# MongoDB Lab Assignments -Day 1

**MongoDB Exercise in mongo shell**

Connect to a running mongo instance, use a database named **mongo\_practice**. Document all your queries in a javascript file to use as a reference.

C:\Users\akcha>mongosh "mongodb+srv://cluster0.lfos3.mongodb.net/demo" --username SC-Test

Enter password: \*\*\*\*\*

Current Mongosh Log ID: 61dd461801baa5fa2e138f5a

Connecting to: mongodb+srv://cluster0.lfos3.mongodb.net/demo

Using MongoDB: 4.4.11

Using Mongosh: 1.1.7

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-pa1ndl-shard-0 [primary] demo> show dbs

admin 336 kB

local 4.21 GB

Atlas atlas-pa1ndl-shard-0 [primary] demo> use mongodb\_practice

switched to db mongodb\_practice

# Insert Documents

Insert the following documents into a **movies** collection.

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.createCollection('movies')

{ ok: 1 }

title : Fight Club

writer : Chuck Palahniuko year : 1999

actors : [ Brad Pitt

Edward Norton

]

title : Pulp Fiction

writer : Quentin Tarantino year : 1994

actors : [

John Travolta Uma Thurman

]

title : Inglorious Basterds writer : Quentin Tarantino year : 2009

actors : [ Brad Pitt

Diane Kruger Eli Roth

]

title : The Hobbit: An Unexpected Journey writer : J.R.R. Tolkein

year : 2012

franchise : The Hobbit

title : The Hobbit: The Desolation of Smaug writer : J.R.R. Tolkein

year : 2013

franchise : The Hobbit

title : The Hobbit: The Battle of the Five Armies writer : J.R.R. Tolkein

year : 2012

franchise : The Hobbit

synopsis : Bilbo and Company are forced to engage in a war against an array of combatants and keep the Lonely Mountain from falling into the hands of a rising darkness.

title : Pee Wee Herman's Big Adventure title : Avatar

Reference https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_insert_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_insert\_document.htm](http://www.tutorialspoint.com/mongodb/mongodb_insert_document.htm)

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.movies.insertMany([{title:'Fight Club', writer: 'Chuck Palahniuko', year:1999, actors:['Brad Pitt','Edwrd Norton']}, {title:'Pulp Fiction',writer:'Quentin Tarantino',year:1994,actors:['John Travolta','Uma Thurman']},{title: 'Inglorious Basterds', writer: 'Quentin Tarantino', year: 2009, actors: ['Brad Pitt','Diane Kruger','Eli Roth']}, {title: 'The Hobbit: An Unexpected Journey', writer: 'J.R.R Tolkein', year: 2012, franchise: 'The Hobbit'}, {title: 'The Hobbit: The Desolation of Smaug', writer: 'J.R.R Tolkein', year: 2013, franchise: 'The Hobbit'}, {title: 'The Hobbit: The Battle of the Five Armies', writer: 'J.R.R Tolkein', year: 2012, franchise: 'The Hobbit', synopsis: 'Bilbo and Company are forced to engage in a war against array of combatants and keep the Lonely Mountain from falling into the hands of a rising darkness'}, {title:"Pee Wee Herman's Big Adventure"},{title: 'Avatar'}])

{

acknowledged: true,

insertedIds: {

'0': ObjectId("61dd576452c5e516304fd8e6"),

'1': ObjectId("61dd576452c5e516304fd8e7"),

'2': ObjectId("61dd576452c5e516304fd8e8"),

'3': ObjectId("61dd576452c5e516304fd8e9"),

'4': ObjectId("61dd576452c5e516304fd8ea"),

'5': ObjectId("61dd576452c5e516304fd8eb"),

'6': ObjectId("61dd576452c5e516304fd8ec"),

'7': ObjectId("61dd576452c5e516304fd8ed")

}

}

# Query / Find Documents

query the **movies** collection to

1. get all documents

db.movies.find()

