Intro to MongoDB II

Discussion material introduced with Unit 3

I435 / I535 / B669: Management, Access and Uses of Big Data
Fall 2016
School of Informatics and Computing
Indiana University

MongoDB Schema

```
{
        _id: ObjectID(7df78ad8902c),
        title: 'MongoDB sample',
        description: 'This is a sample document',
        tags: ['MongoDB', 'sample', 'NoSQL'],
        number: 1000,
        people: [
                {
                        user: ‘Yu’,
                        message: ‘Hi’
                },
                {
                        user: ‘Dimitar’,
                        message: ‘Hillo’
                }
        ]
}
```

How to use MongoDB

- The mongo Shell
  - The mongo shell is an interactive JavaScript interface to MongoDB.
- Online MongoDB shell
  - Not the same as an actual MongoDB installation, but a good starting point
  - https://www.tutorialspoint.com/mongodb_terminal_online.php

Mongo Shell

codingground SIMPLYEASYCODING | MONGODB TERMINAL ONLINE

Project File Edit View Shut Down Help

Project

Terminal

```
sh-4.3$ echo 'System is initializing....please wait'
System is initializing....please wait
sh-4.3$ /usr/bin/mongo
MongoDB shell version: 3.0.9
connecting to: test
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        http://docs.mongodb.org/
Questions? Try the support group
        http://groups.google.com/group/mongodb-user
Server has startup warnings:
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten]
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] ** WARNING: You are running on a NUMA machine.
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] **          We suggest launching mongod like this to avoid performance problems
:
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] **              numactl --interleave=all mongod [other options]
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten]
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] **        We suggest setting it to 'never'
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten]
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/defrag is 'always'.
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] **        We suggest setting it to 'never'
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten]
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten] ** WARNING: soft rlimits too low. rlimits set to 250 processes, 1024 files. Num
ber of processes should be at least 512 : 0.5 times number of files.
2016-09-15T04:59:21.947+0000 I CONTROL  [initandlisten]
>
```

Tutorials

C++

JAVA Programming

Python

Ruby

Swift

Scala

Create Database

- Create database
  - Command: use database_name

```
> show dbs
local  0.078GB
> use mydb
switched to db mydb
```

- Check the currently selected database
  - Command: db

```
> show dbs
local  0.078GB
> use mydb
switched to db mydb
> db
mydb
>
```

Create Database

- If the database exists, the `use` command will select the existing database.
- Check the database list
  - Command: show dbs

```
> show dbs
local  0.078GB
> use mydb
switched to db mydb
> db
mydb
> show dbs
local  0.078GB
>
```

  - The created database mydb is not present, you need to insert at least one document into it.

Create Collection

- Create a collection without options
  - Collection name is myCollection

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

- Check the created collection
  - Command: show collections

```
> use mydb
switched to db mydb
> db.createCollection("myCollection")
{ "ok" : 1 }
> show collections
myCollection
system.indexes
>
```

Create Collection

- If you don't create a collection, the MongoDB will automatically create a collection for the inserted document
  - command: db.noCollection.insert({'name' : 'Yu'})

```
> show collections
myCollection
system.indexes
> db.noCollection.insert({'name':'Yu'})
WriteResult({ "nInserted" : 1 })
> show collections
myCollection
noCollection
system.indexes
>
```

Drop Collection

- Drop a collection
  - command: db.collection_name.drop()

```
> show dbs
local  0.078GB
mydb   0.078GB
> use mydb
switched to db mydb
> show collections
myCollection
noCollection
system.indexes
> db.noCollection.drop()
true
> show collections
myCollection
system.indexes
>
```

Drop Database

- Drop a database
  - command: db.dropDatabase()
  - If you want to delete a database, you should select the database, and then delete it.

```
> show dbs
local  0.078GB
mydb   0.078GB
> use mydb
switched to db mydb
> db.dropDatabase()
{ "dropped" : "mydb", "ok" : 1 }
> show dbs
local  0.078GB
>
```

Drop collection and database

- If you drop a collection, the database is still there.
- If you drop a database, the included collections are deleted.

