# Lab 7 (Mongo - Create/Delete DB/Collection/Documents)

# DBS311 D - Lab 7 *– In-Lab submission only, unless you have a permission from me to submit by e-mail. Your Word or TXT Document must have both the CODE and the OUTPUTS. You may also submit your Lab Printout to our Reception Office, Room A3058 by due date.*

# Due by Thursday, March 28th by 4pm

## Objective

In this Lab, you learn to create and remove the following from MongoDB

* Databases
* Collections
* Documents

## Getting Started

Open your Windows command prompt and go the following directory where MongoDB is installed:

* cd C:\Program Files\MongoDB\Server\**5.0\bin**

To run MongoDB, execute ***mongod***

* mongod

When MongoDB starts successfully, open another Windows command prompt and go the same *bin* directory:

* cd C:\Program Files\MongoDB\Server\**5.0\bin**

and execute ***mongo***

* mongo

This opens the command line shell where you can work with MongoDB.

To test you solutions before you submit, test them individually and make sure they work correctly.

## Tasks

1. In this question you create a new database named *toronto* and a collection *student*. We store student data in this collection.

* use toronto

This command makes “toronto” your current database. However, the database will not be created until you insert a document into this database.

db.collection\_name.insertOne({doc})

Create (Insert) a new document into your collection *student* with the following data:

first\_name: Sarah  
last\_name: Stone  
email: [s\_stone@email.com](mailto:s_stone@email.com)  
city: Toronto  
status: full-time  
gpa: 3.4  
program: CPA

|  |
| --- |
| db.student.insertOne({  first\_name: "Sarah",  last\_name: "Stone",  email: "s\_stone@email.com",  city: "Toronto",  status: "full-time",  gpa: 3.4,  program: "CPA"  }) |

1. Write a command to check if the document has been created successfully.

You use *find()* method to search and fetch documents.

|  |
| --- |
| > db.student.find()  { "\_id" : ObjectId("6601db1be504633cf7b90053"), "first\_name" : "Sarah", "last\_name" : "Stone", "email" : "s\_stone@email.com", "city" : "Toronto", "status" : "full-time", "gpa" : 3.4, "program" : "CPA" } |

How many fields are in your document? \_\_8\_\_\_

Is there any new field added to your document? \_\_\_yes\_\_\_

If yes, what is the name of the field? \_\_\_id\_\_\_

See the following example (where we have 4 students already entered).

* db.student.find()

|  |
| --- |
| { "\_id" : 3, "name" : "Bao Ziglar", "scores" : [ { "score" : 71.64343899778332, "type" : "exam" }, { "score" : 24.80221293650313, "type" : "quiz" }, { "score" : 42.26147058804812, "type" : "homework" } ] }  { "\_id" : 6, "name" : "Jenette Flanders", "scores" : [ { "score" : 37.32285459166097, "type" : "exam" }, { "score" : 28.32634976913737, "type" : "quiz" }, { "score" : 81.57115318686338, "type" : "homework" } ] }  { "\_id" : 0, "name" : "aimee Zank", "scores" : [ { "score" : 1.463179736705023, "type" : "exam" }, { "score" : 11.78273309957772, "type" : "quiz" }, { "score" : 35.8740349954354, "type" : "homework" } ] }  { "\_id" : 8, "name" : "Daphne Zheng", "scores" : [ { "score" : 22.13583712862635, "type" : "exam" }, { "score" : 14.63969941335069, "type" : "quiz" }, { "score" : 75.94123677556644, "type" : "homework" } ] } |

To see the result in *JSON* format, you can run the following statement:

* db.student.find().forEach(printjson)

|  |
| --- |
| {  "\_id" : 3,  "name" : "Bao Ziglar",  "scores" : [  {  "score" : 71.64343899778332,  "type" : "exam"  },  {  "score" : 24.80221293650313,  "type" : "quiz"  },  {  "score" : 42.26147058804812,  "type" : "homework"  }  ]  }  {  "\_id" : 6,  "name" : "Jenette Flanders",  "scores" : [  {  "score" : 37.32285459166097,  "type" : "exam"  },  {  } etc … |

1. Write a command to remove the document that you created in Question 1. (We have only one document at this time, but when removing documents make sure you will not remove some other documents if not needed. So, make sure your command will remove “Sarah Stone” from your collection. For now, we assume that we do not have duplicate names in our database.)