1. get all documents with writer set to "Quentin Tarantino"

db.movies.find({writer:"Quentin Tarantino"})

1. get all documents where actors include "Brad Pitt"

db.movies.find({actors:"Brad Pitt"})

1. get all documents with franchise set to "The Hobbit"

db.movies.find({franchise:"The Hobbit"})

1. get all movies released in the 90s

db.movies.find({year:{$gt:1990, $lt:2000}})

1. get all movies released before the year 2000 or after 2010
2. db.movies.find({$or:[{year:{$gt:2010}},{year: {$lt:2000}}]})

Reference: https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_query_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_query\_document.htm](http://www.tutorialspoint.com/mongodb/mongodb_query_document.htm)

# Update Documents

1. add a synopsis to "The Hobbit: An Unexpected Journey" : "A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.movies.update({title:'The Hobbit: An Unexpected Journey'}, {$set:{synopsis:"A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."}})

DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

1. add a synopsis to "The Hobbit: The Desolation of Smaug" : "The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of a mysterious and magical ring."

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.movies.update({title:'The Hobbit: The Desolation of Smaug'}, {$set:{synopsis:"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of a mysterious and magical ring."}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

1. add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.movies.update({title:'Pulp Fiction'}, {$push:{actors:"Samuel L. Jackson"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

Reference: https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_update_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_update\_document.htm](http://www.tutorialspoint.com/mongodb/mongodb_update_document.htm)

# Text Search

1. find all movies that have a synopsis that contains the word "Bilbo"

db.movies.find({synopsis:{$regex:"Bilbo"}})

1. find all movies that have a synopsis that contains the word "Gandalf"

db.movies.find({synopsis:{$regex:"Gandalf"}})

1. find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"

db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]})

1. find all movies that have a synopsis that contains the word "dwarves" or "hobbit"

db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$r$regex:"hobbit"}}]})

1. find all movies that have a synopsis that contains the word "gold" and "dragon"

db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$reg$regex:"dragon"}}]})

Reference: https[://w](http://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)ww[.t](http://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)u[torialspoint.com/mongodb/mongodb\_text\_search.htm](http://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)

# Delete Documents

1. delete the movie "Pee Wee Herman's Big Adventure"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.movies.remove({title:"Pee Wee Herman's Big Adventure"})

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice>

{ acknowledged: true, deletedCount: 1 }

1. delete the movie "Avatar"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.movies.remove({title:"Avatar"})

{ acknowledged: true, deletedCount: 1 }

Reference: https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_delete_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_delete\_document.htm](http://www.tutorialspoint.com/mongodb/mongodb_delete_document.htm)

# Relationships

Insert the following documents into a **users** collection

username : GoodGuyGreg first\_name : "Good Guy" last\_name : "Greg" username : ScumbagSteve full\_name :

first : "Scumbag" last : "Steve"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.users.insertMany([{ username: 'GoodGuyGreg', first\_name: 'GoodAtlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice>

{

acknowledged: true,

insertedIds: {

'0': ObjectId("61dd653167bb55bd7dd40551"),

'1': ObjectId("61dd653167bb55bd7dd40552")

}

}

Insert the following documents into a **posts** collection

username : GoodGuyGreg title : Passes out at party

body : Wakes up early and cleans house

username : GoodGuyGreg title : Steals your identity

body : Raises your credit score

username : GoodGuyGreg

title : Reports a bug in your code body : Sends you a Pull Request

username : ScumbagSteve title : Borrows something body : Sells it

username : ScumbagSteve title : Borrows everything body : The end

username : ScumbagSteve

title : Forks your repo on github body : Sets to private

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.posts.insertMany([{username: 'GoodGuyGreg', title: 'Passes out at party', body: 'Wakes up early and cleans house'},{username: 'GoodGuyGreg', title: 'Steals your identity', body: 'Raises your credit score'},{username: 'GoodGuyGreg', title: 'Reports a bug in your code', body: 'Sends you a pull request'},{username: 'ScumbagSteve', title: 'Borrows something', body: 'Sells it'},{username: 'ScumbagSteve', title: 'Borrows everything', body: 'The end'},{username: 'ScumbagSteve', title: 'Forks your repo in github', body: 'Sets to private'}])