Insert a Document

- Inset a document with a defined _id into a collection
  - command: db.collection_name.insert(document)
- Follow the JSON schema

```
> show collections
myCollection
system.indexes
> db.myCollection.insert(
... { _id : 1,
... title : 'MongoDB',
... description : 'this is a sample document',
... tags : ['MongoDB', { type : 'string'}],
... number : 10000
... }
... )
WriteResult({ "nInserted" : 1 })
> db.myCollection.find()
{ "_id" : 1, "title" : "MongoDB", "description" : "this is a sample document", "tags" : [ "MongoDB", { "type" : "string" } ], "number"
: 10000 }
>
```

Insert a Document

- The document defines the _id

```
> show collections
myCollection
system.indexes
> db.myCollection.insert(
... { _id : 1,
... title : 'MongoDB',
... description : 'this is a sample document',
... tags : ['MongoDB', { type : 'string'}],
... number : 10000
... }
... )
WriteResult({ "nInserted" : 1 })
> db.myCollection.find()
{ "_id" : 1, "title" : "MongoDB", "description" : "this is a sample document", "tags" : [ "MongoDB", { "type" : "string" } ], "number"
: 10000 }
> db.myCollection.findOne()
{
        "_id" : 1,
        "title" : "MongoDB",
        "description" : "this is a sample document",
        "tags" : [
                "MongoDB",
                {
                        "type" : "string"
                }
        ],
        "number" : 10000
}
>
```

Insert a Document

- Insert a document without a defined _id into a collection

```
> show collections
myCollection
system.indexes
> db.myCollection.insert(
... { title : 'MongoDB',
... description : 'this is a sample document',
... tags : ['MongoDB', { type : 'string'}],
... number : 10000
... }
... )
WriteResult({ "nInserted" : 1 })
> db.myCollection.find()
{ "_id" : ObjectId("57da48bdd1f2aa1008f92b45"), "title" : "MongoDB", "description" : "this is a sample document", "tags" : [ "MongoDB", { "type" : "stri
ng" } ], "number" : 10000 }
> db.myCollection.findOne()
{
        "_id" : ObjectId("57da48bdd1f2aa1008f92b45"),
        "title" : "MongoDB",
        "description" : "this is a sample document",
        "tags" : [
                "MongoDB",
                {
                        "type" : "string"
                }
        ],
        "number" : 10000
}
>
```

Query Document

- Query data from a collection
  - Select the correct database
    - command: use database_name
  - Select the correct collection
  - command: db.collection_name.find()

```
> show dbs
local  0.078GB
mydb   0.078GB
> use mydb
switched to db mydb
> show collections
myCollection
system.indexes
> db.myCollection.find()
{ "_id" : ObjectId("57da4b65d1f2aa1008f92b4d"), "title" : 1, "tag" : 2 }
{ "_id" : ObjectId("57da4b6ed1f2aa1008f92b4e"), "title" : 2, "tag" : 1 }
{ "_id" : ObjectId("57da4b77d1f2aa1008f92b4f"), "title" : 3, "tag" : 1 }
>
```

Query Document

- If you want to show the document in a formatted way.
  - Command: db.collection_name.find().pretty()

```
> db.myCollection.find()
{ "_id" : ObjectId("57da4d87d1f2aa1008f92b50"), "title" : 1, "description" : "this is a sample document", "tags" :
[ "MongoDB", { "type" : "string" } ], "number" : 10000 }
> db.myCollection.find().pretty()
{
        "_id" : ObjectId("57da4d87d1f2aa1008f92b50"),
        "title" : 1,
        "description" : "this is a sample document",
        "tags" : [
                "MongoDB",
                {
                        "type" : "string"
                }
        ],
        "number" : 10000
}
>
```

Query Document

- If you just want to show the first document of the collection
  - Command: db.collection_name.findOne()

```
> db.myCollection.find()
{ "_id" : ObjectId("57da4e8fd1f2aa1008f92b51"), "title" : 1 }
{ "_id" : ObjectId("57da4e93d1f2aa1008f92b52"), "title" : 2 }
{ "_id" : ObjectId("57da4e96d1f2aa1008f92b53"), "title" : 3 }
{ "_id" : ObjectId("57da4e99d1f2aa1008f92b54"), "title" : 4 }
> db.myCollection.findOne()
{ "_id" : ObjectId("57da4e8fd1f2aa1008f92b51"), "title" : 1 }
>
```

