MongoDB CRUD

Estimated time needed: **30** minutes

Objectives

After completing this lab, you will be able to:

- Create documents in MongoDB with the insert method
- Read documents by listing them, counting them and matching them to a query
- Update and delete documents in MongoDB based on specific criteria

About Skills Network Cloud IDE

Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project related labs. Theia is an open source IDE (Integrated Development Environment), that can be run on desktop or on the cloud. to complete this lab, we will be using the Cloud IDE based on Theia and MongoDB provided by Skills Network.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session, to avoid losing your data.

Exercise 1 - Getting the environment ready

Open the MongoDB database page by clicking the button below:

Open MongoDB Page in IDE

On that page, click the `Create` button to create a MongoDB database.

After creation has finished, click the `MongoDB CLI` button (below the `Create` button from the previous step) to connect to the database using the mongosh CLI.

1. Problem:

   *Select the **training** database.*

   ▶ Click here for Hint
   ▶ Click here for Solution

2. Problem:

*Create a collection named **languages**.*

- ▶ Click here for Hint
- ▶ Click here for Solution

Exercise 2 - Insert documents

Let us insert five documents into the collection **languages**

On the mongo client run the below commands.

```
db.languages.insert({"name":"java","type":"object oriented"})
db.languages.insert({"name":"python","type":"general purpose"})
db.languages.insert({"name":"scala","type":"functional"})
db.languages.insert({"name":"c","type":"procedural"})
db.languages.insert({"name":"c++","type":"object oriented"})
```

Exercise 3 - Read documents

Let us try out different ways of querying documents.

Find the count of documents.

```
db.languages.countDocuments()
```

List the first document in the collection.

```
db.languages.findOne()
```

List all documents in the collection.

```
db.languages.find()
```

List first 3 documents in the collection.

```
db.languages.find().limit(3)
```

Query for “python” language.

```
db.languages.find({"name":"python"})
```

Query for “object oriented” languages.

```
db.languages.find({"type":"object oriented"})
```

List only specific fields.

Using a projection document you can specify what fields we wish to see or skip in the output.

This command lists all the documents with only name field in the output.

```
db.languages.find({},{"name":1})
```

This command lists all the documents without the name field in the output.

```
db.languages.find({},{"name":0})
```

This command lists all the “object oriented” languages with only “name” field in the output.

```
db.languages.find({"type":"object oriented"},{"name":1})
```

Exercise 4 - Update documents

Update documents based on a criteria.

Add a field to all the documents.

The 'updateMany' command is used to update documents in a mongodb collection, and it has the following generic syntax.

```
db.collection.updateMany({what documents to find},{$set:{what fields to set}})
```

Here we are adding a field **description** with value **programming language** to all the documents.

```
db.languages.updateMany({},{$set:{"description":"programming language"}})
```

Set the creater for python language.

```
db.languages.updateMany({"name":"python"},{$set:{"creator":"Guido van Rossum"}})
```

Set a field named **compiled** with a value **true** for all the **object oriented** languages.

```
db.languages.updateMany({"type":"object oriented"},{$set:{"compiled":true}})
```

Exercise 5 - Delete documents

Delete documents based on a criteria.

Delete the **scala** language document.

```
db.languages.remove({"name":"scala"})
```

Delete the **object oriented** languages.

```
db.languages.remove({"type":"object oriented"})
```

Delete all the documents in a collection.

```
db.languages.remove({})
```

Practice exercises

Run the below code on mongo console. It will insert 5 documents, which will serve as sample data for the next steps.

```
use training
db.languages.insert({"name":"java","type":"object oriented"})
db.languages.insert({"name":"python","type":"general purpose"})
db.languages.insert({"name":"scala","type":"functional"})
db.languages.insert({"name":"c","type":"procedural"})
db.languages.insert({"name":"c++","type":"object oriented"})
```

1. Problem:

   Insert an entry for 'Haskell' programming language which is of type 'functional .

▶ Click here for Hint
▶ Click here for Solution

2. Problem:

   Query for all functional languages.

▶ Click here for Hint
▶ Click here for Solution

3. Problem:

   Add 'Bjarne Stroustrup' as creator for c++.

▶ Click here for Hint
▶ Click here for Solution

4. Problem:

   Delete all functional programming languages.

▶ Click here for Hint
▶ Click here for Solution

5. Problem:

   Disconnect from the mongodb server.

▶ Click here for Hint
▶ Click here for Solution

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja