CIND-110
Data Organization for Data Analysts

Lab Manual Module 11
NoSQL Database

Lead Instructor: Dr. Tamer Abdou

Contents

Objectives

To introduce MongoDB a document based NoSQL database and perform basic CRUD NoSQL operations on documents.

1. Mongo Installation

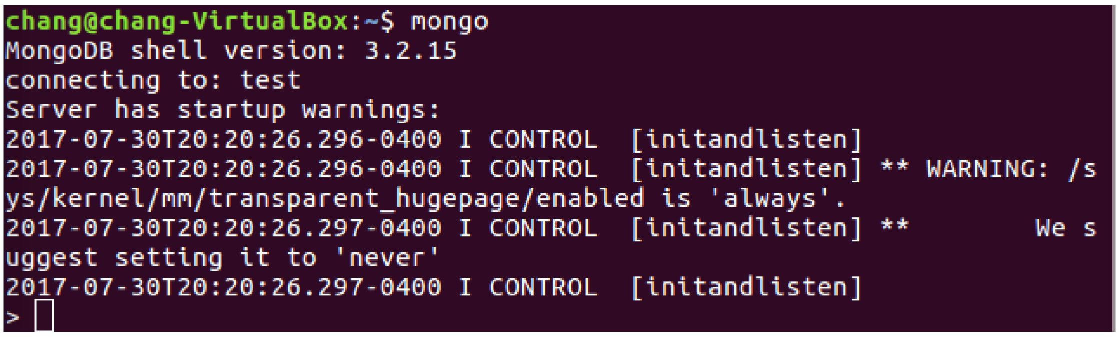
By now, mongo should be installed on your machines. If mongo is not installed, you would need to run the commands below.

sudo apt-key adv –keyserver hkp://keyserver.ubuntu.com:80 –recv EA312927
echo "deb http://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.2.list
sudo apt-get update
sudo apt-get install -y mongodb-org
sudo service mongod start

Now you have to launch Mongo shell using the below command:

Launching Mongo Shell

```
mongo
```

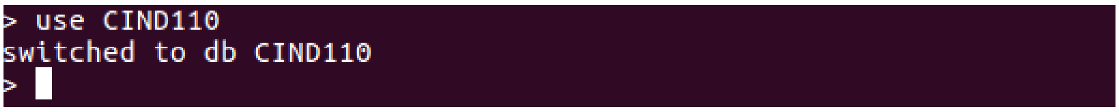


```
chang@chang-VirtualBox:~$ mongo
MongoDB shell version: 3.2.15
connecting to: test
Server has startup warnings:
2017-07-30T20:20:26.296-0400 I CONTROL  [initandlisten]
2017-07-30T20:20:26.296-0400 I CONTROL  [initandlisten] ** WARNING: /s
ys/kernel/mm/transparent_hugepage/enabled is 'always'.
2017-07-30T20:20:26.297-0400 I CONTROL  [initandlisten] **        We s
uggest setting it to 'never'
2017-07-30T20:20:26.297-0400 I CONTROL  [initandlisten]
> 
```

Now reate a new NoSQL database. Type the below command to create a database named ‘CIND110’:

Creating a Database

```
use CIND110
```


```
> use CIND110
switched to db CIND110
> 
```

2. Background

2.1 NoSQL

A NoSQL (originally referring to "non SQL", "non relational" or "not only SQL") database provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases.

2.2 Types and Exmaples of NoSQL Databases

Column	Accumulo, Cassandra, Druid, HBase, Vertica
Document	Apache CouchDB, Clusterpoint, Couchbase, DocumentDB, HyperDex, IBM,Domino, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
Key-Value	Aerospike, Couchbase, Dynamo, FairCom c-treeACE, FoundationDB, HyperDex, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, Berkeley DB
Graph	AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB, Virtuoso, Stardog
Multi-Model	Alchemy Database, ArangoDB, CortexDB, Couchbase, FoundationDB, MarkLogic, OrientDB

In this lab we will be going through the **Document** type using the **MongoDB** through the basic **Create, Read, Update, Delete** operations.

2.3 CRUD Operations on NoSQL

- Create Operations: db.collection.insert(), db.collection
- Read Operations: db.collection.find()
- Update Operations: db.collection.update()
- Delete Operations: db.collection.remove()

2.4 NoSQL and SQL Comparison

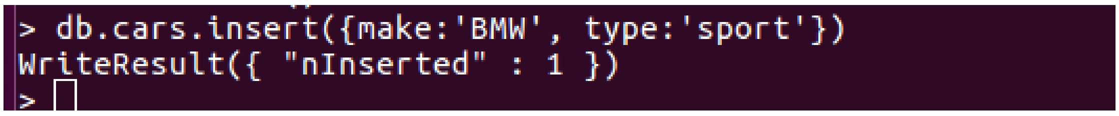
SQL	**NoSQL**
Tables	Collections
Rows	Documents
Insert into table values (‘x’)	db.collection.insert
Select from table where x >10	db.collection.find, db.collection.find(filter)
Update table where x = 10 set y = ‘cind110’	db.collection.update()
Delete from table where x = 10	db.collection.remove()

3. Create Operations in MongoDB

Type "**use CIND110**" to use the database we created previously.

Example to Show Insert Operation into NoSQL MongoDB using db.<collection>.insert()method

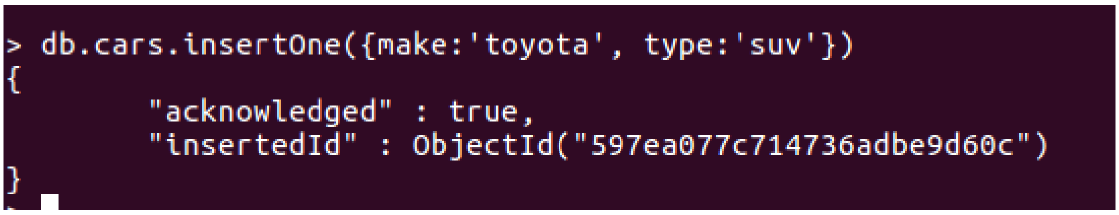
```
> db.cars.insert({make: 'BMW', type: 'sport'})
```



```
> db.cars.insert({make:'BMW', type:'sport'})
WriteResult({ "nInserted" : 1 })
> 
```

Example to insert one document into NoSQL MongoDB using db.<collection>.insertOne() method

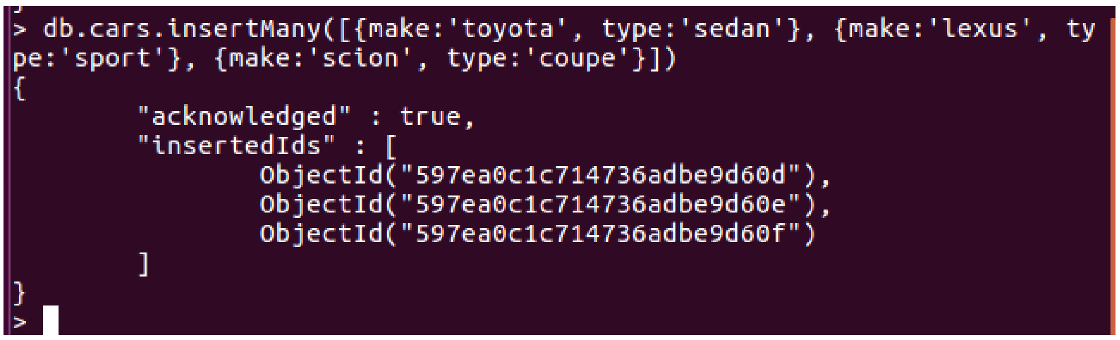
```
> db.cars.insertOne({make: 'toyota', type: 'suv'}
  )
```



```
> db.cars.insertOne({make:'toyota', type:'suv'})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("597ea077c714736adbe9d60c")
}
```