Query Document

- If you want to find documents with specified values
  - Command: db.collection_name.find(criteria)

```
> db.myCollection.find()
{ "_id" : ObjectId("57daa2a55ad9bbb8816eeab1"), "title" : 1 }
{ "_id" : ObjectId("57daa2a85ad9bbb8816eeab2"), "title" : 2 }
{ "_id" : ObjectId("57daa2b85ad9bbb8816eeab3"), "title" : 3 }
{ "_id" : ObjectId("57daa2dc5ad9bbb8816eeab4"), "title" : 4 }
> db.myCollection.find({title : 2})
{ "_id" : ObjectId("57daa2a85ad9bbb8816eeab2"), "title" : 2 }
>
```

Query Document

- Display the limited numbers of documents
  - Command:

db.collection.find().limit(number)

```
> db.myCollection.find()
{ "_id" : ObjectId("57dac07ff6871edf99b492f3"), "title" : 1 }
{ "_id" : ObjectId("57dac082f6871edf99b492f4"), "title" : 2 }
{ "_id" : ObjectId("57dac085f6871edf99b492f5"), "title" : 3 }
{ "_id" : ObjectId("57dac087f6871edf99b492f6"), "title" : 4 }
{ "_id" : ObjectId("57dac089f6871edf99b492f7"), "title" : 5 }
{ "_id" : ObjectId("57dac08cf6871edf99b492f8"), "title" : 6 }
> db.myCollection.find().limit(3)
{ "_id" : ObjectId("57dac07ff6871edf99b492f3"), "title" : 1 }
{ "_id" : ObjectId("57dac082f6871edf99b492f4"), "title" : 2 }
{ "_id" : ObjectId("57dac085f6871edf99b492f5"), "title" : 3 }
>
```

Query Document

- Sort the documents
  - Command:

  db.collection_name.find().sort({key:1/-1})

  1 is used for ascending order

  -1 is used for descending order

```
> db.myCollection.find()
{ "_id" : ObjectId("57dac07ff6871edf99b492f3"), "title" : 1 }
{ "_id" : ObjectId("57dac082f6871edf99b492f4"), "title" : 2 }
{ "_id" : ObjectId("57dac085f6871edf99b492f5"), "title" : 3 }
{ "_id" : ObjectId("57dac087f6871edf99b492f6"), "title" : 4 }
{ "_id" : ObjectId("57dac089f6871edf99b492f7"), "title" : 5 }
{ "_id" : ObjectId("57dac08cf6871edf99b492f8"), "title" : 6 }
> db.myCollection.find().sort({title:-1}).limit(3)
{ "_id" : ObjectId("57dac08cf6871edf99b492f8"), "title" : 6 }
{ "_id" : ObjectId("57dac089f6871edf99b492f7"), "title" : 5 }
{ "_id" : ObjectId("57dac087f6871edf99b492f6"), "title" : 4 }
> db.myCollection.find().sort({title:1}).limit(3)
{ "_id" : ObjectId("57dac07ff6871edf99b492f3"), "title" : 1 }
{ "_id" : ObjectId("57dac082f6871edf99b492f4"), "title" : 2 }
{ "_id" : ObjectId("57dac085f6871edf99b492f5"), "title" : 3 }
>
```

Query Document

- To query a document on the basis of some conditions, you can use following operations.

