	Description
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

•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" }
```

6

The dropIndexes() method

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})

8

The getIndexes() method

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"
}
] >
```

9

MongoDB - Indexing Limitations

Extra Overhead

Every index occupies some space as well as causes an overhead on each insert, update and delete. So if you rarely use your collection for read operations, it makes sense not to use indexes.

RAM Usage

Since indexes are stored in RAM, you should make sure that the total size of the index does not exceed the RAM limit. If the total size increases the RAM size, it will start deleting some indexes, causing performance loss.

Query Limitations

Indexing can't be used in queries which use −
Regular expressions or negation operators like $nin, $not, etc.
Arithmetic operators like $mod, etc.
$where clause

Hence, it is always advisable to check the index usage for your queries.

Index Key Limits 11
Starting from version 2.6, MongoDB will not create an index if the value of existing index field exceeds the index key limit.

Query Limitations

Inserting Documents Exceeding Index Key Limit

MongoDB will not insert any document into an indexed collection if the indexed field value of this document exceeds the index key limit. Same is the case with mongorestore and mongoimport utilities.

Maximum Ranges

12

A collection cannot have more than 64 indexes.
The length of the index name cannot be longer than 125 characters.
A compound index can have maximum 31 fields indexed.

MongoDB - Covered Queries

What is a Covered Query?

As per the official MongoDB documentation, a covered query is a query in which –

All the fields in the query are part of an index.

All the fields returned in the query are in the same index.

Since all the fields present in the query are part of an index, MongoDB matches the query conditions and returns the result using the same index without actually looking inside the documents. Since indexes are present in RAM, fetching data from indexes is much faster as compared to fetching data by scanning documents.

MongoDB - Covered Queries

Using Covered Queries

To test covered queries, consider the following document in the **users** collection

```
{
"_id": ObjectId("53402597d852426020000003"),
"contact": "987654321", "dob": "01-01-1991",
"gender": "M",
"name": "Tom Benzamin",
"user_name": "tombenzamin“
 }
```

14

MongoDB - Covered Queries

We will first create a compound index for the **users** collection on the fields **gender** and **user_name** using the following query

```
>db.users.createIndex({gender:1,user_name:1})
{
"createdCollectionAutomatically" : false,
 "numIndexesBefore" : 1,
"numIndexesAfter" : 2,
 "ok" : 1
}
```


Now, this index will cover the following query

```
>db.users.find({gender:"M"},
{user_name:1,_id:0})
{ "user_name" : "tombenzamin" }
```

MongoDB - Covered Queries

That is to say that for the above query, MongoDB would not go looking into database documents. Instead it would fetch the required data from indexed data which is very fast.

Since our index does not include **_id** field, we have explicitly excluded it from result set of our query, as MongoDB by default returns _id field in every query. So the following query would not have been covered inside the index created above

```
>db.users.find({gender:"M"},
{user_name:1})
{ "_id" : ObjectId("53402597d852426020000003"),
"user_name" : "tombenzamin" }
```

16

Lastly, remember that an index cannot cover a query if –

Any of the indexed fields is an array

Any of the indexed fields is a subdocument

Reference

1. https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm
2. https://docs.mongodb.com/manual/indexes/
3. https://www.tutorialspoint.com/mongodb/mongodb_covered_queries.htm

THANK YOU