Example showing inseting many documents into NoSQL MongoDB using db.<collection>.insertMany() method

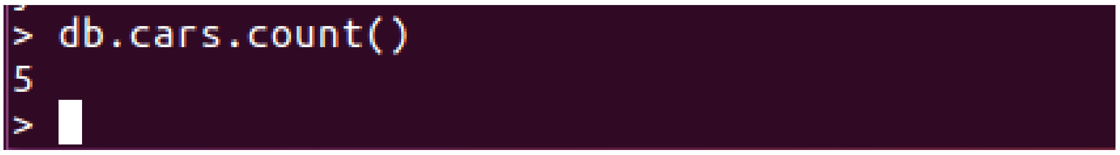
```
> db.cars.insertMany([{make: 'toyota', type: 'sedan'}, {make: 'lexus', type: 'sport'}, {make: 'scion', type: 'coupe'}]))
```



```
> db.cars.insertMany([{make:'toyota', type:'sedan'}, {make:'lexus', ty
pe:'sport'}, {make:'scion', type:'coupe'}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("597ea0c1c714736adbe9d60d"),
                ObjectId("597ea0c1c714736adbe9d60e"),
                ObjectId("597ea0c1c714736adbe9d60f")
        ]
}
>
```

Verifying the number of documents inserted into NoSQL MongoDB

```
> db.cars.count()
```

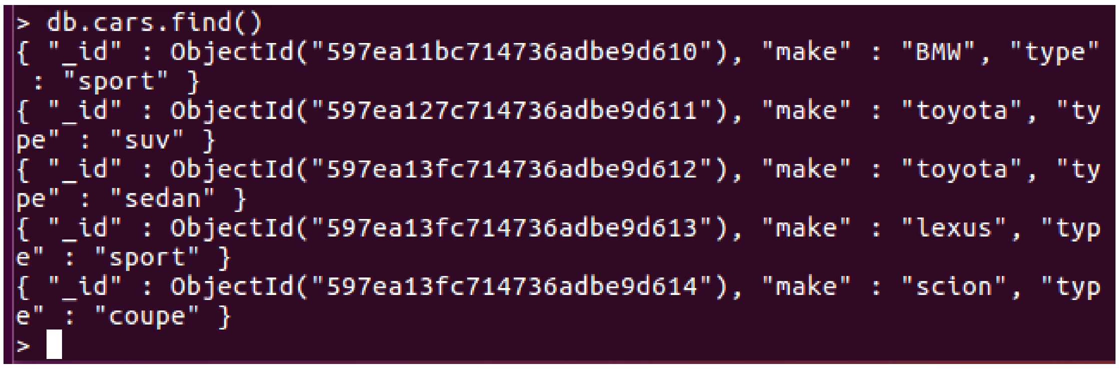


```
> db.cars.count()
5
>
```

4. Read Operations in MongoDB

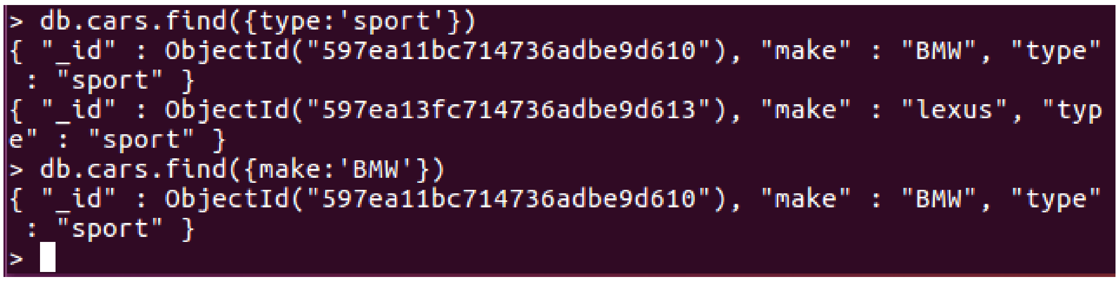
Example of Read operation in NoSQL MongoDB using db.<collection>.find() method

```
> db.cars.find()
```


```
> db.cars.find()
{ "_id" : ObjectId("597ea11bc714736adbe9d610"), "make" : "BMW", "type"
 : "sport" }
{ "_id" : ObjectId("597ea127c714736adbe9d611"), "make" : "toyota", "ty
pe" : "suv" }
{ "_id" : ObjectId("597ea13fc714736adbe9d612"), "make" : "toyota", "ty
pe" : "sedan" }
{ "_id" : ObjectId("597ea13fc714736adbe9d613"), "make" : "lexus", "typ
e" : "sport" }
{ "_id" : ObjectId("597ea13fc714736adbe9d614"), "make" : "scion", "typ
e" : "coupe" }
> 
```