Equality	{ key : value }	db.myCollection.find({ number: 1})	Where number is 1
Less than	{ key : { $ls : value} }	db.myCollection.find({ number: {$ls : 1}})	Where number is less than 1
Less then and equals	{ key : { $lse : value} }	db.myCollection.find({ number: {$lse : 1}})	Where number is less and equals to 1
Greater than	{ key : { $gt : value} }	db.myCollection.find({ number: {$gt : 1}})	Where number is greater than 1
Greater than and equals	{ key : { $gte : value} }	db.myCollection.find({ number: {$get : 1}})	Where number is greater than and equals to 1
Not equals	{ key : {$ne : value} }	db.myCollection.find({ number: {$ne : 1}})	Where number is not 1

Query Document

```
> db.myCollection.find()
{ "_id" : ObjectId("57daa2a55ad9bbb8816eeab1"), "title" : 1 }
{ "_id" : ObjectId("57daa2a85ad9bbb8816eeab2"), "title" : 2 }
{ "_id" : ObjectId("57daa2b85ad9bbb8816eeab3"), "title" : 3 }
{ "_id" : ObjectId("57daa2dc5ad9bbb8816eeab4"), "title" : 4 }
> db.myCollection.find({title : {$lt : 2}})
{ "_id" : ObjectId("57daa2a55ad9bbb8816eeab1"), "title" : 1 }
> db.myCollection.find({title : {$lte : 2}})
{ "_id" : ObjectId("57daa2a55ad9bbb8816eeab1"), "title" : 1 }
{ "_id" : ObjectId("57daa2a85ad9bbb8816eeab2"), "title" : 2 }
> db.myCollection.find({title : {$gt : 2}})
{ "_id" : ObjectId("57daa2b85ad9bbb8816eeab3"), "title" : 3 }
{ "_id" : ObjectId("57daa2dc5ad9bbb8816eeab4"), "title" : 4 }
> db.myCollection.find({title : {$gte : 2}})
{ "_id" : ObjectId("57daa2a85ad9bbb8816eeab2"), "title" : 2 }
{ "_id" : ObjectId("57daa2b85ad9bbb8816eeab3"), "title" : 3 }
{ "_id" : ObjectId("57daa2dc5ad9bbb8816eeab4"), "title" : 4 }
> db.myCollection.find({title : {$ne : 2}})
{ "_id" : ObjectId("57daa2a55ad9bbb8816eeab1"), "title" : 1 }
{ "_id" : ObjectId("57daa2b85ad9bbb8816eeab3"), "title" : 3 }
{ "_id" : ObjectId("57daa2dc5ad9bbb8816eeab4"), "title" : 4 }
>
```

Query Document

- And in Query method
  - Command:

db.collection_name.find({key1 : value, key2 : value2})

```
> db.myCollection.find()
{ "_id" : ObjectId("57daa8945ad9bbb8816eeab5"), "title" : 1, "tag" : 1 }
{ "_id" : ObjectId("57daa8985ad9bbb8816eeab6"), "title" : 1, "tag" : 2 }
{ "_id" : ObjectId("57daa8a15ad9bbb8816eeab7"), "title" : 2, "tag" : 1 }
{ "_id" : ObjectId("57daa8a45ad9bbb8816eeab8"), "title" : 2, "tag" : 2 }
{ "_id" : ObjectId("57daa8ac5ad9bbb8816eeab9"), "title" : 3, "tag" : 1 }
{ "_id" : ObjectId("57daa8b25ad9bbb8816eeaba"), "title" : 3, "tag" : 2 }
>
```

Query Document

- Query document with And conditions
  - title < 2 AND tag >= 1
  - title < 2 AND tag > 1

```
> db.myCollection.find()
{ "_id" : ObjectId("57daa8945ad9bbb8816eeab5"), "title" : 1, "tag" : 1 }
{ "_id" : ObjectId("57daa8985ad9bbb8816eeab6"), "title" : 1, "tag" : 2 }
{ "_id" : ObjectId("57daa8a15ad9bbb8816eeab7"), "title" : 2, "tag" : 1 }
{ "_id" : ObjectId("57daa8a45ad9bbb8816eeab8"), "title" : 2, "tag" : 2 }
{ "_id" : ObjectId("57daa8ac5ad9bbb8816eeab9"), "title" : 3, "tag" : 1 }
{ "_id" : ObjectId("57daa8b25ad9bbb8816eeaba"), "title" : 3, "tag" : 2 }
> db.myCollection.find({title : {$lt : 2}, tag : {$gte : 1}})
{ "_id" : ObjectId("57daa8945ad9bbb8816eeab5"), "title" : 1, "tag" : 1 }
{ "_id" : ObjectId("57daa8985ad9bbb8816eeab6"), "title" : 1, "tag" : 2 }
> db.myCollection.find({title : {$lt : 2}, tag : {$gt : 1}})
{ "_id" : ObjectId("57daa8985ad9bbb8816eeab6"), "title" : 1, "tag" : 2 }
>
```