{

acknowledged: true,

insertedIds: {

'0': ObjectId("61dd676367bb55bd7dd40553"),

'1': ObjectId("61dd676367bb55bd7dd40554"),

'2': ObjectId("61dd676367bb55bd7dd40555"),

'3': ObjectId("61dd676367bb55bd7dd40556"),

'4': ObjectId("61dd676367bb55bd7dd40557"),

'5': ObjectId("61dd676367bb55bd7dd40558")

}

}

Insert the following documents into a **comments** collection

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.createCollection('comments')

{ ok: 1 }

username : GoodGuyGreg

comment : Hope you got a good deal! post : [post\_obj\_id]

where [post\_obj\_id] is the ObjectId of the posts document: "Borrows something"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.comments.insert({username:'GoodGuyGreg', comment: 'Hope you got a good deal!', post:ObjectId("61dd676367bb55bd7dd40556")})

DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.

{

acknowledged: true,

insertedIds: { '0': ObjectId("61dd696167bb55bd7dd40559") }

}

username : GoodGuyGreg comment : What's mine is yours! post : [post\_obj\_id]

where [post\_obj\_id] is the ObjectId of the posts document: "Borrows everything"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.comments.insert({username:'GoodGuyGreg', comment: 'Hope you got a good deal!', post:ObjectId("61dd676367bb55bd7dd40557")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61dd69ea67bb55bd7dd4055a") }

}

username : GoodGuyGreg

comment : Don't violate the licensing agreement! post : [post\_obj\_id]

where [post\_obj\_id] is the ObjectId of the posts document: "Forks your repo on github

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice>>db.comments.insert({username: 'GoodGuyGreg', comment: "What's mine is yours!", post:ObjectId("61dd676367bb55bd7dd40558")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61dd6a7767bb55bd7dd4055b") }

}

username : ScumbagSteve comment : It still isn't clean post : [post\_obj\_id]

where [post\_obj\_id] is the ObjectId of the posts document: "Passes out at party"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.comments.insert({username: 'ScumbagSteve', comment: "It still isn't clean", post:ObjectId("61dd676367bb55bd7dd40553")})

DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.

{

acknowledged: true,

insertedIds: { '0': ObjectId("61dd6cbeb17221b450ed8810") }

}

username : ScumbagSteve

comment : Denied your PR cause I found a hack post : [post\_obj\_id]

where [post\_obj\_id] is the ObjectId of the posts document: "Reports a bug in your

code"

Atlas atlas-pa1ndl-shard-0 [primary] mongodb\_practice> db.comments.insert({username: 'ScumbagSteve', comment: "Denied your PR cause I found a hack",post:ObjectId("61dd676367bb55bd7dd40555")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61dd6d34b17221b450ed8811") }

}

# Querying related collections

1. find all users

db.users.find().pretty()

1. find all posts

db.posts.find().pretty()

1. find all posts that was authored by "GoodGuyGreg"

db.posts.find({username:"GoodGuyGreg"})

1. find all posts that was authored by "ScumbagSteve"

db.posts.find({username:"ScumbagSteve"})

1. find all comments

db.comments.find().pretty()

1. find all comments that was authored by "GoodGuyGreg"

db.comments.find({username:"GoodGuyGreg"})

1. find all comments that was authored by "ScumbagSteve"

db.comments.find({username:'ScumbagSteve'})

1. find all comments belonging to the post "Reports a bug in your code"

db.comments.find({post:ObjectId("61dd676367bb55bd7dd40555”)})

References: <https://docs.mongodb.com/manual/reference/method/db.collection.find/>

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@