Example of Read operation in NoSQL MongoDB using filters

```
> db.cars.find({type:'sport'})
> db.cars.find({make: 'BMW'})
```

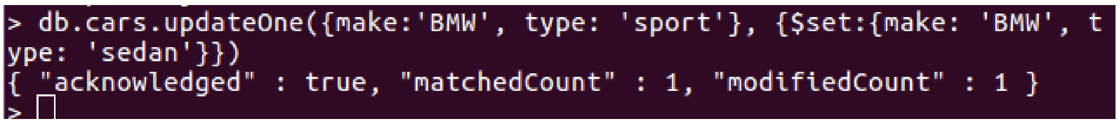

```
> db.cars.find({type:'sport'})
{ "_id" : ObjectId("597ea11bc714736adbe9d610"), "make" : "BMW", "type"
 : "sport" }
{ "_id" : ObjectId("597ea13fc714736adbe9d613"), "make" : "lexus", "typ
e" : "sport" }
> db.cars.find({make:'BMW'})
{ "_id" : ObjectId("597ea11bc714736adbe9d610"), "make" : "BMW", "type"
 : "sport" }
> 
```

5. Update Operations in MongoDB

In the following example we are update the type key value from sport to sedan for make key value of BMW:

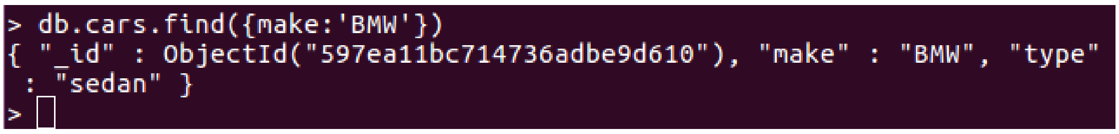
Example of Update operation in NoSQL MongoDB using db.<collection>.UpdateOne() method

```
> db.cars.updateOne({"make": "BMW", type: "sport"
  }, { $set: {"make": "BMW", type: "sedan"} })
```


```
> db.cars.updateOne({make:'BMW', type: 'sport'}, {$set:{make: 'BMW', t
ype: 'sedan'}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> 
```

Verifying the update process

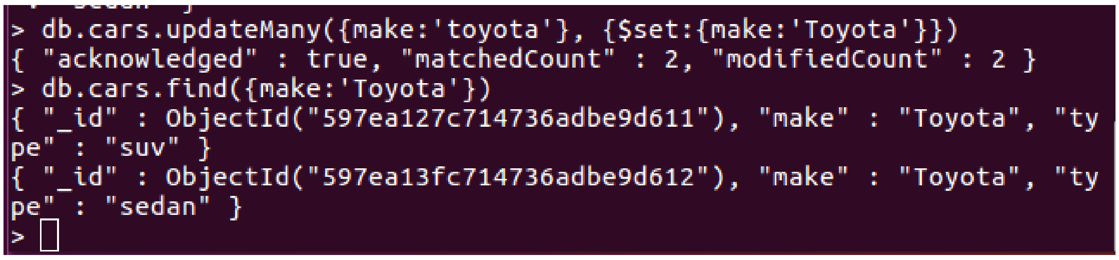
```
> db.cars.find({"make": "BMW"})
```



```
> db.cars.find({make:'BMW'})
{ "_id" : ObjectId("597ea11bc714736adbe9d610"), "make" : "BMW", "type"
 : "sedan" }
> 
```

