Lab 9 – Week 11

(MongoDB – UPDATE)

Objective

In this lab, students learn how to update documents in a MongoDB database.

update(): This method updates one document by default. If you want to update all documents that match the criteria using this method, you need the option {multi:true}.

update(<filter>,<update>,<option>)

The *filter* parameter specifies the criteria. For instance:

{"_id"= 0}

{} for updating all documents

The *update* parameter specifies the changes that will be applied to a document.

updateOne(): This method updates only the first document that matches the criteria.

updateOne(<filter>,<update>)

updateMany(): This method updates all documents that match the criteria.

updateMany(<filter>,<update>)

Getting Started

In this lab, you will use students.json dataset. Download students.json from Blackboard and store it in a folder named dataset.

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

- cd C:\Program Files\MongoDB\Server\4.2**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\4.2**bin**

and execute ***mongo***

- mongo

Or you execute a batch file to start up MongoDB.

You will import students.json to the *college* database. To import data, go to the *bin* directory:

- cd C:\Program Files\MongoDB\Server\4.2**bin**

Execute the following command:

- mongoimport --db college --collection students --file ..\dataset\students.json

To import the *json* file, provide the full path to the students.json. After executing the command, the data is imported to the *college* database. To make sure data is imported successfully, go to the MongoDB shell and execute the following command to see the imported documents:

- show dbs

You should see the database *college* added to the list of your databases. To see the documents inside the database:

- use college
- db.students.find().forEach(printjson)

or

- db.students.find().pretty()

Submission

You submit this file with answers (in the provided space). Name the file L10_ID#_LASTNAME.docx".

Tasks

1. Write an update statement to add new fields *program* and *term* to all documents in the *students* collection and set them to values "*CPA*" and *1*.

```
db.getCollection('students').update({},{$set : {"program" : "CPA", "term" : 1}},{multi : true})
```

2. Write an update statement to modify the value of the *program* field to "*BTM"* for all documents in the *students* collection.

```
db.getCollection('students').update({},{$set : {"program" : "BTM"}},{multi : true})
```

3. Write an update statement to modify the value of the program field to "*CPA"* for the student named *Jonie Raby*.
   Before executing an update statement or a delete statement, you can use the *find()* method with the update or delete criteria, to see how many documents will be affected.
   Write the update statement in the box below.

```
db.getCollection('students').find({"name" : "Jonie Raby"})

db.getCollection('students').update({"name" : "Jonie Raby"},{$set : {"program" : "CPA"}}
```

   How many documents are there with the value *Jonie Raby* for the *name* field? one
   How many documents were updated? one

4. Write a query to show only the *program* field for the document that the value of the filed *name* is *Jonie Raby*.

```
db.getCollection('students').find({"name" : "Jonie Raby"},{program : 1, _id : 0})
```

5. Write an update statement to increase the value of the *term* field by 2 for documents with *_id* 20, 22, and 24.

```
db.getCollection('students').update({"_id" : {$in : [20,22,24]}},{$inc : {"term" : 2}},{multi : true})
```

6. Write an update statement to remove the *term* field from documents that the value of the *term* filed is 3.

```
db.getCollection('students').update({"term" : 3},{$unset : {"term" : ""}},{multi : true}
```