**MongoDB Lab Assignments -Day 1**

**-By Abhishek Sharma**

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.

**Query / Find Documents**

query the movies collection to

**Q1. get all documents**

Ans.

MongoshInvalidInputError: [COMMON-10001] 'collection()' is not a valid argument for "show".

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find()

[

{

\_id: ObjectId("61ed664022b7eaad130be19e"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: '1999',

actors: [ 'Brad Pitt', 'Edward Norton' ]

},

{

\_id: ObjectId("61ed664022b7eaad130be19f"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: '1994',

actors: [ 'John Travolta', 'Uma Thurman' ]

},

{

\_id: ObjectId("61ed664022b7eaad130be1a0"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: '2009',

actors: [ 'Bradpitt', 'Diane Kruger', 'Eli Roth' ]

},

{

\_id: ObjectId("61ed664022b7eaad130be1a1"),

title: 'The Hobbit:An Unexpected Journey',

writer: 'J.R.R. Tolkein',

year: '2012',

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61ed664022b7eaad130be1a2"),

title: 'The Desolation of Smaug',

writer: 'J.R.R Tolkein',

year: '2013',

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61ed664022b7eaad130be1a3"),

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 fron falling into the hands of a rising darkness.'

},

{

\_id: ObjectId("61ed664022b7eaad130be1a4"),

title: "Pee Wee Herman's Big Adventure"

},

{ \_id: ObjectId("61ed664022b7eaad130be1a5"), title: 'Avatar' }

]

**Q.2. get all documents with writer set to "Quentin Tarantino"**

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({"writer":"Quentin Tarantino"})

[

{

\_id: ObjectId("61ed664022b7eaad130be19f"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: '1994',

actors: [ 'John Travolta', 'Uma Thurman' ]

},

{

\_id: ObjectId("61ed664022b7eaad130be1a0"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: '2009',

actors: [ 'Bradpitt', 'Diane Kruger', 'Eli Roth' ]

}

]

Q.3**. get all documents where actors include "Brad Pitt"**

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({"actors":"Brad Pitt"})

[

{

\_id: ObjectId("61ed664022b7eaad130be19e"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: '1999',

actors: [ 'Brad Pitt', 'Edward Norton' ]

}

]

**Q. 4. get all documents with franchise set to "The Hobbit"**

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({"franchise":"The Hobbit"})

**Text Search**

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.findOne({"synopsis":{$regex:"Bilbo"}})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({"synopsis":{$regex:"Gandalf"}})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({$and:[{"synopsis":{$regex:"Bilbo"}},{"synopsis":{$not:{$regex:"Gandalf"}}}]})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({$or:[{"synopsis":{$regex:"dwarves"}},{"synopsis":{$regex:"hobbit"}}]})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.mongo\_practice.find({$and:[{"synopsis":{$regex:"gold"}},{"synopsis":{$regex:"dragon"}}]})

**Delete Documents**

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.movies.deleteOne({"title":"Pee Wee Herman's Big Adventure"})

{ acknowledged: true, deletedCount: 1 }

2. delete the movie "Avatar"

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.movies.deleteOne({"title":"Avatar"})

{ acknowledged: true, deletedCount: 1 }

**Querying related collections**

1. find all users

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.users.find()

2. find all posts

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.posts.find()

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.posts.find({username:"GoodGuyGreg"})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.posts.find({username:"ScumbagSteve"})

5. find all comments

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.comments.find()

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.comments.find({username:"GoodGuyGreg"})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.comments.find({username:"ScumbagSteve"})

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

Atlas atlas-45a6ou-shard-0 [primary] mongo\_pratice> db.comments.find({post:db.posts.find({title:"Reports a bug in your code"}).\_id})