**Note:** To avoid making mistakes, you can first write a find command with the proper criteria to see if the required document is fetched. Then, you can use the same criteria in your delete statement. (Write the statement to remove “Sarah Stone” from the database in the box below.)

|  |
| --- |
| db.student.find({ first\_name: "Sarah", last\_name: "Stone" })  db.student.deleteOne({ first\_name: "Sarah", last\_name: "Stone" }) |

What is the message as a result of your delete statement? Copy the message in the following box:

|  |
| --- |
| { "acknowledged" : true, "deletedCount" : 1 }\_ |

To see if the document is removed successfully, write a search statement to see if the document exists. (We look for one document not all).

|  |
| --- |
| > db.student.findOne({ first\_name: "Sarah", last\_name: "Stone" })  null |

1. We want to add some students to our collection, but this time, we define the value for the *\_id* field. (If the \_id is not defined in your document, it will be added automatically.)

\_id: 1001

first\_name: Sarah  
last\_name: Stone  
email: [s\_stone@email.com](mailto:s_stone@email.com)  
city: Toronto  
status: full-time  
gpa: 3.4  
program: CPA

\_id: 1002

first\_name: Jack  
last\_name: Adam  
email: [j\_adam@email.com](mailto:j_adam@email.com)  
city: North York  
status: part-time  
gpa: 3.6  
program: CPA

To add these students, we want to store these documents into a variable first.

Define a variable named *starray* and add these two document to the variable. (You are storing more than one document so you need to define an array.

|  |
| --- |
| starray = [  {  \_id: 1001,  first\_name: "Sarah",  last\_name: "Stone",  email: "s\_stone@email.com",  city: "Toronto",  status: "full-time",  gpa: 3.4,  program: "CPA"  },  {  \_id: 1002,  first\_name: "Jack",  last\_name: "Adam",  email: "j\_adam@email.com",  city: "North York",  status: "part-time",  gpa: 3.6,  program: "CPA"  }  ]; |

Now, use the *starray* array to insert the documents to your collection *student*. Write your insert statement in the box bellow.

|  |
| --- |
| db.student.insertMany(starray) |

What message is displayed after you execute the insert statement. Copy the message in the following box:

|  |
| --- |
| > db.student.insertMany(starray)  { "acknowledged" : true, "insertedIds" : [ 1001, 1002 ] } |

Write a statement that shows all documents inserted in your collection *student*:

|  |
| --- |
| > db.student.find()  { "\_id" : 1001, "first\_name" : "Sarah", "last\_name" : "Stone", "email" : "s\_stone@email.com", "city" : "Toronto", "status" : "full-time", "gpa" : 3.4, "program" : "CPA" }  { "\_id" : 1002, "first\_name" : "Jack", "last\_name" : "Adam", "email" : "j\_adam@email.com", "city" : "North York", "status" : "part-time", "gpa" : 3.6, "program" : "CPA" } |

1. Write a statement to remove all documents in the *student* collection.

|  |
| --- |
| > db.student.deleteMany({})  { "acknowledged" : true, "deletedCount" : 2 } |

1. Write a statement to drop the database *toronto*.

Before dropping a database, make sure your current database is the one you want to delete. For this question, we want to delete (drop) the *toronto* database.

You can write the *use* statement before removing the database to make sure your current database is *toronto*

> use toronto

Or, you can write the *db* or db.getName() statement to see the name of your current database: > db OR > db.getName()

If your current database is not *toronto*, write the use statement to switch to *toronto* and then delete the database.

|  |
| --- |
| > use Toronto  switched to db toronto  > db.dropDatabase() |

What message is displayed after you execute the drop statement? Copy the message in the box below:

|  |
| --- |
| > db.dropDatabase()  { "ok" : 1 } |