Query Document

- OR in Query method
  - Command:

```
db.collection_name.find(
    {$or : [
        {key1 : value},  {key2 : value2}
        ]
    }
)
```

Query Document

- Query documents with OR conditions
  - title <= 2 OR tag < 2

```
> db.myCollection.find({$or : [{title : {$lte : 2}}, {tag : {$lt : 2}}]})
{ "_id" : ObjectId("57daa8945ad9bbb8816eeab5"), "title" : 1, "tag" : 1 }
{ "_id" : ObjectId("57daa8985ad9bbb8816eeab6"), "title" : 1, "tag" : 2 }
{ "_id" : ObjectId("57daa8a15ad9bbb8816eeab7"), "title" : 2, "tag" : 1 }
{ "_id" : ObjectId("57daa8a45ad9bbb8816eeab8"), "title" : 2, "tag" : 2 }
{ "_id" : ObjectId("57daa8ac5ad9bbb8816eeab9"), "title" : 3, "tag" : 1 }
>
```

Query Document

- Query documents with AND and OR conditions
  - tag : 1 AND (title = 1 OR title = 2)

```
> db.myCollection.find({tag : 1, $or : [{title : 1}, {title : 2}]})
{ "_id" : ObjectId("57daa8945ad9bbb8816eeab5"), "title" : 1, "tag" : 1 }
{ "_id" : ObjectId("57daa8a15ad9bbb8816eeab7"), "title" : 2, "tag" : 1 }
>
>
```

Update a Document

- Update the values in the document
  - Command:

  db.collection_name.update({key:value}, {$set : {key:value}})

```
> db.myCollection.find()
{ "_id" : 1, "title" : "title 1" }
{ "_id" : 2, "title" : "title 2" }
{ "_id" : 3, "title" : "title 3" }
{ "_id" : 4, "title" : "title 4" }
> db.myCollection.update({_id : 1}, {$set : {title : 'update title 1'}})
> db.myCollection.find()
{ "_id" : 2, "title" : "title 2" }
{ "_id" : 3, "title" : "title 3" }
{ "_id" : 4, "title" : "title 4" }
{ "_id" : 1, "title" : "update title 1" }
>
```

Update a Document

- Update the values in the document
  - Command:

  db.collection_name.update({key:value}, {$set : {key:value}})

```
> db.myCollection.find()
{ "_id" : 1, "title" : "title 1" }
{ "_id" : 2, "title" : "title 2" }
{ "_id" : 3, "title" : "title 3" }
{ "_id" : 4, "title" : "title 4" }
> db.myCollection.update({_id : 1}, {$set : {title : 'update title 1'}})
> db.myCollection.find()
{ "_id" : 2, "title" : "title 2" }
{ "_id" : 3, "title" : "title 3" }
{ "_id" : 4, "title" : "title 4" }
{ "_id" : 1, "title" : "update title 1" }
>
```

Delete a Document

- Delete a document from a collection
  - command: db.collection_name.remove()
  - The remove() method accepts two parameters
    - deletion criteria: the documents will be removed if they satisfy the criteria
    - justOne: if set to true or 1, the method only removes the first document.

Delete a Document

- Delete a document from a collection
  - command: db.collection_name.remove()
  - The remove() method accepts two parameters
    - deletion criteria: the documents will be removed if they satisfy the criteria
    - justOne: if set to true or 1, the method only removes the first document.

Delete a Document

- Delete a document from a collection
  - command: db.collection_name.remove()
  - The remove() method accepts two parameters
    - deletion criteria: the documents will be removed if they satisfy the criteria
    - justOne: if set to true or 1, the method only removes the first document.

ToDo

- Go to the Online MongoDB shell and enter the commands you saw today.
  - You will need to insert some extra data from slide 16 onward.