No SQL: MongoDB

MEAN/MERN STACK

Learning Objectives:

- Learn MongoDB
- Create Database
- Create Collection
- MongoDB Insert
- MongoDB Find
- MongoDB Query
- MongoDB Delete
- MongoDB Drop Collection
- MongoDB Update

Introduction to MongoDB

Introduction

1. MongoDB is a No SQL database. It is an open-source, cross-platform, document-oriented database written in C++.
2. It is an open-source document database that provides high performance, high availability, and automatic scaling.
3. In simple words, you can say that - Mongo DB is a document-oriented database. It is an open source product, developed and supported by a company named 10gen.
4. MongoDB is a cross-platform, document-oriented database that provides, high performance, high availability, and easy scalability.
5. MongoDB works on concept of collection and document.

Installation:

- Download a free MongoDB database at https://www.mongodb.com.
- Or get started right away with a MongoDB cloud service at https://www.mongodb.com/cloud/atlas.

Database

Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

Collection

Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

Document

A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

Introduction to MongoDB

The following table shows the relationship of RDBMS terminology with MongoDB.

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by MongoDB itself)

Introduction to MongoDB

Sample Document

Following example shows the document structure of a blog site, which is simply a comma separated key value pair.

```
{
  _id: ObjectId(7df78ad8902c)
  title: 'MongoDB Overview',
  description: 'MongoDB is no sql database',
  url: 'http://www.something.com',
  tags: ['mongodb', 'database', 'NoSQL'],
  likes: 100,
  comments: [
     {
        user:'user1',
        message: 'My first comment',
        dateCreated: new Date(2011,1,20,2,15),
        like: 0
     },
     {
        user:'user2',
        message: 'My second comments',
        dateCreated: new Date(2011,1,25,7,45),
        like: 5
     }
  ]
}
```

Introduction to MongoDB

Advantages of MongoDB over RDBMS

- Schema less – MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Structure of a single object is clear.
- No complex joins.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- Ease of scale-out – MongoDB is easy to scale.
- Conversion/mapping of application objects to database objects not needed.
- Uses internal memory for storing the (windowed) working set, enabling faster access of data.

Why Use MongoDB?

- Document Oriented Storage – Data is stored in the form of JSON style documents.
- Index on any attribute
- Replication and high availability
- Rich queries
- Fast in-place updates
- Professional support by MongoDB

Introduction to MongoDB

The use Command

- MongoDB use **DATABASE_NAME** is used to create database. The command will create a new database if it doesn't exist, otherwise it will return the existing database.

Syntax

- Basic syntax of use DATABASE statement is as follows –

```
use DATABASE_NAME
```

Example

If you want to use a database with name <**mydb**>, then use **DATABASE** statement would be as follows –

```
>use mydb
switched to db mydb
```

To check your currently selected database, use the command db

```
>db
mydb
```

If you want to check your databases list, use the command show dbs.

Introduction to MongoDB

The dropDatabase() Method

MongoDB **db.dropDatabase()** command is used to drop an existing database.

Syntax

Basic syntax of **dropDatabase()** command is as follows

```
db.dropDatabase()
```

This will delete the selected database. If you have not selected any database, then it will delete default 'test' database.

Example

First, check the list of available databases by using the command, **show dbs**.

```
>show dbs
local      0.78125GB
mydb       0.23012GB
test       0.23012GB
>
```

If you want to delete new database <**mydb**>, then **dropDatabase**() command would be as follows –

```
>use mydb
switched to db mydb
>db.dropDatabase()
>{ "dropped" : "mydb", "ok" : 1 }
>
```

Introduction to MongoDB

The createCollection() Method

MongoDB **db.createCollection**(name, options) is used to create collection.

Syntax

Basic syntax of **createCollection**() command is as follows:-

```
db.createCollection(name, options)
```

In the command, name is name of collection to be created. Options is a document and is used to specify configuration of collection.