Example of Update operation in NoSQL MongoDB using db.collection.updateMany() method

```
> db.cars.updateMany({"make": "toyota"}, { $set:
   {"make": "Toyota"} })
> db.cars.find({"make": "Toyota"})
```

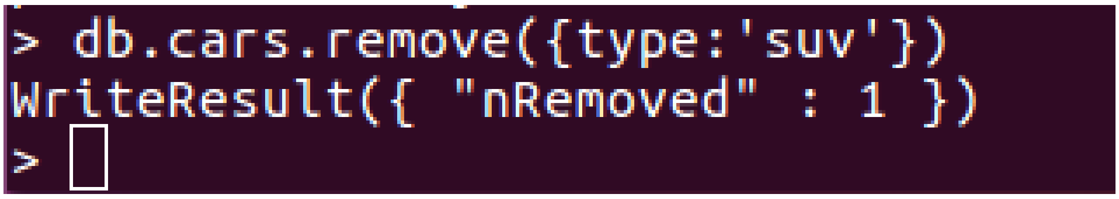


```
> db.cars.updateMany({make:'toyota'}, {$set:{make:'Toyota'}})
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
> db.cars.find({make:'Toyota'})
{ "_id" : ObjectId("597ea127c714736adbe9d611"), "make" : "Toyota", "ty
pe" : "suv" }
{ "_id" : ObjectId("597ea13fc714736adbe9d612"), "make" : "Toyota", "ty
pe" : "sedan" }
> 
```

6. Delete Operation in MongoDB

Example of remove operation in NoSQL MongoDB using db.<collection>.remove() method

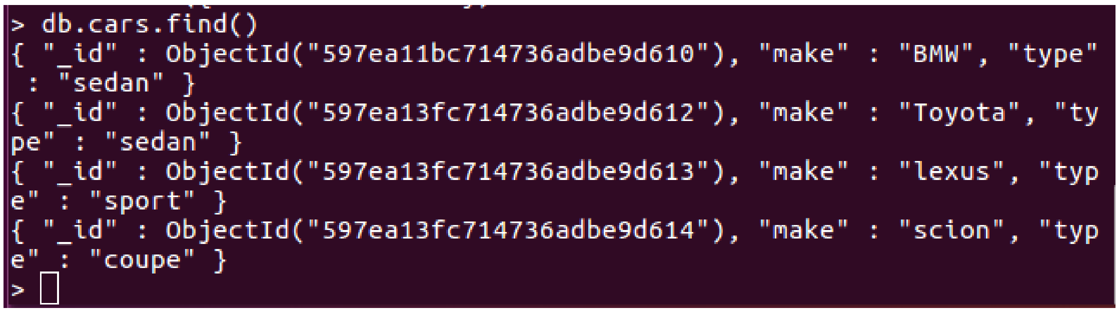
```
> db.cars.remove({"type": "suv"})
```



```
> db.cars.remove({type:'suv'})
WriteResult({ "nRemoved" : 1 })
> 
```

Verifying the number of documents after deletion

```
> db.cars.find()
```



```
> db.cars.find()
{ "_id" : ObjectId("597ea11bc714736adbe9d610"), "make" : "BMW", "type"
 : "sedan" }
{ "_id" : ObjectId("597ea13fc714736adbe9d612"), "make" : "Toyota", "ty
pe" : "sedan" }
{ "_id" : ObjectId("597ea13fc714736adbe9d613"), "make" : "lexus", "typ
e" : "sport" }
{ "_id" : ObjectId("597ea13fc714736adbe9d614"), "make" : "scion", "typ
e" : "coupe" }
> 
```