Parameter	Type	Description
Name	String	Name of the collection to be created
Options	Document	(Optional) Specify options about memory size and indexing

Introduction to MongoDB

Options parameter is optional, so you need to specify only the name of the collection. Following is the list of options you can use –

Field	Type	Description
capped	Boolean	(Optional) If true, enables a capped collection. Capped collection is a fixed size collection that automatically overwrites its oldest entries when it reaches its maximum size. If you specify true, you need to specify size parameter also.
autoIndexId	Boolean	(Optional) If true, automatically create index on _id field.s Default value is false.
size	number	(Optional) Specifies a maximum size in bytes for a capped collection. If capped is true, then you need to specify this field also.
max	number	(Optional) Specifies the maximum number of documents allowed in the capped collection.

Introduction to MongoDB

While inserting the document, MongoDB first checks size field of capped collection, then it checks max field.

Examples

Basic syntax of createCollection() method without options is as follows:-

```
>use test
switched to db test
>db.createCollection("mycollection")
{ "ok" : 1 }
>
```

You can check the created collection by using the command show collections.

```
>show collections
mycollection
system.indexes
```

The following example shows the syntax of createCollection() method with few important options –

```
> db.createCollection("mycol", { capped : true, autoIndexID : true, size : 6142800, max : 10000 } ){
  "ok" : 0,
  "errmsg" : "BSON field 'create.autoIndexID' is an unknown field.",
  "code" : 40415,
  "codeName" : "Location40415"
  }
  >
```

Introduction to MongoDB

In MongoDB, you don't need to create collection. MongoDB creates collection automatically, when you insert some document.

```
>db.new_collection.insert({"name" : "Hello"}),
WriteResult({ "nInserted" : 1 })
```

The drop() Method

MongoDB's db.collection.drop() is used to drop a collection from the database.

Syntax

Basic syntax of drop() command is as follows:

```
db.COLLECTION_NAME.drop()
```

Example

First, check the available collections into your database mydb.

```
>use mydb
switched to db mydb
>show collections
mycol
mycollection
>
```

Introduction to MongoDB

Now drop the collection with the name mycollection.

```
>db.mycollection.drop()
true
```

Again check the list of collections into database.

```
>show collections
mycol
```

drop() method will return true, if the selected collection is dropped successfully, otherwise it will return false.

The insert() Method

To insert data into MongoDB collection, you need to use MongoDB's insert() or save() method.

Syntax

The basic syntax of insert() command is as follows:

```
>db.COLLECTION_NAME.insert(document)
```

Introduction to MongoDB

Query Documents:

The find() Method

To query data from MongoDB collection, you need to use MongoDB's find() method.

Syntax

The basic syntax of find() method is as follows:

```
>db.COLLECTION_NAME.find()
```

find() method will display all the documents in a non-structured way.

For more details on Query Documents visit MongoDB official documentation.

MongoDB Update() Method

The update() method deletes the values in the existing document.

Syntax

The basic syntax of update() method is as follows:

```
>db.COLLECTION_NAME.update(SELECTION_CRITERIA, UPDATED_DATA)
```

Introduction to MongoDB

MongoDB Save() Method

The save() method replaces the existing document with the new document passed in the save() method.

Syntax

The basic syntax of MongoDB save() method is shown below –

```
>db.COLLECTION_NAME.save({_id:ObjectId(),NEW_DATA})
```

The remove() Method

MongoDB's remove() method is used to remove a document from the collection. remove() method accepts two parameters. One is deletion criteria and second is justOne flag.

- deletion criteria – (Optional) deletion criteria according to documents will be removed.
- justOne – (Optional) if set to true or 1, then remove only one document.

Syntax

Basic syntax of remove() method is as follows

```
db.COLLECTION_NAME.remove(DELLETION_CRITTERIA)
```

Introduction to MongoDB

Remove Only One

If there are multiple records and you want to delete only the first record, then set justOne parameter in remove() method.

```
>db.COLLECTION_NAME.remove(DELETION_CRITERIA,1)
```

Remove All Documents

If you don't specify deletion criteria, then MongoDB will delete whole documents from the collection. This is equivalent of SQL's truncate command.

```
> db.mycol.remove({})
WriteResult({ "nRemoved" : 2 })
> db.mycol.find()
>
```

Questions ?